UNIVERSIDAD DE DEUSTO

DESARROLLO DE MÉTRICAS PARA LA CLASIFICACIÓN SUPERVISADA EN APLICACIONES CON DATOS EMPÍRICOS

Tesis doctoral presentada por Julio Manuel Revilla Ocejo

Programa de Doctorado en Informática

Dirigida por Dr. Evaristo Kahoraho Bukubiye

El Doctorando | El Director

Julio Manuel Revilla Ocejo | Dr. Evaristo Kahoraho Bukubiye

Bilbao, julio de 2015

Resumen

El objetivo de esta tesis es el estudio y desarrollo de nuevas métricas que amplíen la forma tradicional de medir la distancia entre casos, siendo de aplicación en algoritmos de clasificación como *k*-NN y, plausiblemente, en otros como las máquinas de vectores de soporte (SVM).

Su aplicación está ligada a problemas de clasificación en los que los casos se puedan expresar mediante un conjunto de atributos, los cuales serán representados como colecciones de valores numéricos (vectores). Partiendo de un conjunto de casos empíricos previamente bien clasificados, se elegirán u optimizarán los parámetros de la métrica, la cual será posteriormente aplicada para calcular la distancia entre dos casos cualesquiera (y así poder evaluar la similitud entre ellos).

En esta tesis se ha diseñado en primer lugar una métrica, denominada BTW, inducida por una transformación. Propone una misma expresión matemática para realizar el cálculo de las distancias en todo el espacio de los atributos (es pues una métrica global). También se ha propuesto otra métrica, a la que se le ha puesto por nombre LOM, cuyo cálculo de la distancia depende de un conocimiento "a priori", y en el que dicho cálculo varía dependiendo de la localización de ambos casos (es una métrica local).

Estas nuevas métricas estarán orientadas, bien a disminuir el tiempo que se tarda en buscar los vecinos más próximos a uno dado y la cantidad de memoria requerida para almacenar la información que permite realizar esta tarea (métrica BTW); bien a mejorar la precisión de los algoritmos de clasificación (métrica LOM).

Las prestaciones de la métrica BTW y su algoritmo asociado han sido evaluadas mediante problemas reales que nos permiten asegurar que, manteniendo una alta precisión en la clasificación, se consigue reducir el espacio de almacenamiento y el tiempo necesario para encontrar los vecinos más próximos de un caso dado.

Para la métrica LOM se ha diseñado un problema sintético que, resuelto mediante un algoritmo de clasificación basado en los vecinos más próximos, permite comprobar una mejora en la precisión de la clasificación (al comparar sus prestaciones respecto al mismo algoritmo cuando usa una métrica euclídea).

Abstract

The objective of this thesis is the study and development of new metrics which modify the traditional method of measuring the distance between two cases. They could be used in classification algorithms such as k-NN and, probably, in some others like Support Vector Machines (SVM).

Its application is bound to classification problems in which the cases can be expressed by a collections of attributes (that are represented as vectors of numerical values). From a set of well-classified empirical cases, the metric parameters will be optimized, and this metric will be later applied to calculate the distance between any two other cases (and this way, evaluate the similarity of the two cases).

In this thesis, first of all, a metric induced by a transformation, called BTW, has been designed. This metric poses the same mathematical expression to calculate the distances in all the attribute's space (it is a global metric). Secondly, a metric called LOM is also proposed, its distance calculation depends on a beforehand knowledge, and distances between cases vary depending on the location of both (it is a local metric).

These new metrics will be oriented on the one hand (BTW metric), to decrease the time to find near neighbors and reduce the required memory to carry out this work. On the other hand (LOM metric), to improve the precision of the classification algorithms.

The benefits of the BTW metric have been evaluated with real problems. Their analysis allow us to claim that, maintaining a high precision in the classification, both, the storage and time to find the nearest neighbors to a given case, are reduced.

For the LOM metric, a synthetic problem has been designed. This problem, solved by a nearest neighbor classification algorithm, lets us validate an improvement in the precision of the classification (when comparing it to the results of the same algorithm using a Euclidean metric).

Laburpena

Tesi honen helburua, kasuen arteko distantzia neur dezaketen edota k-NN motako sailkapen algoritmoetan, eta ziur aski oinarri bektorezko makinetan (SVM), aplikagarriak diren metrika berriak ikertu eta garatzea izan da.

Metrika berri hauen aplikazioa, kasuak ezaugarri multzo baten bidez adierazi daitezkeen sailkapen arazoei lotuta dago. Ezaugarri hauek, balio numeriko sorta gisa azaltzen dira (bektoreak). Aurretik sailkatutako kasu enpiriko multzo batez abiatuz, metrikaren parametroak optimizatuko dira. Geroago, metrika hau, edozein bi kasuen arteko distantzia neurtzeko aplikatuko da (modu honetan, bi kasuen arteko antzekotasuna ebaluatu ahal izango da).

Tesi honetan, lehenik eta behin, transformazio batez induzitutako metrika diseinatu da, BTW izenekoa. Metrika honek, adierazpen berdina erabiltzen du espazio osoan, ezaugarrien arteko distantzia neurtzeko (metrika orokorra). Bestalde, beste bat proposatu da, LOM izenekoa, non bere distantzia neurtzeko kalkulua "aurre- ezagutza" batetan oinarritu eta kasuen arteko distantzia, ezaugarrien balioen arabera aldatzen den (tokiko metrika).

Metrika berri hauek, helburu bikoitza dute. Alde batetik, balio baten ondokoena aurkitzeko denbora eta beharrezkoa den informazioa biltzeko memoria espazioa murriztea bilatuko da (BTW metrika). Bestetik, sailkapen algoritmoen zehaztasuna hobetzea (LOM metrika).

BTW metrikaren prestakuntzak, benetako arazoen bidez aztertu dira. Hortaz, sailkapenean zehaztasun handia mantenduz, metatze-espazioa eta "auzotarrak" bilatzeko denbora murrizten dela ondoriozta dezakegu.

LOM metrikarako, arazo sintetiko bat diseinatu da, non ondokoen "auzotarretan" oinarritutako sailkapen algoritmo baten bidez, sailkapen honen zehaztasun hobekuntza bat erakusten digun (algoritmo bera metrika euklidearekin erabiltzean lortutako emaitzekin konparatuz).

Agradecimientos

Esta tesis no hubiera visto la luz sin el esfuerzo de muchas personas, algunas que se citarán a continuación y otras muchas anónimas, a las que quiero desde aquí agradecer su esfuerzo y apoyo.

En primer lugar a mi director de tesis, el Dr. Evaristo Kahoraho Bukubiye, sin cuya infinita paciencia, exigencia, consejos y buen hacer, esta disertación nunca se hubiera dado.

También quiero agradecer el continuo aliento del Dr. Javier García Zubía, sin el cual quizás esto nunca hubiera llegado a buen puerto; este agradecimiento tendrá que compartirlo con todos los responsables y compañeros de Departamento, siempre con una palabra de ánimo cuando más la necesitaba.

Agradezco a la Universidad de Deusto el respaldo ofrecido durante estos años, y los medios y recursos que han dispuesto para que haya podido ser realizada.

Un recuerdo cariñoso es para mi primer "jefe", Iñaki Yañez Cortés, y resto de compañeros de IK4-Tekniker, donde me inculcaron la pasión por la informática y la inteligencia artificial (que continua viva después de casi treinta años).

Un apartado especial de gratitud es para mi familia, mi esposa Begoña, mis hijas Aintzane y Eneritz y mi hermano Javier. Las horas que les ha quitado esta tesis no habrá forma de compensarlas de forma alguna; su sacrificio, ánimo y comprensión durante estos años han permitido que esta tesis haya ido hacia adelante. Por último, a mis padres, Julio y Gloria, que por muy poco no pudieron ver la culminación de su enorme esfuerzo vital para conseguir que sus hijos "tuvieran estudios".

Tabla de contenidos

Índice de figuras

Motivación personal del doctorando

Corría el final de los años ochenta del siglo pasado cuando yo participaba, como ingeniero novel, en el desarrollo y puesta en marcha del sistema de fabricación flexible (FMS) que el centro de investigación Tekniker (actualmente IK4-Tekniker) diseñaba para ser explotado por Fatronik System (actualmente parte de Tecnalia Research & Innovation) en la localidad guipuzcoana de Elgoibar.

Mi superior de aquella época, Iñaki Yáñez, uno de los pioneros en este país de la informática industrial y extraordinaria persona a la que siempre estaré agradecido, decidió que, aparte del sistema de comunicaciones industriales, me hiciera cargo también del "diagnóstico" de aquella célula flexible.

Para mí significó un auténtico reto; la solución al problema de las comunicaciones entre ordenadores y máquinas, problema que era secuencial y determinista[1], no era aplicable en el nuevo escenario. El conjunto de datos que se podía recoger de una máquina era muy limitado (ya que no había sensores especialmente dedicados a ello), asociar las distintas combinaciones de los valores de estos sensores a diagnósticos concretos no era sencillo: no existía experiencia previa, el desvío de un sensor de su "habitual" valor no se correspondía siempre con el mismo problema, cada día aparecían más y más situaciones a diagnosticar…

Era la época en la que estaban apareciendo los "Sistemas expertos" basados en reglas (con "Lisp" a la cabeza de los lenguajes que explotaban este paradigma). Por otra parte, los científicos japoneses propugnaban que el lenguaje "Prolog" iba a ser la base de su inmediata "quinta generación" de dispositivos informáticos (que convertiría las máquinas en entes que, más que calcular, se comportarían de forma inteligente. Revolución que, como conocemos, nunca llegó a plasmarse).

Aquel reto que por falta de financiación no se pudo plasmar convenientemente en el FMS citado, supuso para mí la fascinación por un nuevo campo completamente desconocido, y al que dedicaría muchos cientos de horas de estudio en las siguientes décadas.

[1] Y cuyos criterios de calidad en la ejecución eran, aparte del trasvase de datos, no dejar de contemplar ninguna de las múltiples posibilidades que pudieran darse (en caso de que algo no transcurriese de acuerdo a lo "habitual" en la comunicación), ofreciendo una respuesta que permitiera recuperar el flujo fiable de información lo antes posible.

El inicio del nuevo siglo supuso mi llegada a la Universidad de Deusto y el planteamiento de realizar una tesis doctoral cuyo campo iba a estar relacionado con aquel interés que permanecía latente.

La perspectiva con la que se contempla una herramienta en un buen número de sectores de la industria y en la Universidad es completamente diferente. Mientras que en muchas de las industrias que nos rodean el objetivo es que las "cosas funcionen lo suficientemente bien", con elementos ya probados, y sin incurrir en grandes gastos; la Universidad estudia con detalle el estado del arte, las partes a mejorar, dónde están las fronteras del conocimiento y de su aplicabilidad... Mi primer planteamiento fue el realizar un sistema experto de diagnóstico industrial (aunque ya en esa época su interés empezaba a decaer) basado en algún algoritmo de tipo *k*-NN cuya aplicabilidad general lo hiciese atractivo en el mundo industrial. A cabo de unos cuantos meses me di cuenta que este enfoque, como tema de investigación, era más bien "pobre" y que había que volver a replantearse los objetivos.

La literatura científica de aquel momento, tal como se puede apreciar en el siguiente recuadro, apuntaba a las métricas empleadas en los algoritmos de clasificación y a su optimización como las "fronteras del conocimiento" y hacía ahí se redirigieron los estudios.

"An area that requires further study is in fast data-based methods for choosing appropriate distance measures, variable selection and the appropriate number of neighbors." [6].

"…, the choice of a distance measure becomes crucial in determining the outcome of nearest neighbor classification." [53].

"…The commonly used Euclidean distance measure, while simple computationally, implies that the input space is isotropic. However, the assumption for isotropy is often invalid and generally undesirable in many practical applications." [54].

"Proximity based classifiers such as RBF-networks and nearest-neighbor classifiers are notoriously sensitive to the metric used to determine distance between samples…" [55].

"By the very nature of its decision rule, the performance of kNN classification depends crucially on the way that distances are computed between different examples. When no prior knowledge is available, most implementations of kNN compute simple Euclidean distances (assuming the examples are represented as vector inputs). Unfortunately, Euclidean distances ignore any statistical regularities that might be estimated from a large training set of labeled examples." [56].

> *"… the improved performance with multiple metrics suggest that LMNN classification could benefit from even more adaptive transformations of the input space …" [56].*
>
> *"The close relationship between NN classification and density estimation suggests that an analogous performance improvement in classification error may also be obtained by the proper choice of a distance measure…" [17].*

Por otra parte, el interés de los sistemas de clasificación y decisión automática se ha incrementado notablemente en los últimos años. Los actuales sistemas de minería de datos, el "Big Data" y, en un futuro próximo, la Internet de las cosas son y serán grandes demandantes de algoritmos que procesen, clasifiquen e infieran conclusiones partiendo de datos empíricos.

Después de finalizada la tesis, y aún con mucho campo para investigar por delante, esta decisión la juzgo acertada. Aunque en este campo de la informática existen múltiples aspectos cuyo estudio resulta apasionante, las distancias, la "geometría del espacio" que permita obtener las mejores prestaciones y su optimización cumplen con las expectativas que vislumbré hace casi tres décadas.

1 Objetivo y desarrollo de la presente tesis

La rama de la informática conocida como Inteligencia Artificial (IA), cuyo nombre le fue asignado por John McCarthy [1] en 1955, intenta crear entidades que sean capaces de resolver cuestiones por sí mismas empleando como paradigma el modo de pensar y actuar de los seres humanos.

Dentro de la Inteligencia Artificial existen múltiples campos de estudio relacionados con la percepción, el razonamiento, la actuación autónoma… Probablemente la IA esté llamada a ser una de las ramas de la ciencia con mayor porvenir en el siglo XXI, pero presenta tal cantidad de problemas abiertos que requerirá de ingentes esfuerzos en investigación en las próximas décadas si se desea conseguir el objetivo expuesto en el párrafo anterior.

Dentro de los múltiples problemas de la IA, esta tesis pretende contribuir, por una parte, a mejorar alguna de las características de la búsqueda de casos existentes similares a uno dado; casos que estarán descritos mediante un conjunto de atributos numéricos. Y por otra parte, a caracterizar el nuevo caso con la etiqueta de la clase a la que más probablemente pertenezca, a la vista de los casos similares previamente recogidos.

Si la Inteligencia Artificial pretende emular el procesamiento de información que realizan los seres humanos, bueno será reflexionar sobre cómo estos aprenden a clasificar objetos.

Cuando un niño pequeño empieza a identificar distintos objetos, rápidamente aprende que el "atributo" número de patas del animal no sirve para distinguir entre un perro y un gato, pero el tipo de sonidos que emiten sí es un atributo muy valioso para obtener la clasificación correcta. Este aprendizaje es automático e inconsciente, la ponderación de las distintas características que su cerebro evalúa para avanzar en la comprensión del mundo evoluciona permanentemente. Los atributos que son importantes en unos casos, presentan menor relevancia en otros. En su mente se va construyendo progresivamente un mapa que sopesa la importancia de las diferencias entre atributos en distintos escenarios, las relaciones que se deben dar entre ellos, cuáles son imprescindibles y cuáles irrelevantes, etc.

No existe aún una modelización aproximada del modo de aprendizaje y razonamiento que llevan a cabo los seres humanos pero en mi opinión, el

funcionamiento descrito en el anterior párrafo, sería acertado representarlo mediante una medida de distancia entre los atributos de los objetos; esta podría basarse en una medida de distancia no euclídea (cuya matriz de la métrica dependiera de los objetos a clasificar). Así pues, para evaluar la disimilitud entre casos se utilizarían distancias que tratarán de ponderar en mayor medida aquellos atributos más influyentes, relevando a un segundo plano aquellos cuyo uso se revelase poco relevante, innecesario o incluso perjudicial. No soy el único que tiene esta convicción, en [2] Frank Jäkel exponía:

"There was no a priori reason to believe that mental representations should be Euclidean. There was ever little reason to believe that measurements of similarity or dissimilarity were linearly related to the distance or the inner product in a Euclidean space".

Así pues, el estudio que se ha desarrollado en esta tesis se podría encuadrar dentro del campo de la investigación básica sobre clasificación, medidas de distancia, la geometría del espacio y su optimización; todo ello orientado a mejorar la capacidad de los métodos de clasificación que emplean distancias. Su destino es que, en un futuro, sus resultados puedan ser incorporados como parte integrante de algoritmos que sirvan para resolver problemas de clasificación automática.

1.1 Hipótesis de partida

La hipótesis inicial de esta tesis fue la siguiente:

"Sería posible mejorar las prestaciones de los sistemas de clasificación si en vez de emplear una métrica euclídea se utilizase otra medida de distancia que se acomodara a las características del problema en cuestión. Esta nueva medida de distancia podría ser global o local, y debería poder ajustarse a cada problema bien mediante un cierto conocimiento previo y/o mediante técnicas de optimización".

1.2 Objetivos específicos

Para tratar de validar la hipótesis de partida se plantearon los siguientes objetivos específicos:

O1) Estudio de los métodos de clasificación que se han utilizado en las investigaciones y que representan el estado del arte. Ante la multitud de propuestas en este campo, muchas de ellas efímeras, se analizarán primordialmente aquellos algoritmos sobre los que existe

un cierto consenso sobre su calidad predictiva y que han sobrevivido al paso del tiempo.

O2) Estudio de los fundamentos matemáticos que permitan crear métricas avanzadas. En concreto técnicas de optimización, con y sin restricciones, y el funcionamiento de las métricas de Riemann.

O3) Estudio del estado del arte y desarrollo detallado de los métodos específicos de referencia sobre las métricas globales y locales utilizadas en la última década.

O4) Propuesta, estudio, justificación e implementación en forma de programa informático de una nueva métrica global que mejorase alguna de las características de los algoritmos que resuelven problemas de clasificación basándose en los vecinos próximos a uno dado.

O5) Propuesta, estudio, justificación e implementación en forma de programa informático de una nueva métrica local que mejorase la capacidad de predicción de los algoritmos que resuelven problemas de clasificación basándose también en los vecinos próximos a uno dado.

1.3 Metodología

Dentro de esta sección se va a exponer el método general para el análisis y generación de nuevas ideas en esta tesis, a continuación se enumerarán los problemas y los métodos que se emplearán para validar los resultados y por último se discutirá el método que resuelve la clasificación cuando se presentan problemas con múltiples clases.

1.3.1 Método de trabajo

El método de trabajo seguido ha consistido en:

- En primer lugar se llevó a cabo un estudio general de los algoritmos de clasificación, analizando sus ventajas e inconvenientes. En esta fase también se estudiaron las bases matemáticas que permitían entenderlos y los recursos de programación que podrían emplearse en una futura implementación.
- Posteriormente se analizaban en detalle los algoritmos relacionados con el objetivo específico que se quería cumplimentar, partiendo de varias fuentes:
    - Libros de referencia (autoridades en la materia). Tienen como ventaja que proporcionan información completa sobre algoritmos muy asentados, pero presentan el inconveniente de su falta de actualización.

    - Tutoriales de conferencias [3] y contenidos de los cursos que se imparten en las Universidades americanas más relevantes (Berkeley, George Mason…). Tiene la ventaja de que se actualizan frecuentemente, sus profesores son autoridades destacadas en el tema, y ayudan a fijar qué es importante y qué no.
    - Artículos publicados en revistas. Buscando de forma jerárquica es posible encontrar referencias de los avances más recientes en el área.
- Después del análisis anterior se enumeraban los campos abiertos, o en opinión de autor mal resueltos, que dejaban los algoritmos (como fue el caso con los algoritmos DANN o LaMaNNa). De ellos nacían propuestas de nuevos desarrollos para esta tesis.
- Implementación en distintos lenguajes de programación de los algoritmos de clasificación empleando la nueva métrica. En una primera fase en Matlab, pero posteriormente en lenguaje C++ (para aprovechar así su mayor velocidad de procesamiento).
- Elección de los problemas de test. Después de una fase de depuración y puesta en marcha de los algoritmos, estos eran probados mediante un conjunto de problemas seleccionados (sección 1.3.2).
- Análisis de los resultados obtenidos (de acuerdo a alguna de las técnicas de validación explicadas en la sección 1.3.3), cumplimiento de los objetivos y discusión de sus ventajas e inconvenientes.
- Publicación de los resultados.

1.3.2 Problemas usados en esta tesis

En todo estudio de clasificación es necesario emplear un conjunto de problemas que validen o refuten las hipótesis asociadas a la investigación.

En esta tesis se ha utilizado un conjunto de once problemas habitualmente utilizados en la literatura científica relacionada con la clasificación y se ha desarrollado uno propio (el denominado "Escuadra diagonal").

Los problemas sintéticos[1] que han sido utilizados son:

- "Waveforms".
- Escuadra diagonal.

Los problemas reales que han sido utilizados son [4]:

- Flores del Iris.
- "Australian Credit".

[1] La principal ventaja de los problemas sintéticos es que su número de casos se puede ampliar a voluntad y siempre es posible disponer de un conjunto extenso de casos de test.

- "Breast Cancer".
- "German Credit".
- "Glass".
- "Heart".
- "Image Segmentation".
- "Satellite Image".
- "Shuttle Control".
- "Vehicle Silhouettes".

En el anexo A1 se describen con detalle los distintos problemas que se han empleado en el estudio de las dos métricas nacidas de esta tesis.

Para analizar el grado de aciertos que consiguen los distintos algoritmos se han utilizado alguna de las técnicas que se explican en la sección siguiente.

1.3.3 Técnicas de validación

En esta investigación se han utilizado distintas técnicas para tratar de estimar la precisión de la clasificación:

- Empleo de un conjunto independiente de casos de test.
- Validación cruzada.
- "Leave-One-out" (LOO).
- "Bootstrap".

Cada uno de estos métodos tiene sus ventajas e inconvenientes y en los siguientes apartados se comentarán sus características y en qué estudios se han empleado.

1.3.3.1 El empleo de casos independientes de test

El disponer de un amplio conjunto de datos que representen de forma correcta a la población de casos a clasificar es quizás la situación ideal en cualquier investigación, pero no es la habitual porque:

- En primer lugar, disponer de multitud de casos suele ser infrecuente por el esfuerzo que esto supone (habitualmente técnico y económico).
- En otras ocasiones, el conjunto de casos es fijo y cerrado, y viene proporcionado por una tercera parte (p.ej. el repositorio de problemas de la UCI [4]).
- Por otra parte, si se disponen de múltiples casos empíricos, en vez de utilizarlos para conocer la bondad de la clasificación, podría juzgarse más juicioso emplearlos como casos de aprendizaje para así mejorar el modelo empírico.

Desde un punto de vista técnico, obviando los anteriores inconvenientes, y si los casos de aprendizaje representasen perfectamente al universo del problema, este sería el método óptimo para evaluar la bondad del modelo.

Su principal campo de utilización se encuentra en aquellos problemas sintéticos, donde es posible generar un extenso bloque de datos simplemente aplicando algún tipo de algoritmo.

Hay otros investigadores que lo emplean realizando una partición inicial en los datos disponibles y tomando un grupo (relativamente extenso) como datos de test. Esta decisión es más que discutible, ya que renuncian a un conjunto de casos que podría ser muy útil en la fase de ajuste del modelo.

En esta investigación se ha utilizado este método para la estimación de la precisión de la clasificación en la métrica LOM.

1.3.3.2 Validación cruzada

De entre todos los métodos empleados para estimar la precisión de una clasificación, quizás sea este el más utilizado en el mundo de la investigación.

Se basa en dividir el conjunto de casos de aprendizaje en dos subconjuntos.

El primero, denominado $\mathcal{A}$ (normalmente de mayor tamaño), que servirá para ajustar el modelo $f_{clasif}(\ \ )$ con el que se predice la clasificación:

$$f_{clasif}(\boldsymbol{\mathcal{P}}, \boldsymbol{X}_{\mathcal{A}}), \qquad \boldsymbol{X}_{\mathcal{A}} = \{\boldsymbol{x}_i\}, \ \boldsymbol{x}_i \in \mathcal{A} \tag{1.1}$$

donde $\boldsymbol{\mathcal{P}}$ son los parámetros actuales del modelo, $\boldsymbol{X}_{\mathcal{A}}$ los vectores con los atributos de todos los casos que pertenecen a $\mathcal{A}$.

Y en otro subconjunto $\mathcal{T}$, independiente del anterior, que toma temporalmente el rol de casos de test.

$$\hat{y}_n = f_{clasif}(\boldsymbol{\mathcal{P}}, \boldsymbol{X}_{\mathcal{A}}, \boldsymbol{x}_n), \qquad \boldsymbol{x}_n \in \mathcal{T} \tag{1.2}$$

El valor medio de los aciertos cometidos al clasificar los casos del subconjunto $\mathcal{T}$ será:

$$E\left(c_{\hat{y}_n}(\boldsymbol{x}_n)\right) = \frac{1}{|\mathcal{T}|}\sum_{\boldsymbol{x}_n \in \mathcal{T}} c_{\hat{y}_n}(\boldsymbol{x}_n) \quad , \ \boldsymbol{x}_n \in \mathcal{T} \tag{1.3}$$

donde $c_{\hat{y}_n}(\boldsymbol{x}_n)$ es una función que se evalúa como 1 cuando el caso $\boldsymbol{x}_n$ pertenece realmente a la clase predicha $\hat{y}_n$, siendo 0 en caso contrario[2].

Este procedimiento se repite, volviendo a dividir el conjunto total de casos de aprendizaje en dos nuevos subconjuntos $\mathcal{A}$ y $\mathcal{T}$ (distintos de los anteriores). Se vuelve a calcular la tasa de aciertos con esta nueva distribución, y después de varias iteraciones siguiendo esta metodología, se promedia el valor de las distintas tasas de aciertos.

En la práctica, lo que se hace es dividir el conjunto de casos de aprendizaje en K subconjuntos de aproximadamente el mismo tamaño. En la primera iteración se usa como subconjunto de test el primero de estos grupos y el resto se usan para entrenar el modelo empírico, con el modelo estimado y el subconjunto de casos de test se calcula la tasa de aciertos obtenida. En la

[2] En otros casos, y dependiendo del tipo de algoritmo, lo que se mide es el error medio cuadrático (MSE) entre el valor real de la clase y el predicho.

segunda iteración se emplea el segundo de los *K* subconjuntos como casos de test y el resto se usan para ajustar un nuevo modelo...

A esta técnica se le denomina validación cruzada mediante *K* partes o divisiones ("*K*-Fold Cross Validation").

Mientras que en muchos problemas la validación cruzada sirve para evaluar la precisión de clasificación de un determinado algoritmo, en otros muchos escenarios el objetivo de esta técnica es estimar, sin "sobreaprendizaje"[3], el valor de los parámetros $\boldsymbol{\mathcal{P}}$ que optimizan la precisión de clasificación de un determinado modelo de clasificación que está siendo ajustado.

En esta tesis se emplea una validación cruzada de 10 divisiones en el entrenamiento de las máquinas de vectores de soporte.

1.3.3.3 "Leave-One-Out" (LOO)

Podría considerarse como una versión extrema de la validación cruzada de *K* divisiones cuando se dispone de *N* casos experimentales y se decide dividir el conjunto de casos en $K = N$ partes. Así pues, se emplearán todos los casos disponibles menos uno (*N*-1) para ajustar el modelo empírico, y el caso que resta (supóngase que es el *n*-ésimo) se utilizará para evaluar la precisión de la clasificación.

Obviamente, evaluar la precisión mediante un solo caso proporcionará resultados extremos (0-100% de aciertos), pero al repetir el procedimiento para todos y cada uno de los casos disponibles, y promediar los resultados, se obtiene una estimación correcta (tan próxima al error real como es posible con estos datos empíricos).

$$\hat{y}_n = f_{clasif}\left(\boldsymbol{\mathcal{P}}\,,\, \boldsymbol{X}_{N-n}\,,\, \boldsymbol{x}_n\right)$$
$$tasa\ aciertos_{LOO} = \frac{1}{N}\sum_{n=1}^{N} c_{\hat{y}_n}(\,\boldsymbol{x}_n) \tag{1.4}$$

donde $f_{clasif}\left(\boldsymbol{\mathcal{P}},\, \boldsymbol{X}_{N-n}\,,\, \boldsymbol{x}_n\right)$ proporciona la clase estimada para el caso $\boldsymbol{x}_n$ (de acuerdo al algoritmo de clasificación) en el que $\boldsymbol{X}_{N-n}$ representa el conjunto de los casos de aprendizaje excluyendo el caso *n*-ésimo;.

Luntz y Brailovsky [5] probaron que la tasa de aciertos LOO es un estimador no polarizado.

Uno de sus principales inconvenientes es que se deben estimar *N* modelos de clasificación, con el consiguiente gasto de recursos computacionales.

Se usa muy habitualmente en las técnicas de clasificación basadas en *k*-NN y en esta tesis lo hemos utilizado para estimar la precisión de clasificación de este algoritmo.

---

[3] "Overfitting" en inglés.

1.3.3.4 "Bootstrap"

El método de "bootstrap"[4] propuesto por Efron y Tibshirani [6] consiste en crear una gran cantidad (sea esta el número *B*) de conjuntos "artificiales" de casos para poder entrenar múltiples instancias de algoritmos de clasificación.

Para formar cada uno de estos conjuntos se tomarán aleatoriamente, y con reemplazamiento, *M* casos del conjunto global de casos de aprendizaje[5]. A continuación, con este conjunto, se ajustarán los parámetros de un clasificador y se estimará su precisión de clasificación con el conjunto global de datos de aprendizaje.

Este procedimiento se repite con los *B* conjuntos de "bootstrap" y al final se promedian los resultados obtenidos.

La técnica de "bootstrap" se debe modificar cuando se emplea para estimar la precisión de un algoritmo de tipo *k*-NN (y con mayor razón si fuese 1-NN). La presencia de los casos comunes entre los conjuntos de test y los del conjunto "bootstrap" sobreestimaría fuertemente la estimación de la precisión de la clasificación. Para evitarlo se deben eliminar de los casos de test aquellos que ya se encuentran en el de "bootstrap".

En esta tesis se ha utilizado esta técnica en el cálculo de la precisión de clasificación del algoritmo BTW.

1.3.4 Método empleado para resolver la clasificación en los problemas con múltiples clases

Muchos de los problemas reseñados en la sección 1.3.2 presentan múltiples clases y las métricas diseñadas en esta tesis tratan de orientar las medidas de distancia en las direcciones que mejor permiten separar cada pareja de clases. Es por lo tanto necesario adoptar un algoritmo que permita trabajar con las distintas combinaciones de clases, obtener las métricas correspondientes, y la mejor clasificación de los distintos casos en estos escenarios; para, por último, combinar los resultados de todas estas predicciones binarias en una única decisión de clasificación final [7].

En su libro [8], Christopher Bishop analiza las dos soluciones que representan el estado del arte actual[6] para abordar este problema:

- "Una vs. una": hace "competir" a las clases una frente a otra obteniendo así la clase ganadora de cada confrontación. Posteriormente recuenta las veces que se ha impuesto cada una de las clases, y la que presente mejor coeficiente será la clase a asignar al caso nuevo. El mayor problema es que hay que ejecutar el algoritmo

[4] En algunos libros el término "bootstrap" se traduce como "muestreo autodocimante", aunque esta acepción no está muy extendida.

[5] Habitualmente *M* coincide con la cardinalidad del conjunto de casos, es decir *N.*

[6] Y sus posibles variantes.

de clasificación $J.(J-1)/2$ veces (donde J es el número de clases), aunque también es cierto que suelen ser problemas más pequeños (en número de casos). Este método se vuelve intratable cuando el número de clases se hace muy elevado.

- "Una vs. resto": aquí se hace competir a cada clase frente a la unión de todas las demás. La clase que mejor resultado obtenga será la elegida. En este supuesto solo es necesario ejecutar el algoritmo de clasificación J veces.

Muchos autores [9], [10], [11] se han postulado a favor de la opción "una vs. una", mientras que otros como Vapnik [12] apuestan por "una vs. resto".

Una profunda investigación realizada por Rifkin et al. [13] deja establecido que ambas estrategias dan resultados que no son claramente superiores unos a otros.

En esta tesis se ha preferido emplear el método "una vs. una", tanto para la métrica LOM como para la BTW, ya que es más fácil la interpretación geométrica de una métrica adaptada a separar los casos de dos clases y conduce a superficies de separación más simples.

1.4 Estructura de la tesis

Este primer capítulo está dedicado a fijar la información básica de los contenidos de esta tesis, qué hipótesis se pretende probar, cuáles son los objetivos; y termina con un amplio apartado dedicado a todos los aspectos metodológicos (de naturaleza general, problemas de test, técnicas de validación…).

Los capítulos 2 y 3 cubren la necesidad de introducir los conceptos básicos que permitan entender las posteriores explicaciones, tanto del estado del arte como del desarrollo de las métricas. En concreto, el capítulo 2 presenta las bases informáticas de la teoría de clasificación supervisada y el 3 se divide en dos secciones de carácter matemático: una dedicada a las técnicas de optimización relacionadas con esta tesis y otra a dar una breve (y simple) explicación de lo que son las métricas riemannianas que se utilizarán en el posterior desarrollo de la métrica LOM. Si el lector posee conocimientos sobre alguna de estas técnicas puede omitir la lectura de las correspondientes secciones.

El estado del arte se expone en el capítulo 4. Se divide en cinco secciones diferenciadas, en todas ellas se procura solo comentar aquellas contribuciones que tienen relación directa con la tesis.

La primera sección tiene que ver con las métricas. Aparte de una explicación general de las consideradas básicas, como pueden ser las de Minkowsky y

Mahalanobis, se ha incluido una recopilación personal y breve de las métricas más especializadas que aparecen frecuentemente en la literatura científica.

En la segunda sección se exponen las características de las tres técnicas de clasificación en las que se apoya la tesis: la clasificación basada en los vecinos próximos, el análisis del discriminante y las máquinas de vectores de soporte. Se ha procurado describir todas ellas con un esquema similar.

En la sección tercera se compilan los métodos que se han utilizado en la literatura científica para el aprendizaje de las métricas, en particular técnicas que tienen que ver con la optimización de una matriz de tipo Mahalanobis.

La cuarta sección tiene que ver con aquellos artículos publicados en revistas que han influido directamente en los desarrollos de las métricas BTW y LOM. Para todos los artículos se ha intentado fijar el objetivo que se persigue, cómo los autores plantean su algoritmo, cuál es la complejidad computacional y qué ventajas e inconvenientes presenta en relación con las aportaciones novedosas de esta tesis.

Este capítulo 4 termina con una aportación propia al estado del arte sobre la distorsión de la geometría, su formulación en forma del factor de magnificación y su repercusión en la mejora de la precisión de la clasificación.

El desarrollo de las métricas BTW y LOM se estructura en los capítulos 5 y 6, ambos tienen una distribución similar; en primer lugar se expone el objetivo básico, el enfoque general y la relación con los trabajos del estado del arte; para a continuación explicar los detalles del diseño de cada métrica y su implementación. Cada uno de estos capítulos termina con la presentación de los resultados experimentales obtenidos y una discusión final sobre sus ventajas e inconvenientes.

El último capítulo técnico es una recopilación breve sobre la validación de la hipótesis de partida y los objetivos de esta tesis, conclusiones y futuras líneas de trabajo para cada una de las dos métricas.

En los capítulos 8 y 9 se adjuntan sendas tablas con los acrónimos y la notación matemática que se va usar en esta memoria, para terminar en el capítulo décimo donde se recogen las referencias bibliográficas, indicadas de acuerdo al estilo recomendado por la asociación IEEE.

2 Elementos de la teoría de clasificación

Este capítulo se dedica a presentar la problemática de la clasificación dentro del mundo de la Inteligencia Artificial.

Se comenzará en la sección 2.1 proporcionando una definición de esta actividad, relacionándola con la actividad de los seres humanos y la informática. En la sección 2.2 se expone una taxonomía de los métodos de clasificación y la descripción de qué es el aprendizaje supervisado, no supervisado y semisupervisado.
Mientras que la subsección 2.2.3 de este capítulo ofrece una visión rápida y superficial de los algoritmos de clasificación supervisados más importantes, en el capítulo 4 se profundizará en los tres algoritmos de clasificación que han sido básicos en esta tesis.

En la tercera sección de este capítulo se aborda la formulación clásica (estadística) del problema de clasificación supervisado; para incluir en la 2.4 la formulación de la función de pérdida y en la sección 2.5 se ofrece la solución (más bien teórica) que proporciona el teorema de Bayes.

Por último, este capítulo termina con la sección 2.6, donde se exponen las características generales de los modelos de clasificadores. En ella se comentarán aspectos como la robustez, escalabilidad, los problemas del sobreaprendizaje..., terminando con una pequeña exposición del principio de la minimización del riesgo estructural debido a Vapnik.

2.1 La clasificación y su entorno

Una de las tareas habituales de las aplicaciones informáticas tradicionales es la manipulación de información en soporte digital. Prácticamente todas las tareas actuales en las que se utilizan bases de datos presentan una fase de búsqueda de información, para que luego esta sea modificada, eliminada, etc. Las aplicaciones de visión artificial tradicionales comparan los píxeles que se encuentran en determinadas áreas de una imagen frente a un patrón, si la coincidencia supera un valor umbral se da como válida la similitud entre ellas y sus consecuencias.

Los ordenadores digitales son capaces de realizar estas búsquedas de coincidencias exactas de forma rápida y eficiente; el algoritmo subyacente se basa en comparar secuencialmente los valores que se le presentan para evaluar si son iguales o no. Cada una de estas comparaciones las puede realizar un ordenador mediante unas pocas instrucciones de su CPU (cuyo tiempo de ejecución se reduce a escasos nanosegundos).

Pero ya desde los primeros tiempos de la informática se planteó el problema de resolver problemas no tan "determinísticos", sino tratar de emular al ser humano en su actividad cotidiana de procesar información.

Clasificar es la acción de asignar a un objeto la pertenencia a una determinada clase[1]. Tradicionalmente los seres humanos han clasificados los elementos de su entorno sin seguir un procedimiento muy específico. Durante muchos siglos la clasificación y sus aspectos teóricos han sido campo de estudio para los filósofos; las categorías de Aristóteles y Kant podrían ser los ejemplos más relevantes.

En el mundo de la informática, clasificar es conseguir el objetivo antes citado mediante un procedimiento algorítmico que intente obtener la mayor precisión posible en la tipificación de aquellos objetos cuya pertenencia a un grupo no es conocida.

Mediante la programación tradicional es posible abordar los problemas más simples, un conjunto de reglas expresadas mediante sentencias `if… then…` pueden automatizar clasificaciones muy sencillas. Actualmente, cualquier problema de interés presenta tal dificultad que solo se considera el diseño de algoritmos de clasificación mediante técnicas de aprendizaje [14].[2]

Por aprendizaje se puede entender cualquier tipo de estrategia que permita evolucionar el modelo de clasificación de forma que se incremente la calidad de las predicciones.

---

[1] No se considera en esta tesis la otra definición típica de clasificación que está relacionada con ordenar, de acuerdo a un criterio, un conjunto de objetos.

[2] De acuerdo a estos mismos autores, cualquier método que incorpore información de los casos empíricos en el diseño del algoritmo clasificador se puede decir que aplica algún tipo de "aprendizaje".

Así pues, en los métodos de clasificación mediante aprendizaje, el objetivo es construir un algoritmo que pueda ser ajustado para devolver, con la máxima certeza posible, la clase concreta o grupo a la que pertenece un cierto caso partiendo de la información de otros casos conocidos.

El tipo de aprendizaje empleado permite agrupar los métodos de clasificación en tres grandes bloques:

- Métodos de aprendizaje supervisados.
- Métodos de aprendizaje no supervisados.
- Métodos de aprendizaje semisupervisados.

No se pretende en esta tesis realizar un estudio detallado de todos los algoritmos que pueden ser empleados en la clasificación de casos. En un estudio en el año 2001 [15] se cita que pueden ser por encima de 200 o 300 los métodos que han sido descritos en los últimos 50 años. Una de las mejores descripciones detalladas de unas decenas de estos algoritmos puede encontrase en [16][3].

A modo de introducción, en la siguiente sección se proporciona una taxonomía de los métodos de clasificación, realizando especial énfasis en los métodos de aprendizaje supervisados (que serán sobre los que se basará esta tesis).

[3] Existe una versión actualizada a 2013.

2.2 Taxonomía para los métodos de clasificación

En la siguiente Figura se puede apreciar una posible taxonomía para los métodos de clasificación, ha sido creada combinando información de distintas fuentes [14], [8], [16].

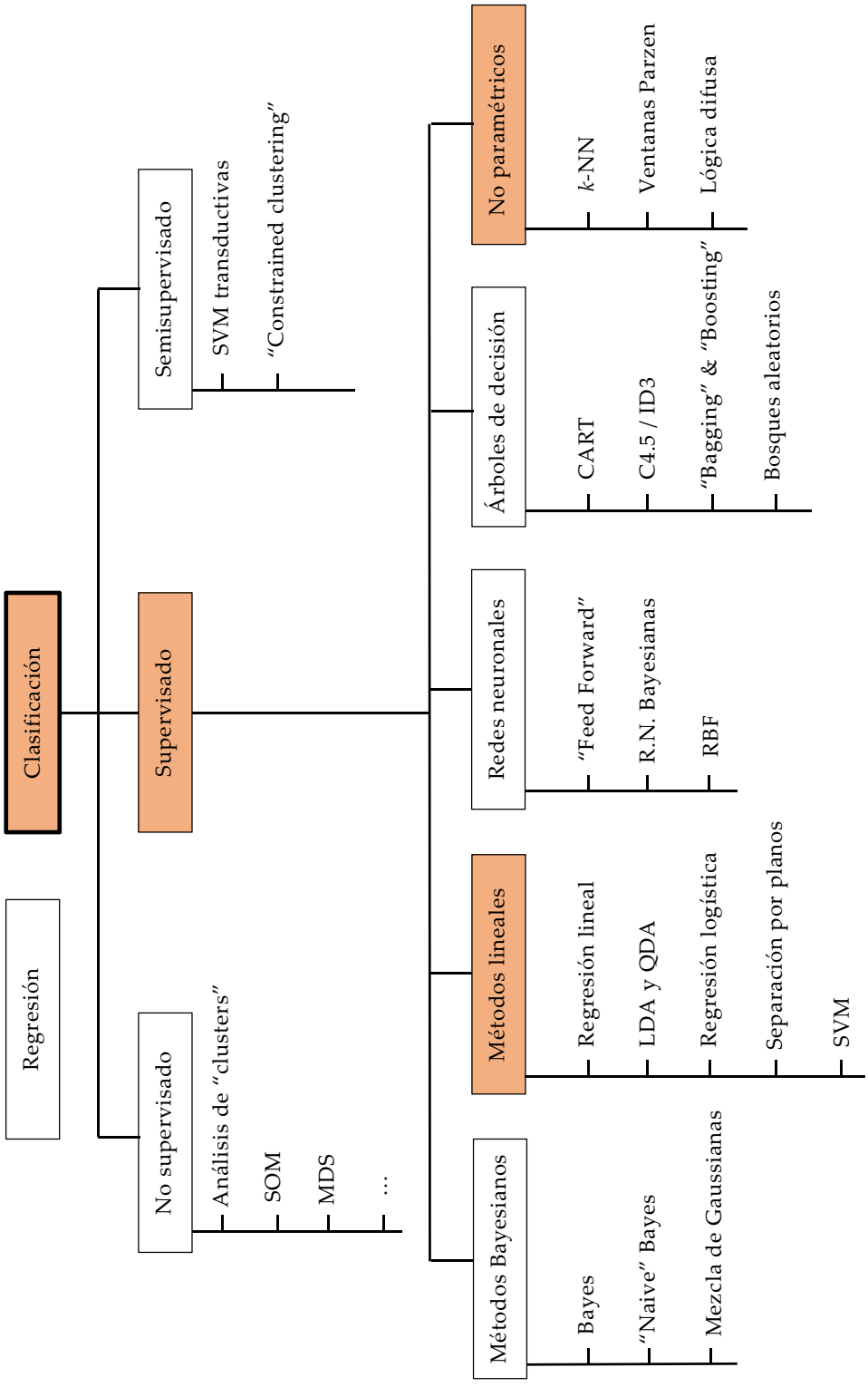


Figura 2.1.- Taxonomía para los métodos de clasificación

2.2.1 Métodos de aprendizaje no supervisados

En los problemas de aprendizaje no supervisado el objetivo, más que proporcionar un resultado concreto a la hora de clasificar un caso, es describir las relaciones, densidades de probabilidad, asociaciones y patrones comunes entre los casos del problema.

En este tipo de problemas los casos se describen mediante un conjunto de atributos que se recogerán en un vector $\boldsymbol{x}$ de dimensión A; aquí no tiene sentido hablar de la clase a la que pertenecen los casos de aprendizaje.

$$\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N\}, \qquad \boldsymbol{x}_i \in \mathbb{R}^A \tag{2.1}$$

Así pues, en este escenario la mira está más puesta en discernir las agrupaciones naturales de casos. Por ejemplo, para los problemas de dimensionalidad pequeña existen métodos muy eficientes que permiten estimar la densidad de probabilidad para la distribución de los casos y, en función de sus modas, establecer distintas regiones diferenciadas (que se podrían suponer asociadas a distintas clases de casos, aunque esto no se sepa con certeza).

Quizás el gran inconveniente, en la mayor parte de los problemas, es que no existe una medida que permita "verificar" la calidad de un aprendizaje no supervisado (ya que no existe con qué comparar)[4].

Dentro de los métodos de aprendizaje no supervisado se pueden destacar:

- El análisis de "clusters". Su objetivo es agrupar conjuntos de casos en bloques homogéneos, de forma que los casos de un "cluster" tengan en común más que los que pertenecen a distintos "clusters".

  Los distintos algoritmos de este tipo se pueden agrupar en:

  - Algoritmos combinacionales ("Combinatorial algorithms"). Proceden a realizar su trabajo sin necesidad de un modelo probabilístico subyacente.
  - Modelización mediante mezclas de funciones de densidad de probabilidad[5] ("Mixture modelling"). Presuponen que los casos proceden de distintas distribuciones de probabilidad. Se asume que los datos que responden a una misma pdf son parte de un cierto "cluster".
  - Búsqueda de modas ("Mode seeking"). Desde una perspectiva no paramétrica, buscan las modas de funciones de densidad de

[4] E incluso hay controversias sobre si es más propio el objetivo de encontrar una relación entre atributos para formar una agrupación de casos, o describir estos mediante una combinación de distribuciones de tipo gaussiano.

[5] El término "función de densidad de probabilidad" se suele abreviar mediante su acrónimo en inglés: "pdf".

probabilidad. Los casos más próximos a cada una de las modas forman un "cluster".

Las medidas de similitud (o disimilitud) son básicas en este tipo de algoritmos.

El más popular de ellos es el conocido como "K-means" y pertenece al grupo de algoritmos combinacionales. Su objetivo es optimizar una función de error basada en la distancia entre cada caso y el prototipo[6] del "cluster" al que dicho caso está siendo asignado.

- Mapas autoorganizativos ("Self-Organizing Maps"). Se puede considerar una versión del algoritmo "K-means" en el que los prototipos deben residir en una variedad unidimensional o bidimensional.
- Escalado multidimensional ("Multidimensional scaling"). Plantea la transformación de los atributos del conjunto de casos a un espacio de dimensión menor, de tal forma que las distancias entre casos en el espacio original se preserven, tanto como sea posible, en el nuevo espacio reducido.

2.2.2 Métodos de aprendizaje semisupervisados

En estos métodos se emplean tanto casos cuya clase es conocida como no [17]. Su interés nace de aquellos problemas donde es fácil disponer de una gran cantidad de casos cuya clase es desconocida y solo unos pocos donde esta se conoce, por ejemplo: imágenes de cámaras de seguridad, documentos descargados de Internet...

El objetivo de estos métodos es intentar mejorar la calidad de predicción que se obtendría con un algoritmo de aprendizaje supervisado que utilizase solamente los casos que tienen una clase asignada. Para tener una cierta esperanza de éxito es necesario que el problema cumpla con una condición:

"si dos casos, en una zona de alta densidad de casos, están representados por conjuntos de atributos próximos en el espacio, también sus clases deberían estar próximas".

Así pues, los métodos de aprendizaje semisupervisado se pueden aplicar en escenarios donde se usa tradicionalmente:

- El aprendizaje supervisado. Y aquí se puede distinguir entre:
  - Aprendizaje semisupervisado inductivo: el campo de predicción del algoritmo entrenado es todo el espacio de atributos, es decir trata de generalizar más allá de los casos de aprendizaje que se han utilizado.

[6] Por prototipo se entiende un elemento del conjunto de casos de aprendizaje que es muy representativo en esa vecindad.

    - Aprendizaje transductivo: el campo de predicción se limita a los casos de aprendizaje que están sin etiquetar con una clase; y no se pretende generalizar a todo el espacio de atributos.
- El aprendizaje no supervisado. Modifican alguna técnica de "clustering" para incluir la información de si dos casos deben estar o no en un determinado agrupamiento.

En algunos algoritmos, en vez de conocer la clase concreta de cada caso, se dispone de un conjunto de relaciones que indican si cada pareja de casos pertenece o no a la misma clase (si son similares o no) [18] [19]. Un tipo de algoritmo para este tipo de aprendizaje semisupervisado [18] emplea una distancia cuadrática para medir la disimilitud entre casos. El problema de aprendizaje (en este caso el aprendizaje se lleva a cabo mediante una optimización) quedaría expresado como:

$$\min_{\boldsymbol{G}} \sum_{\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{S}} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_{\boldsymbol{G}}^2$$

$$\text{sujeto a: } \begin{cases} \boldsymbol{G} \succcurlyeq 0\,, \\ \sum_{\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{D}} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_{\boldsymbol{G}}^2 \geq 1 \end{cases} \tag{2.2}$$

siendo $\mathcal{S}$ el conjunto de elementos que poseen relaciones de semejanza entre ellos, $\mathcal{D}$ el conjunto de elementos que poseen relaciones de no semejanza y $\boldsymbol{G}$ la matriz de la métrica que es objeto de optimización.

Su resolución pasa por un método de optimización mediante programación semidefinida (este tipo de planteamientos de optimización se recogen en la sección 4.3 y las principales características de la programación semidefinida se discuten en la sección 3.1.4).

2.2.3 Métodos de clasificación supervisados

En los algoritmos de aprendizaje supervisado, los casos se componen tanto de sus atributos[7], que son las características numéricas, categóricas,… que los definen, como de la identificación de las clases[8] a las que pertenecen.

Para el conjunto de estos casos son deseables dos características:

- La primera es que cada elemento del conjunto de casos esté perfectamente medido y clasificado (su clase ha debido ser

[7] También denominados en la literatura científica como: "entradas", "características", "predictores", "variables independientes"…

[8] En algunos trabajos denominada también como: "salida", "respuesta", "variable dependiente"...

establecida previamente por un método de fiabilidad contrastada), aunque en la práctica se suelen presentar errores.

- También se asume que la distribución de los casos es representativa de lo que sería el conjunto de la población total (lo cual tampoco está garantizado en los problemas reales).

Como ya se ha comentado para los métodos no supervisados, en esta tesis los atributos de un caso se recogerán en un vector $\boldsymbol{x}$ de dimensión A, mientras que la clase será un escalar y se la denominará con la letra y (se utiliza la letra J para referirse al número de clases del problema).
Un conjunto de N casos empíricos se expresa mediante:

$$\begin{aligned} &\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N\}, \qquad \boldsymbol{x}_i \in \mathbb{R}^A \\ &\{y_1, y_2, \dots, y_N\}, \qquad y_i \in \{1, 2, \dots, J\} \end{aligned} \tag{2.3}$$

En los algoritmos que utilicen el aprendizaje supervisado existirá siempre un conjunto de casos empíricos importante: los denominados "casos de aprendizaje". Este conjunto de información será utilizada por el algoritmo de aprendizaje para ajustar los parámetros del modelo de clasificación. Así pues, en este escenario de aprendizaje, el algoritmo utilizará tanto los atributos de los casos empíricos disponibles como sus clases, que también son conocidas.
En muchos casos existe otro conjunto de casos empíricos conocido como "casos de test" que son aquellos que se emplearán para verificar la precisión de la clasificación, y que no se utilizan durante la fase de aprendizaje. Cuando no exista este conjunto se tendrán que emplear alguno de los métodos indicados en la sección 1.3.3 para verificar la bondad del aprendizaje.

Existen otras variantes del aprendizaje supervisado, como es el aprendizaje por refuerzo[9]; este parte de que el algoritmo que está siendo entrenado decide a qué clase pertenece un caso de aprendizaje y, posteriormente, un "maestro" externo le indica si esa decisión ha sido correcta o no (pero no proporciona el valor de la clase correcta para ese caso), es decir, puede saber si su decisión actual ha sido errónea, pero no cuál sería la clase correcta.

En las siguientes páginas se detalla un resumen con las principales características, ventajas e inconvenientes de los métodos de clasificación supervisados más importantes agrupados en clases.

[9] También denominado "método de aprendizaje mediante crítica".

Clase	Enfoque	Ventajas	Inconvenientes	Métodos más representativos	Comentarios
Métodos Bayesianos	Se basan en la aplicación práctica del teorema de Bayes. Estos métodos emplean los casos de aprendizaje como la información que permitirá delinear las probabilidades "a posteriori" (para que se asemejen más a la distribución real que las probabilidades "a priori").	• El método de Bayes sirve como referencia cuando se conocen con certeza las funciones de densidad de probabilidad (o se pueden estimar por la existencia de abundantes datos). • La literatura científica relata buenos resultados del método "Naive" Bayes cuando la dimensionalidad del espacio de atributos es alta.	• En los problemas reales es muy poco habitual conocer las funciones de densidad de probabilidad. • Es fácil que se presente el problema COD.	• Bayes. • "Naive" Bayes. • Modelos aditivos (suma de gaussianas). • Redes Bayesianas, "Bayesian belief networks".	Estimar las funciones de densidad de probabilidad como suma de gaussianas ha sido un método muy explorado, pero suele conducir a problemas de sobreaprendizaje. Los métodos de "máxima verosimilitud", también empleados en el aprendizaje supervisado, representan la vieja disputa entre la escuela probabilística y la Bayesiana.

Clase	Enfoque	Ventajas	Inconvenientes	Métodos más representativos	Comentarios
Métodos lineales	Este tipo de métodos permite calcular fronteras de separación entre clases de tipo lineal.	• Son simples de utilizar. • Son estables. • Buena capacidad de generalización. • En muchos problemas prácticos ofrecen muy buenos resultados.	• Funciones de separación de clases poco adaptables (en problemas complejos). • Es muy difícil que la solución real del problema sea realmente lineal. • Algunos requieren que los datos cumplan condiciones poco realistas (p.ej. distribuciones gaussianas para los valores de los atributos).	• Regresión lineal. • Regresión logística. • Análisis de discriminante lineal. • Separación por planos (perceptrón). • SVM.	En muchos casos es posible adaptarlo a problemas no linealmente separables mediante transformaciones de las variables de entrada, p.ej. expansiones lineales de funciones, polinomios, “splines”, “wavelets”, “kernels”…

Clase	Enfoque	Ventajas	Inconvenientes	Métodos más representativos	Comentarios
Árboles de decisión	Su objetivo es particionar el espacio de atributos en conjuntos de rectángulos multidimensionales, asignando a cada uno de ellos una clase.	• La interpretabilidad de los resultados de la clasificación es excelente. • Representa el modo de razonar de muchos expertos, procediendo primero a considerar las decisiones más relevantes, y terminado con los matices. • Poseen una representación gráfica muy explicativa.	• Las particiones se suelen realizar de forma paralela a los ejes coordenados de los atributos. • Tienen tendencia al sobreaprendizaje y hay que "podarlos". • Presentan una gran varianza: pequeños cambios en los casos de aprendizaje conducen a importantes modificaciones en la estructura del árbol. • Conduce a superficies de separación de clases poco "suaves".	• CART. • ID3 / C4.5 / C5.0. • "Bagging & Boosting", bosques aleatorios.	Es necesario adoptar un criterio para terminar las sucesivas divisiones, existen múltiples posibilidades para este aspecto: errores de clasificación, índice de Gini, entropía cruzada. Las técnicas de "Bagging - Boosting" promedian los resultados de muchos árboles para tratar de reducir la varianza y mejorar la predicción. En la última década se han utilizado también para mejorar las predicciones resultantes de redes neuronales.

Clase	Enfoque	Ventajas	Inconvenientes	Métodos más representativos	Comentarios
Redes neuronales	Se caracterizan por emplear los propios casos tanto para crear el modelo como para ajustar los pesos que conectan las unidades de procesamiento básicas. En un principio trataron de representar los mecanismos de funcionamiento de las neuronas biológicas.	• No es necesario determinar un modelo "a priori". • Se adaptan perfectamente a realizar predicciones en problemas fuertemente no lineales.	• No existe una teoría para fijar las características de los niveles ocultos. • Tiene mucha tendencia al sobreaprendizaje. • No ofrecen explicaciones sobre el resultado de la clasificación ni sobre su "lógica" de razonamiento. • La función error es no convexa y su minimización (empleada para optimizar los pesos entre neuronas) puede quedar atrapada en mínimos locales.	• Redes neuronales multicapa de tipo "Feedforward" (entrenadas mediante "Backpropagation"). • Redes neuronales Bayesianas. • Redes neuronales basadas en RBF. • Redes estocásticas (Máquinas de Boltzmann, "simulated annealing").	Algunos autores han estudiado métodos de regularización (como el decaimiento de los pesos) para tratar de combatir el sobreaprendizaje. Las redes neuronales de tipo RBF (muy utilizadas a principios del siglo XXI) hoy en día se pueden reemplazar ventajosamente por las SVM-RBF.

Clase	Enfoque	Ventajas	Inconvenientes	Métodos más representativos	Comentarios
Métodos no paramétricos	Emplean estrategias muy variadas. En su mayoría pretenden clasificar un caso en función de la clase de su(s) prototipo(s) más cercano(s).	• Son simples de utilizar. • En muchos problemas prácticos ofrecen muy buenos resultados. • Se adaptan perfectamente a realizar predicciones en problemas fuertemente no lineales. • Pueden basarse en información procesada por el ser humano (y no en los casos de aprendizaje, sobre todo si estos son escasos).	• Si los prototipos son muy abundantes el tiempo de cálculo se puede alargar significativamente. • En k-NN, la métrica empleada influye mucho en la calidad de la clasificación. • El número de atributos afecta muy negativamente a las posibilidades de inferencia basadas en lógica difusa.	• k-NN. • Ventanas de Parzen. • Clasificación basada en lógica difusa.	Son especialmente efectivos si los prototipos se pudieran editar eficientemente. Hay variaciones que explotan las invariancias particulares de un problema en las medidas de distancias, de forma que se adaptan perfectamente a él, p.ej. la distancia tangente para la clasificación de dígitos escritos a mano.

Tabla 2.1.- Características de los métodos de clasificación supervisados.

2.3 Formulación estadística del problema de clasificación supervisado

A primera vista, la tarea de clasificar un caso parece simple, pero se complica cuando se empieza a considerar cómo cuantificar el término "clase correcta". Bajo un enfoque estadístico, el sentido común nos indica que la clase correcta debería ser aquella que mayor probabilidad tenga de ser la verdadera.

Si solo se conociesen las proporciones de casos que pertenecen a cada clase, las probabilidades "a priori": $p(y_j)$ guiarían esta decisión[10]. Por ejemplo, si el 90% de los casos perteneciesen a la clase 1 y el 10% a la clase 2, asignando cualquier caso nuevo a la clase 1 se cometería de promedio un 10% de errores.

Pero los algoritmos de clasificación basados en el aprendizaje supervisado disponen de un conjunto de N casos experimentales, formados por los valores de los atributos y sus clases $\{\boldsymbol{x}, y\}_n,\ n = 1..N$, para mejorar la calidad de sus predicciones.

Así pues sería más apropiado utilizar la probabilidad " a posteriori", es decir, la probabilidad de que la clase real sea y_j una vez conocido el conjunto de valores que toman los atributos $\boldsymbol{x}$ de un caso, esto es: $p(y_j|\boldsymbol{x})$.

De esta reflexión es fácil concluir que aquella clase que haga máxima esta última probabilidad será la elegida preferentemente para clasificar el caso:

$$y = arg \max_{1 \le j \le J} p(y_j|\boldsymbol{x}) \tag{2.4}$$

Como en los problemas reales habitualmente no se disponen de las funciones de densidad de probabilidad $p(y_k|\boldsymbol{x})$, estas deben ser estimadas de forma indirecta. Existen diversos métodos para aproximar dicha probabilidad, muchos de los métodos reseñados en la Tabla 2.1 buscan directa o indirectamente este objetivo.

2.4 Reformulación del problema teniendo en consideración la función de pérdida

Por otra parte, la clasificación es a menudo un problema práctico e inmerso en el mundo real, y dar como solución la clase más probable estadísticamente para un cierto caso, puede que no sea tampoco lo más adecuado.

Supóngase un hipotético ejemplo de tipo médico en el que existen dos posibles pronósticos, y en el que el riesgo que sufre el paciente si se sigue el

[10] Este valor podría considerarse como un límite inferior para la precisión de la clasificación.

tratamiento correspondiente al primero de ellos es muy superior al del segundo. Se ve claramente que en la toma de esta decisión no solo tienen que participar las probabilidades, sino también una función que penalice los resultados de una clasificación incorrecta; para todo ello se puede utilizar la función de pérdida[11].

Formalizando este concepto, se expresa la función de pérdida mediante $perd_{jk}(\boldsymbol{x})$, que se interpreta como la pérdida causada al asignar el caso $\boldsymbol{x}$ a la clase j cuando la verdadera es la clase k. En general, y para las posibles combinaciones de distintas clases, los valores retornados por el conjunto de funciones de pérdida formarán una matriz de J x J celdas.

Asignar valores a estas funciones depende, normalmente, de múltiples aspectos relacionados específicamente con el problema en cuestión; aunque en la mayoría de las situaciones reales y, por razones obvias, $perd_{jj}(\boldsymbol{x}) = 0$.

Bajo este nuevo enfoque, y para una instancia en concreto, la clase a elegir será aquella que minimice la función de pérdida:

$$\begin{aligned} y_k &= arg \min_{1 \le k \le J} PERD_k(\boldsymbol{x}) = \\ &= arg \min_{1 \le k \le J} \sum_{j=1}^{J} perd_{jk}(\boldsymbol{x}) \,.\, p(y_j|\boldsymbol{x}) \end{aligned} \tag{2.5}$$

la cual se reduce a la fórmula (2.4) en el caso, muy frecuente, de que la función de pérdida se defina como:

$$perd_{jk}(\boldsymbol{x}) = 1 - \delta_{jk} \ , \qquad \delta_{jk} = \begin{Bmatrix} 1 & j = k \\ 0 & j \neq k \end{Bmatrix} \tag{2.6}$$

El objetivo de un algoritmo de clasificación es buscar aquella estrategia, fórmula… de clasificación que minimice el "error de generalización" (error de test, o "suma de pérdidas" para casos no "vistos" previamente) al asignar los distintos casos a las clases.

Si se conoce que el caso $\boldsymbol{x}_n$ pertenece con certeza a la clase k, esta función de error se puede evaluar mediante:

$$f_{errorTest}(\quad) = \sum_{n=1}^{N} \sum_{j=1}^{J} perd_{jk}(\boldsymbol{x}_n) \,.\, p(y_j|\boldsymbol{x}) \tag{2.7}$$

donde se puede apreciar que la estrategia ha sido acumular las pérdidas de todos los posibles errores de clasificación y para todos los casos.

---

[11] En la literatura relacionada con los algoritmos de clasificación el concepto de "función de pérdida" es en cierta forma polisémico. El enfoque que aquí le damos sigue lo reflejado por Duda y Hart en su libro [14].

2.5 Solución ideal al problema de clasificación: el método de Bayes

La función de probabilidad "a posteriori" $p(y_j|\boldsymbol{x})$ o, más general, la función que hay que minimizar para conseguir una pérdida $PERD_j(\boldsymbol{x})$ óptima, pertenecen a un funcional donde no es fácil seleccionar cuál es la óptima.

En la práctica, a lo sumo se puede esperar estimar de forma indirecta la probabilidad de que se den un conjunto de valores $\boldsymbol{x}$ conociendo la clase a la que pertenece el caso, es decir la probabilidad condicionada: $p(\boldsymbol{x}|y_j)$.[12]

Ahora bien, el teorema de Bayes dice que si se conocen:

- Las probabilidades condicionadas $p(\boldsymbol{x}|y_j)$ de que se produzcan ciertos valores para los atributos $\boldsymbol{x}$, en función de las distintas clases y_j posibles[13].
- Y las probabilidades "a priori" $p(y_j)$ de que en este problema se presenten unas ciertas proporciones de casos para cada una de las clases.

es posible conocer $p(y_j|\boldsymbol{x})$, y por tanto clasificar los casos de forma óptima.

La bien conocida fórmula de Bayes facilita este cálculo:

$$p(y_j|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|y_j)\,p(y_j)}{\sum_{k=1}^{J} p(\boldsymbol{x}|y_k)\,p(y_k)} \tag{2.8}$$

La probabilidad "a posteriori" $p(y_j|\boldsymbol{x})$ señala lo probable que es que el caso determinado por los atributos $\boldsymbol{x}$ pertenezca a la clase y_j.

Una vez conocidas las probabilidades "a posteriori" para las *J* clases, el trabajo del algoritmo clasificador sería trivial: simplemente se asignaría el caso a la clase que presentase mayor probabilidad (o menor riesgo), de acuerdo con las fórmulas (2.4) o (2.5).

Este método no suele tener uso práctico ya que normalmente se desconocen las probabilidades $p(\boldsymbol{x}|y_j)$ y, su estimación, a partir de un conjunto normalmente escaso de datos empíricos, es un problema de muy difícil solución (sobre todo si la dimensionalidad del espacio de los atributos es alta).

El método de Bayes se utilizará en esta tesis para el análisis del problema sintético propuesto en la validación de la métrica LOM (ver sección 6.3.1).

---

[12] Por ejemplo, la probabilidad de que un enfermo tenga una temperatura superior a 40º sabiendo que ha contraído la malaria.

[13] A este término en inglés se lo conoce como "likelihood", que se podría traducir como "verosimilitud".

2.6 Modelo de un clasificador

Visto que el enfoque probabilístico no conduce a una metodología que resuelva siempre el problema, en la práctica, el abordar conceptualmente una tarea de clasificación supervisada comienza estableciendo un modelo de clasificador (alguno de los reflejados en la sección 2.2.3 u otro más específico para un determinado tipo de problema). Establecer un determinado modelo es, en muchos casos, una decisión personal del diseñador basada en su experiencia; en otras ocasiones la búsqueda en la literatura científica de cómo otros investigadores han resuelto problemas similares puede guiar esta selección.

La siguiente fase es ajustar, aprender, optimizar… los parámetros del modelo. En los métodos de clasificación supervisada para dar soporte a esta fase se emplea el conjunto de casos de aprendizaje y el algoritmo asociado al modelo del clasificador.

El último paso es comprobar el rendimiento real del clasificador y tomar la decisión de si es apto o no para la tarea encomendada.

A pesar de la existencia de múltiples modelos de clasificación muy distintos entre sí, existen una serie de conceptos comunes a todos ellos:

- Robustez.
- Escalabilidad.
- Sobreaprendizaje.
- El riesgo real de cometer un error.

En las siguientes secciones se expondrán cada uno de estos conceptos.

2.6.1 Robustez

El concepto de robustez [20] de un clasificador está ligado a su capacidad de mantener una alta precisión en la clasificación cuando los atributos de los casos están contaminados con ruido.

En el gráfico de la Figura 2.2 se puede apreciar cómo la recta que hace de separación entre las clases 1 y 2 seguiría sirviendo como criterio clasificador correcto, aunque los cuatro casos más "conflictivos" estuviesen contaminados con una cantidad de ruido tal que los hiciese moverse dentro de las circunferencias que los rodean.

Se puede considerar la robustez de un método de clasificación tanto durante la fase de aprendizaje (es decir la capacidad de determinar correctamente la función de separación entre clases cuando los datos de aprendizaje están contaminados con ruido), como la robustez en la fase de clasificación, es decir, cuando a un clasificador ya entrenado se le pide clasificar un caso nuevo contaminado con ruido.

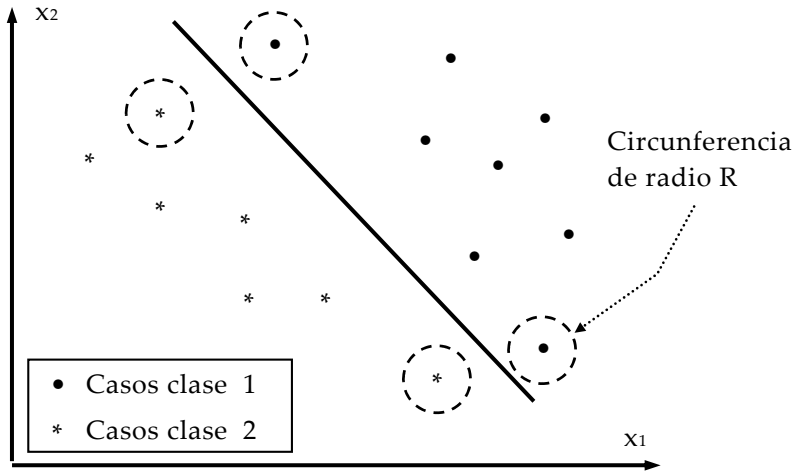


Figura 2.2.- Frontera de separación entre clases robusta frente a variaciones de los atributos de un caso dentro de una circunferencia de radio R.

En escenarios reales, y cuanto más "hostiles" sean estos a una toma precisa de datos, la robustez es una valiosa cualidad del clasificador (y en casos extremos, tan importante como la propia precisión de la clasificación).

2.6.2 Escalabilidad

El concepto de escalabilidad se refiere a la capacidad de un algoritmo para gestionar un incremento en el número de atributos o en el número de casos de un problema sin perder sus principales características (precisión de la clasificación, tiempo de cálculo, necesidades de memoria...).

La presencia de una gran cantidad de casos de aprendizaje[14], o la alta dimensionalidad de su espacio de atributos, son características de muchos problemas de clasificación actuales, sobre todo de aquellos relacionados con la visión artificial [21], la genómica…

Así pues, la primera de las dificultades relacionadas con la escalabilidad es el manejo de múltiples casos de aprendizaje, sobre todo en aquellos algoritmos que basan la clasificación de un nuevo caso en la exploración de todos los casos de aprendizaje (ya que aquellos que "resumen" en un conjunto pequeño de valores los parámetros a utilizar en el proceso de clasificación solo sufren este problema durante la fase de aprendizaje). En la sección 2.6.2.1 se describen un conjunto de algoritmos que mitigan la búsqueda de casos similares a uno dado en problemas con gran abundancia de elementos en la base de casos.

[14] En algunos problemas el número de casos podría ser tan grande que no sea posible mantenerlos en la memoria RAM y se necesite implementar un almacenamiento en disco, con su correspondiente "caché" en memoria RAM. Existen algoritmos de búsqueda rápida para estos escenarios, tal como el que se relata en [23].

El problema de la existencia de un número muy grande de atributos se lo conoce como "la maldición de la dimensionalidad" ("The curse of dimensionality"[15], también denominada "el problema COD"). Su descripción y las posibles medidas para contrarrestarla se exponen en la sección 2.6.2.2.

2.6.2.1 Problemas con gran cantidad de casos

Si el problema presenta tantos casos empíricos que el proceso de clasificación se convierte inviable en la práctica, se debe añadir al algoritmo base un método que permita acceder rápidamente a los casos de aprendizaje necesarios.

La idea clave para resolver el problema es buscar una forma paralela de representación de los casos de aprendizaje que permita una búsqueda rápida cuando estos sean requeridos [22]. Esta indexación (ordenación) debe permitir también al algoritmo de clasificación identificar cuándo un caso puede ser útil o no (para de esta forma minimizar el número de búsquedas).

La forma más habitual es emplear una estructura adicional de tipo árbol [23] (o lista). El objetivo de este árbol es dividir el espacio de atributos en bloques rectangulares ordenados que luego delimitarán la búsqueda de los casos requeridos.

En la siguiente tabla se recogen los principales métodos que permiten realizar este cometido.

	Estrategia	**Ventajas e inconvenientes**
Lista invertida [24]	Consiste en crear un índice por atributo y, para cada índice ordenar los distintos casos de menor a mayor. Las búsquedas (mediante "binary search") se realizan por cada atributo y luego se realizan las correspondientes intersecciones de conjuntos de casos para encontrar los deseados.	• Es fácil de implementar. • El número de casos encontrados que cumplen unas condiciones en las que aparecen varios atributos puede ser alto. • El tiempo de búsqueda aumenta fuertemente con el número de atributos.

[15] Como referencia primigenia se puede citar el libro de Bellman [152].

	Estrategia	Ventajas e inconvenientes
Rejilla fija [25]	Se fracciona cada atributo en un número fijo de divisiones. De esta forma todo el espacio de atributos queda dividido en celdas rectangulares. A cada una de estas celdas se le asocia una lista con los casos contenidos en ella, y a cada caso se le añade la información sobre a qué celda pertenece. En la fase de búsqueda solo será necesario explorar las celdas colindantes a la que contiene el caso dado (es inmediato conocer a qué celda pertenece un caso y a continuación conocer todos los casos de esa celda).	• Es fácil de implementar. • Si el número de atributos es muy grande la cantidad de celdas se dispara. • Muchas celdas pueden quedar vacías de casos y otras pueden estar densamente pobladas. • El número de casos recuperados puede ser relativamente grande.
"Locality-sensitive hashing" (LSH) [26]	Esta técnica aplica distintas funciones de "hash" (sensibles a la similitud entre casos) con la intención de que para cada función se produzcan colisiones en aquellos casos a indexar que sean más similares. Para cada función, y basándose en la similitud de estas claves, agrupa los casos en "cubos". De esta forma, los casos que están más próximos entre sí tienen una alta probabilidad de estar en el mismo "cubo". En el momento de clasificar un nuevo caso se le aplica una de las funciones de "hash" y se busca en el "cubo" que posee dicha clave. Posteriormente se buscan los vecinos próximos entre los recuperados en el paso anterior. El algoritmo se repite para las distintas funciones de "hash".	• Busca el vecino más próximo de forma aproximada. • Las funciones de "hash" son propias para cada métrica.

	Estrategia	Ventajas e inconvenientes
"Quad tree" [27]	Su estructura se basa en un árbol k-ario (donde $k=2^A$). En cada nodo se particiona el espacio 2^A en bloques rectangulares, correspondiendo una rama a cada una de las particiones. Para buscar un rango rectangular de casos próximos a uno dado se utiliza una técnica de podado: se abandona la búsqueda en una subrama si, cuando se llega a un nodo, dicha subrama no puede contener nodos en el rango rectangular deseado.	• Es un árbol de poca "profundidad". • La forma (y el equilibrado del árbol) depende del orden en que se inserten los casos en el árbol. • Existen muchos nodos vacíos.
"K-d tree" [28]	Su estructura es un árbol binario donde el espacio de atributos es particionado en cada nodo en función del valor de un solo atributo de un caso (los casos con menor valor para ese atributo quedarán en las ramas de la izquierda y los de mayor a la derecha). En las implementaciones más habituales, según se va descendiendo en el árbol se van turnando cíclicamente los atributos que sirven para tomar la decisión de cada nodo[16]. Como en los "quad trees", para buscar un rango rectangular de casos próximos a uno dado se utiliza una técnica de podado: se abandona la búsqueda en una subrama si, cuando se llega a un nodo, dicha subrama no puede contener nodos en el rango rectangular deseado.	• Mejores prestaciones que el "quad tree" para un número de atributos elevado (hasta 20 o 30 atributos). • En cada bifurcación solo utilizan un atributo (frente a los "quad tree" que emplean A). • El orden en que se van sucediendo las divisiones en cada rama determina la profundidad del árbol.

[16] En los "Generalized k-d trees" esto no es cierto.

	Estrategia	Ventajas e inconvenientes
"K-d tree" adaptativo [29]	Se diferencia del anterior en que en las divisiones de cada nodo se elige el atributo que presenta la mayor dispersión[17] de valores, y también el valor donde se producirá la división (que no tendrá que coincidir con el de algún caso existente).	• La estrategia empleada hace que el árbol quede más equilibrado y que, por tanto, las futuras búsquedas sean más rápidas.

Tabla 2.2.- Características de distintos métodos que aceleran la búsqueda de casos deseados.

En las conclusiones de la métrica BTW (sección 5.5) se establece una relación conceptual con la técnica LSH.

2.6.2.2 La maldición de la dimensionalidad

La dificultad que se plantea bajo este nombre es que la capacidad de predicción de aquellos algoritmos que emplean estimadores de clasificación locales se degrada exponencialmente según aumenta la dimensión del espacio de los atributos (si simultáneamente no se amplía en consonancia el número de casos empíricos de aprendizaje).

Es fácil comprender que la densidad de los casos, repartidos uniformemente en el espacio, es proporcional a $N^{1/A}$ y que esta decrece rápidamente cuando A se incrementa. Además, se obtiene la percepción de que los puntos se encuentran cerca de los límites del espacio. Por ejemplo, para una esfera de radio unitario en la que se encuentran distribuidos uniformemente N puntos, la mediana de la distancia de un caso que se encuentre en el centro a su vecino más próximo aumenta rápidamente hacia la unidad, de acuerdo a la siguiente fórmula:

$$\text{mediana} = \left[1 - \left(\frac{1}{2}\right)^{1/N}\right]^{1/A} \tag{2.9}$$

Así pues, este aspecto de la clasificación se revela como uno de los grandes retos que debe abordar un algoritmo basado en una métrica que pretenda resolver problemas reales.

La dimensión del espacio de atributos influye de forma fundamental en la "proximidad relativa" de un conjunto de casos.

[17] Por "dispersión" para los casos que quedan en esa rama se puede entender cualquier parámetro estadístico del atributo que se considere oportuno (varianza, rango…).

Así pues, si se opera un espacio de atributos de A dimensiones, el volumen ocupado por una esfera de radio r en este espacio es:

$$V = \frac{\pi^{A/2}\ r^{A}}{\Gamma(A/2+1)} \tag{2.10}$$

donde $\Gamma(x)$ es la función gamma de Legendre[18] y A la dimensión del espacio de atributos.

Otro ejemplo: si se dispone de N casos repartidos uniformemente en un cubo de arista 1, y se emplea una métrica euclídea, el radio medio de la esfera que pasa por el primer vecino de un caso dado es:

$$\overline{\text{radio}} = \left(\frac{A\ \Gamma(A/2)}{2\pi^{A/2}N}\right)^{1/A} \tag{2.11}$$

Del análisis de la fórmula (2.11) se puede deducir, y tal como se aprecia en la Figura 2.3 y en la Figura 2.4, que la distancia media al vecino más cercano aumenta rápidamente con el número de atributos A y que, para tratar de mantener una distancia media fija, es necesario aumentar exponencialmente el número de casos experimentales si tan solo se aumenta ligeramente el número de dimensiones.

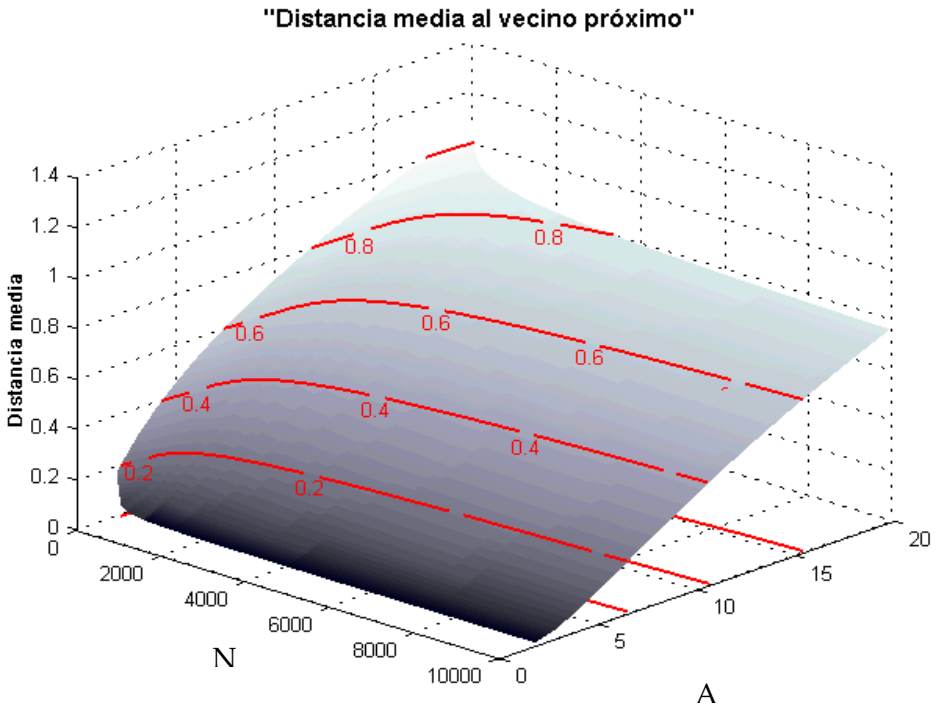


Figura 2.3.- Distancia media de un punto a su vecino inmediato en función de la dimensión del espacio de atributos (A) y del número de casos (N).

[18] Que para números enteros positivos se evalúa como $\Gamma(x) = (x-1)!$, cumpliéndose también que: $\Gamma(1) = 1$, $\Gamma(1/2) = \sqrt{\pi}$

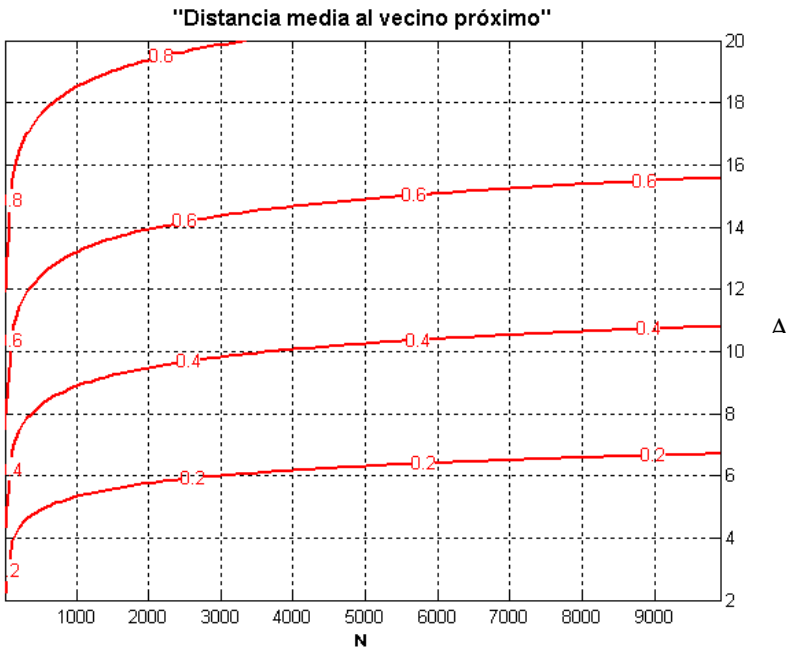


Figura 2.4.- Curvas de contorno en el plano N-A de la anterior Figura.

Para hacerse una idea de los órdenes de magnitud, en la siguiente tabla se recoge el número de casos necesarios *N* para mantener una distancia media de 0,2 unidades al ir aumentando el número de atributos *A*.

A	**2**	**4**	**6**	**8**	**10**
N	8	127	3.024	96.244	3.829.411

Tabla 2.3.- Número de casos prototipo N necesarios para mantener una distancia media constante (de 0,2 unidades) al vecino más cercano.

Se aprecia cómo el efecto de la dimensión causa estragos en el rendimiento de aquellos algoritmos que emplean una distancia como base para la clasificación.

No es de extrañar que muchos de los algoritmos del estado del arte que se han estudiado en esta investigación: VSM, DANN, LaMaNNa... lo hayan tenido en cuenta en su diseño.

2.6.2.2.1 Estrategias para mitigar el problema COD

En los escenarios reales se dan un par de circunstancias que inducen al problema COD:

- En ciertos problemas (visión, genómica…), el número de atributos suele ser bastante elevado (por esta razón, en algunos escenarios, se procede a un preprocesamiento de los datos).
- La cantidad de casos de aprendizaje experimentales suele estar limitada por el costo que supone su adquisición. Es decir, en muchos casos no es muy grande.

La confluencia de estos dos factores suele conducir a problemas de solución compleja. Pero en los problemas también se pueden presentar otras tres características que pueden ayudar a mitigar dicha dificultad:

- Algunos de los atributos pueden ser irrelevantes para la clasificación, por ejemplo: medidas que solo contienen ruido o que no tienen ninguna conexión con el problema actual. Si se prescindiese de ellos no habría ninguna pérdida de información y la dimensionalidad del problema se reduciría.
- Pueden existir atributos que presenten una fuerte correlación entre ellos. Esto significa que, en el fondo, todos ellos están portando la misma información y, por lo tanto, si se eliminasen todos, excepto uno, la calidad de la clasificación no se resentiría. En otros problemas, el valor de uno de los atributos se puede aproximar razonablemente bien mediante una combinación lineal de los valores de otro conjunto de ellos. Siguiendo el mismo razonamiento anterior, se podría prescindir de este atributo y reducir así el número de dimensiones total.
- Aunque un atributo no cumpla con alguna de las anteriores características, si su relevancia es escasa, también podría considerarse beneficioso "reducir al máximo" el número de dimensiones y prescindir de él. En las estrategias de aprendizaje supervisado, este enfoque se puede contemplar bajo el prisma de la reducción del número de parámetros a entrenar y por tanto la reducción del riesgo estructural (ver sección 2.6.4).

Estas reflexiones generales sobre la importancia de un atributo se han plasmado, en la literatura científica, como distintos algoritmos de preprocesado de datos que se han usado para luchar contra la maldición de la dimensionalidad, entre ellas se pueden citar:

- Selección de atributos: quizás una de las estrategias más primitivas fue eliminar del conjunto de atributos aquellos que no tuviesen una fuerte influencia en la clasificación. De esta forma, no solo se reducía el número de dimensiones, sino también se evitaban aquellos atributos muy contaminados por ruidos (o redundantes que no aportaban nueva información).
- Análisis de componentes principales (PCA) [30]: consiste en llevar a cabo una transformación lineal que proyecta los valores originales de los atributos en un subespacio que tiene como ejes principales aquellos que se orientan en las direcciones de máxima varianza de dichos valores.

  Esto se logra diagonalizando la matriz de covarianzas $\boldsymbol{C}$ (donde $\boldsymbol{\mu}$ es el vector de medias de los distintos atributos):

$$\boldsymbol{C} = \frac{1}{N}\sum_{n=1}^{N}(\boldsymbol{x}_n - \boldsymbol{\mu})(\boldsymbol{x}_n - \boldsymbol{\mu})^T \qquad (2.12)$$

Y posteriormente obteniendo sus valores y vectores propios.

El número de dimensiones de este subespacio se elige para cada problema. Para facilitar esta tarea simplemente se ordenan los vectores propios por orden descendente de su valor propio asociado, y se seleccionan los M primeros[19].

Al proyectar el conjunto de atributos (que residen en un espacio de dimensión A) en otro subespacio de dimensión inferior (M), el problema de la maldición de la dimensionalidad queda mitigado[20].

El mayor problema práctico asociado es que los "nuevos atributos" no se identifican con magnitudes que posean significado para el usuario final de la solución de clasificación.

Además, el hecho de que los "atributos nuevos" tengan gran varianza y nula correlación con los demás, no significa que su capacidad de predecir la clase a la que pertenece un caso sea la óptima. En el problema de la Figura 2.5 las curvas de equidistancia en el espacio transformado se alargan en la dirección que ayudaría a clasificar casos nuevos. Por lo tanto, en este caso, la aplicación de las direcciones PCA sí ayudaría a clasificar mejor los casos.

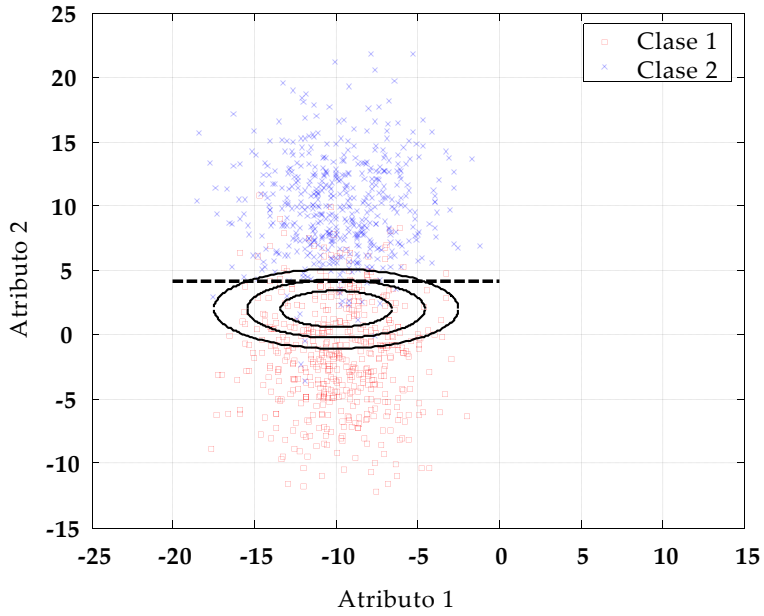


Figura 2.5.- "Kernel" orientado según las direcciones correspondientes a los vectores propios de la matriz de covarianzas (longitudes de los ejes de las curvas de equidistancia inversamente proporcionales a los valores propios). La línea recta indica la frontera óptima de separación de las dos clases.

[19] Siendo este valor M decisión del investigador.

[20] Y, de paso, elimina las correlaciones entre los "nuevos atributos" en el espacio proyectado.

Pero para otro problema diferente, la misma estrategia puede comportarse de forma completamente opuesta. En la Figura 2.6 se puede apreciar otro problema sintético bidimensional, donde en este escenario los casos se distribuyen de una forma que tiende a ser paralela al eje de abscisas.

Como se puede comprobar gráficamente, en esta situación las curvas de equidistancia en el espacio transformado que se ven en la figura, no solo no ayudarían a clasificar mejor los casos, sino que empeoraría el rendimiento del algoritmo de clasificación.

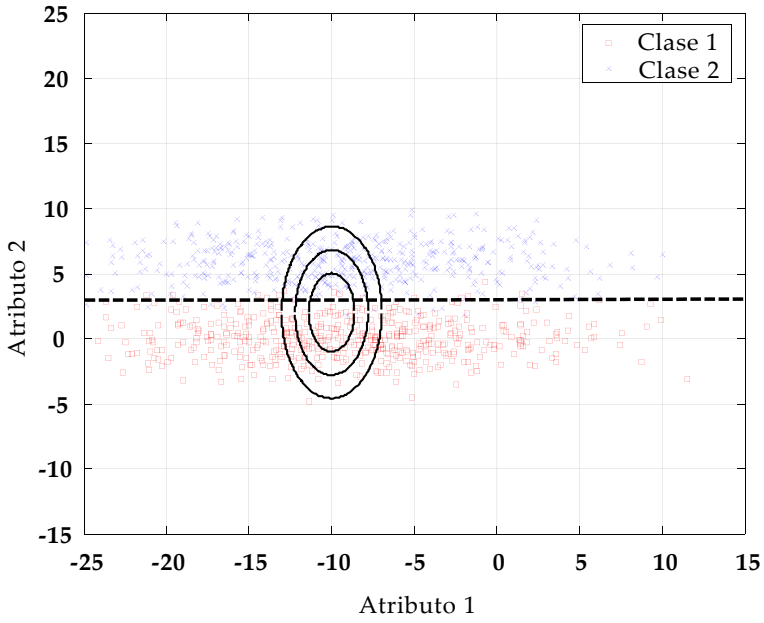


Figura 2.6.- Contraejemplo sobre las posibilidades de emplear los valores y vectores propios del PCA como ejes del "Kernel" orientado de acuerdo a esas direcciones (las longitudes de los ejes de las curvas de equidistancia son inversamente proporcionales a los valores propios).

- Imponer restricciones en los modelos a ajustar: muchos de los algoritmos de clasificación imponen condiciones sobre las propiedades de los casos. Por ejemplo, el algoritmo LDA exige que los atributos de los casos de las distintas clases pertenezcan a distribuciones normales multidimensionales con la misma matriz de covarianzas.

  Estas condiciones no se suelen cumplir en los casos reales, lo cual cuestiona teóricamente la aplicabilidad del algoritmo. Pero, por otra parte, también reduce mucho el número de parámetros libres a estimar/optimizar.

2.6.3 Sobreaprendizaje

El objeto final de un modelo de clasificación podría ser el proporcionar una función algebraica de separación entre clases (también denominada función de decisión). En algunos clasificadores, como en las máquinas de vectores de soporte (ver sección 4.2.3) o en el análisis del discriminante (ver sección 4.2.2), dicha función es el resultado real del método de clasificación; pero en otros casos, como en la clasificación basada en vecinos próximos (ver sección 4.2.1), esta función no sería algebraica, pero sí se podría visualizar por medio de los diagramas de Voronoi.

El problema del sobreaprendizaje (sobreajuste) procede de entrenar repetidamente el modelo, mediante un algoritmo de optimización de los parámetros, en aras a conseguir discriminar con mayor perfección la clase de los casos de aprendizaje. Según se puede apreciar en la Figura 2.7, la superficie, vamos a suponer "correcta", de separación entre clases (curva de trazo continuo) no realiza su labor perfectamente para todos los casos de aprendizaje, pero es muy posible que represente mejor el criterio causal de clasificación subyacente en el problema que la curva "sobreajustada" (que aparentemente realiza su trabajo a la perfección con los casos de aprendizaje).

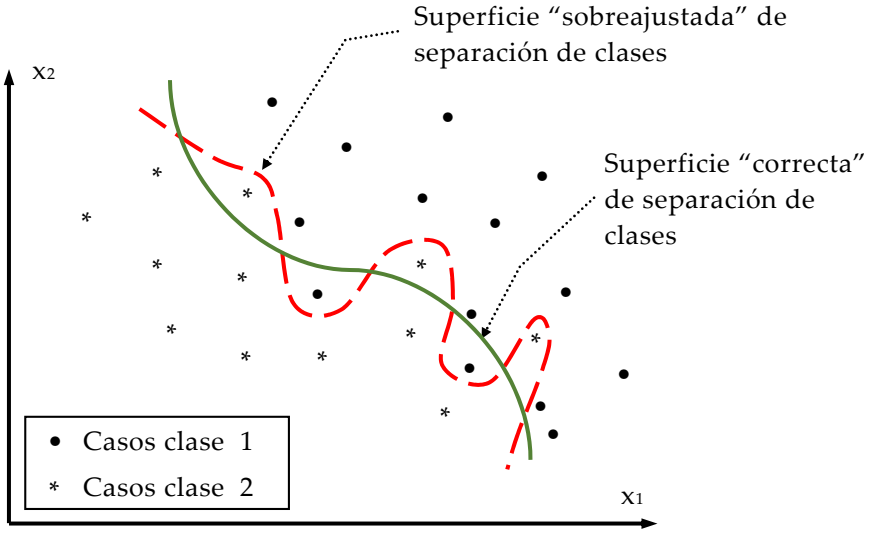


Figura 2.7.- Superficie de separación de clases correcta (trazo continuo) y con sobreajuste (trazo discontinuo).

Cuando el clasificador se aplica a casos nuevos (no pertenecientes al conjunto de aprendizaje) las funciones de decisión sobreajustadas proporcionan resultados mucho peores a los esperados.

Las causas del sobreaprendizaje radican en la "gran expresividad" del modelo clasificador (que podría adaptarse a casi cualquier tipo de superficie), a la contaminación con ruido de los valores de los atributos y las clases, y a la finitud del conjunto de casos de aprendizaje.

La siguiente sección trata de formalizar este problema desde el enfoque proporcionado por Vapnik a finales del siglo XX; el denominado principio de la minimización del riesgo estructural.

2.6.4 La minimización de riesgo estructural

Una de las características deseables para un algoritmo de clasificación es que sea consistente, es decir, que al ir aumentando el número de casos de aprendizaje, el error cometido al clasificar los casos nuevos fuese progresivamente disminuyendo. La consistencia requiere que, asintóticamente, se fuesen aproximando el error de clasificación calculado con los casos de aprendizaje y el obtenido con los casos de test.

Este aspecto requiere considerar por una parte cómo se calcula el error empírico (el obtenido mediante los casos de aprendizaje); y cómo, la función que implementa el algoritmo de la clasificación, incrementa dicho error empírico para dar lugar al error real esperado (el calculado si se dispusiese de infinitos casos de test)[21].

2.6.4.1 El riesgo empírico

Durante muchas décadas, optimizar los parámetros de un método de clasificación o regresión (en aras a mejorar su precisión) significaba casi exclusivamente buscar el conjunto de valores para los que el error medio en la clasificación de los datos de aprendizaje resultase más pequeño[22].

$$E_{emp}(\boldsymbol{\mathcal{P}}) = \frac{1}{N}\sum_{i=1}^{N} c_{f_{dec}(\boldsymbol{x}_i,\boldsymbol{\mathcal{P}})}(\boldsymbol{x}_i) \tag{2.13}$$

donde $E_{emp}(\boldsymbol{\mathcal{P}})$ es el valor del error empírico de la clasificación y $f_{dec}(\boldsymbol{x}_i, \boldsymbol{\mathcal{P}})$ es la función de decisión que clasifica al caso i-ésimo partiendo de los valores de sus atributos y $c_{y_j}(\boldsymbol{x}_i)$ vale 1 cuando el caso $\boldsymbol{x}_i$ pertenece a la clase y_j.

Al error empírico se lo puede conectar con el concepto de "error de clasificación en este problema dada una cierta función de pérdida"; cuando se lo usa bajo este más amplio significado se suele usar el nombre de riesgo empírico: $R_{emp}(\boldsymbol{\mathcal{P}})$.

---

[21] En algunos artículos a este error real esperado también se llama: "error garantizado", "error de generalización"...

[22] Las estrategias que se basan en minimizar solo este término se encuadran dentro de las técnicas conocidas como "de reducción del riesgo empírico" ("*Empirical Risk Minimization*", ERM). El MSE aplicado sobre los datos de aprendizaje es una de las muchas formas de expresar el riesgo empírico, como también podrían ser los métodos de máxima verosimilitud (ML) en las estimaciones de funciones de densidad de probabilidad...

Bajo este enfoque, el conjunto de parámetros óptimo será aquel que cumpla:

$$\boldsymbol{\mathcal{P}}_{opt} = arg\,\min_{\boldsymbol{\mathcal{P}}}\ R_{emp}(\boldsymbol{\mathcal{P}}) \tag{2.14}$$

La minimización, sin otras consideraciones, del error empírico conduce habitualmente al problema del sobreaprendizaje. El método de clasificación acaba seleccionando un conjunto de parámetros que, en lo posible, "memorizan" las clases específicas de cada uno de los casos de aprendizaje, en vez de intentar buscar la "verdadera" función de clasificación (decisión) que subyace en el problema. Bajo esta consideración, es claro que el error empírico calculado puede ser mucho más pequeño que el error real esperado para ese problema.
De hecho, cuando a una función que sufre de sobreaprendizaje se le presentan nuevos casos (no utilizados en el aprendizaje) su error de clasificación suele ser mucho más elevado que el estimado con los casos de aprendizaje.

Así pues, el error de clasificación real esperado puede mejorar o no al incrementar el número de casos de aprendizaje; es decir su consistencia no depende del valor del riesgo empírico. Solo será consistente si:

$$\lim_{N\to\infty}\ p\left\{\sup_{\boldsymbol{\mathcal{P}}}\ \left(R(\boldsymbol{\mathcal{P}})-R_{emp}(\boldsymbol{\mathcal{P}})\right) > \epsilon\right\} = 0 \qquad \forall\,\epsilon \tag{2.15}$$

donde $R(\boldsymbol{\mathcal{P}})$ es el riesgo real esperado.

Para evitar el sobreaprendizaje, tradicionalmente los métodos de aprendizaje supervisados utilizan técnicas de validación, como las reseñadas en la sección 1.3.3, para seleccionar el mejor modelo; o técnicas de regularización en la que se procura que los valores de los parámetros estén restringidos por ciertas reglas. El "decaimiento" de los pesos y la limitación de los valores de las derivadas primeras y/o segundas de la función de decisión son ejemplos de estas últimas técnicas.

$$E_{reg}(\boldsymbol{\mathcal{P}}) = E_{emp}(\boldsymbol{\mathcal{P}}) + \lambda\,\phi\big(f_{dec}(\boldsymbol{\mathcal{P}})\big) \tag{2.16}$$

donde λ es un factor de penalización y $\phi\big(f_{dec}(\boldsymbol{\mathcal{P}})\big)$ un funcional que intenta "restringir la variabilidad" de la función de decisión.

2.6.4.2 El riesgo estructural

En 1992 Vladimir Vapnik [12] presentó un nuevo enfoque para abordar este problema denominado minimización del riesgo estructural ("Structural Risk Minimization", SRM) en el que lo que realmente se busca minimizar es el error real esperado (o "error de generalización").

De acuerdo a Vapnik, el error de generalización se define como la cota superior de la suma de error de aproximación (que se comete al elegir modelo o la función y los parámetros mediante el que se estima la clase de un caso) más el error de estimación que es provocado porque la cardinalidad del conjunto de datos de aprendizaje no es infinita (error de aprendizaje empírico). Habitualmente este error de generalización es imposible de cuantificar, pero sirve como idea de guía en el diseño de los algoritmos.

A modo de ejemplo, y expresado en forma de riesgos, dicha cota superior para las técnicas de separación por planos sería:

$$R(\boldsymbol{\mathcal{P}}) \leq R_{emp}(\boldsymbol{\mathcal{P}}) + 2\,\upsilon^2\left(N, h_{f_{dec}(\boldsymbol{\mathcal{P}})}, \eta\right)\left(1 + \sqrt{1 + \frac{R_{emp}(\boldsymbol{\mathcal{P}})}{\upsilon^2\left(N, h_{f_{dec}(\boldsymbol{\mathcal{P}})}, \eta\right)}}\right) \qquad (2.17)$$

donde $R(\boldsymbol{\mathcal{P}})$ es el riesgo real esperado de cometer un error al clasificar los datos mediante una determinada función de decisión y $\upsilon\left(N, h_{f_{dec}(\boldsymbol{\mathcal{P}})}, \eta\right)$ es la denominada "confianza VC" (ver ecuación (2.18)), que evalúa el riesgo asociado a la función de decisión elegida (y que depende del número de casos de aprendizaje N, de la dimensión VC h, y de un término pequeño $0 < \eta < 1$).

En la siguiente Figura se visualiza esta idea. El riesgo total esperado para los casos de test (línea continua gruesa) está compuesto de un error que introduce el propio modelo por sobreajustarse a los datos (error de aproximación) y otro que tiene que ver con el error de ajuste de los datos empíricos a ese modelo. El punto mínimo de la curva de riesgo real indicaría la complejidad óptima del modelo a utilizar.

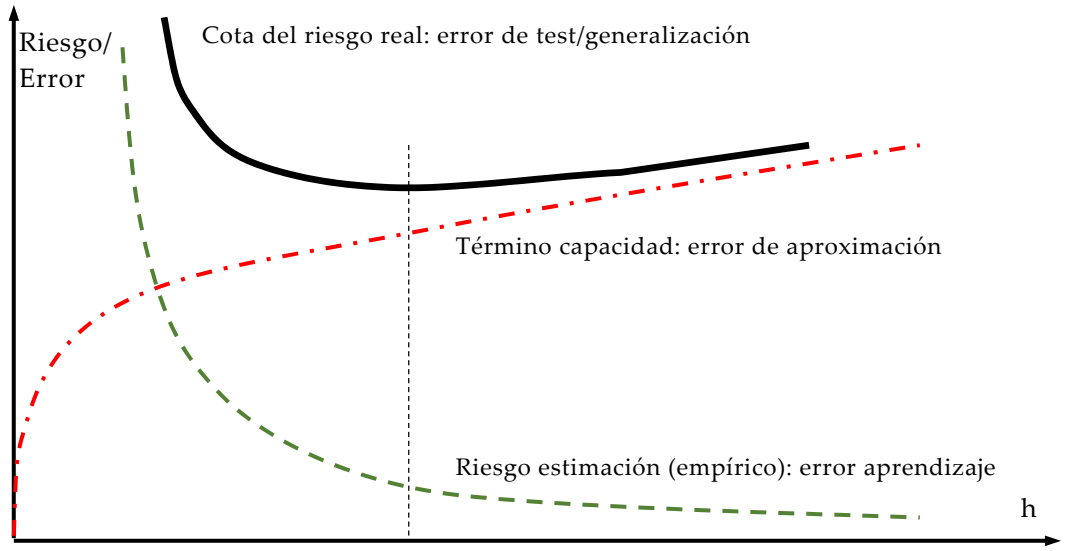


Figura 2.8.- Error de aprendizaje, error de aproximación y error de test en función de la complejidad del modelo (casos que puede separar).

De este enfoque se puede obtener una doble reflexión:

- Por una parte se debe tratar de seleccionar una función de decisión[23] que controle el sobreaprendizaje (sobre todo cuando el número de casos de aprendizaje es relativamente pequeño). La función elegida debería ser aquella, no que proporcionase el menor error empírico, sino la que se correspondería con el mínimo del error de generalización al entrenar el modelo.
- Por otra, sería muy interesante conocer, sin disponer de casos de test, cuál sería el error real de clasificación. Eso se conseguiría si al riesgo empírico se le pudiese sumar un término que nos permitiese estimar el riesgo real, tal como se aprecia en la ecuación (2.17).

Antes de presentarse esta teoría ya habían nacido conceptos relacionados, como la "dimensión de Vapnik-Chervonenkis" (VC) para un conjunto de funciones $\mathcal{L}$, la cual se define como el número máximo de casos (h) que pueden ser separados en dos clases diferentes (en todas las 2^h combinaciones posibles) por ese conjunto de funciones $\mathcal{L}$. La dimensión VC para un algoritmo mide su capacidad y complejidad para representar clasificaciones muy diferentes, y también su tendencia al sobreaprendizaje (principalmente cuando el número de casos es pequeño).
Así existen expresiones como la "confianza VC" (utilizada en la ecuación (2.17)), que expresan la tendencia de una determinada función de decisión de aumentar el riesgo real esperado:

$$v\left(N, h_{f_{dec}(\boldsymbol{P})}, \eta\right) = \sqrt{\frac{h_{f_{dec}(\boldsymbol{P})}.\left(\ln\left(2N/h_{f_{dec}(\boldsymbol{P})}\right)+1\right) - \ln(\eta/4)}{N}} \qquad (2.18)$$

donde N es el número de casos de aprendizaje, $h_{f_{dec}(\boldsymbol{P})}$ es la dimensión VC de la función de decisión (incluyendo los parámetros que esta usen) y η es un valor pequeño comprendido entre 0 y 1 que luego se ligará con la probabilidad de cometer un cierto error.

La aplicabilidad de la dimensión VC en la teoría SRM la establece Vapnik por medio del control de la capacidad de generalización:

> *"To achieve the smallest bound on the test error by controlling (minimizing) the number of training errors, the machine (the set of functions) with the smallest VC dimension should be used".*

Así pues, reducir el porcentaje de error de clasificación de los datos de aprendizaje y mantener una dimensión VC pequeña son objetivos contradictorios. Como se ha indicado en la sección 2.6.3, si se dispone de un conjunto de funciones que permitan clasificar perfectamente los datos de

[23] O sus parámetros, método de optimización...

aprendizaje, el error empírico será cero, pero la dimensión VC será muy grande, y la capacidad de ese modelo de generalizar estará en entredicho.

Si se ajustase con los datos de aprendizaje una secuencia de modelos (que en este caso difieren solo en los parámetros) con dimensión VC creciente:

$$f_{dec}(\boldsymbol{\mathcal{P}}_1) < f_{dec}(\boldsymbol{\mathcal{P}}_2) < \cdots < f_{dec}(\boldsymbol{\mathcal{P}}_i) < \cdots \tag{2.19}$$

Y para cada uno de ellos se calculasen sus riesgos reales: $R(\boldsymbol{\mathcal{P}_i})$, se asumirá como mejor modelo aquel que presente la menor cota superior para el riesgo de generalización:

$$\boldsymbol{\mathcal{P}}_{opt} = \underset{\boldsymbol{\mathcal{P}_i}}{arg\,min}\ \left(R(\boldsymbol{\mathcal{P}_i})\right) \tag{2.20}$$

Como última reflexión, se puede indicar que este enfoque de Vapnik reformuló una vieja idea del siglo XIV conocida como la "navaja de Occam": la explicación más sencilla es la mejor de todas las posibles (que también es conocida como el principio de parsimonia).

3 Fundamentos matemáticos

Este apartado quiere complementar al anterior recogiendo cuestiones transversales en el mundo de las medidas de distancia aplicadas a la clasificación. Muchas de estas técnicas son las que actualmente sustentan, y en el futuro orientarán, investigaciones en este campo.

Así pues, en la sección 3.1 se recoge una taxonomía de las técnicas de optimización, y a continuación se explican con más detalle las distintas técnicas que han tenido relación con esta tesis: optimización sin restricciones para ecuaciones no lineales, optimización con restricciones lineales y la programación semidefinida.

En la sección 3.2 se expondrán de forma breve los aspectos más importantes de las geometrías riemannianas; teoría que se usa de forma muy importante en la métrica LOM.

3.1 Técnicas de optimización

En la mayor parte de los algoritmos, tanto en los estudiados en el estado del arte, como en los desarrollados en esta investigación, se emplean algoritmos de optimización.

En esta tesis, el objetivo es minimizar una función de error f_{err}, normalmente el error en la clasificación empírica de los casos (o alguna de sus variantes):

$$\min_{\boldsymbol{\mathcal{P}}} \; f_{err}(\boldsymbol{x}, \boldsymbol{\mathcal{P}}) \tag{3.1}$$

De acuerdo al "Wisconsin Institute for Discovery", una posible taxonomía para las técnicas de optimización podría ser la reflejada en la Figura 3.1 [31].

No es objeto de esta tesis detallar cada una de estas técnicas, nos limitaremos a describir aquellas que están directamente relacionadas con esta tesis, para una referencia completa el lector puede consultar [32]. Todas las optimizaciones que usamos son determinísticas para atributos continuos:

- Optimización sin restricciones para ecuaciones no lineales.
- Optimización con restricciones lineales.
- Programación semidefinida.

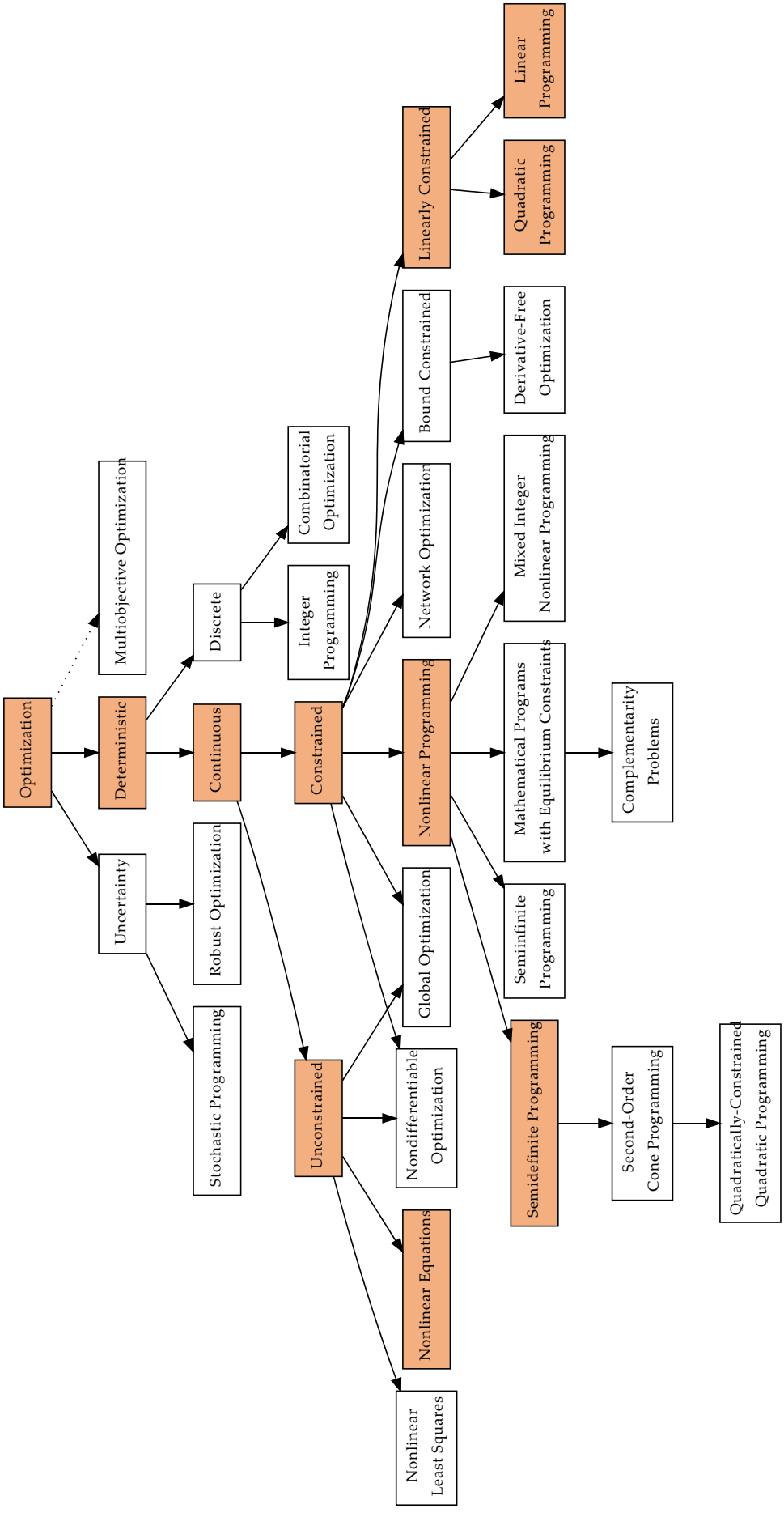


Figura 3.1.- Taxonomía de los métodos de optimización de acuerdo al "Wisconsin Institute for Discovery".

3.1.1 Optimización sin restricciones para ecuaciones no lineales

Bajo el epígrafe de optimización sin restricciones se engloban todos aquellos métodos que no imponen límites en los valores que pueden tomar los parámetros a optimizar.

En algunos problemas estos métodos se aplican directamente sobre la función objetivo, pero en otros se emplean sobre reformulaciones de problemas de optimización con restricciones, en los que estas han sido reemplazadas por términos que penalizan su desviación fuera de los rangos permitidos.

Otro aspecto preliminar importante es si la función de error tiene derivadas de primer y segundo orden continuas. Este aspecto facilitará el planteamiento teórico de su resolución.

La mayor parte de los métodos que se emplean en este tipo de minimización emplean un proceso iterativo:

- Se comienza en un punto del espacio de parámetros $\boldsymbol{\mathcal{P}}^{(0)}$ lo más cercano posible al mínimo que se desea encontrar.
- Se aplica el algoritmo de minimización en cuestión, encontrándose un nuevo valor para los parámetros $\boldsymbol{\mathcal{P}}^{(1)}$, para los cuales la función de error presenta un valor menor.
- Continuar con este procedimiento, generando una secuencia de iteraciones $\boldsymbol{\mathcal{P}}^{(i)}$, hasta que no se consiga rebajar el valor de la función objetivo.

Existen distintos métodos para realizar este tipo de optimización que se repasan en los siguientes subapartados.

3.1.1.1 Métodos de búsqueda exhaustiva

Son adecuados cuando el número de parámetros a optimizar es pequeño (normalmente uno o dos). Su estrategia[1] consiste en ir evaluando sistemáticamente la función de error en distintos emplazamientos del espacio de parámetros, registrando los valores que va tomando, y luego eligiendo el valor mínimo. Existen variantes que pasan por la simple exploración en una malla uniformemente distribuida o más sofisticadas versiones, como la búsqueda de Fibonnanci [33] o la búsqueda mediante la sección áurea [33] [34], que son efectivas para reducir el número de evaluaciones de la función cuando se conoce que en un determinado intervalo solo existe un mínimo.

Este método tiene como ventaja que solo es necesario evaluar la función a optimizar, no es necesario calcular derivadas primeras ni segundas.

Se usa en la optimización de los parámetros C y γ en las SVM-RBF [35], en técnicas de criptografía [36]...

[1] También conocido como algoritmo de "fuerza bruta".

En esta investigación se ha usado la búsqueda mediante la sección áurea en la optimización de los parámetros del algoritmo BTW, y la búsqueda en malla en el entrenamiento de las SVM.

3.1.1.2 Método del gradiente conjugado

Tratando de evitar el cálculo de derivadas segundas, el método del gradiente conjugado implementa la búsqueda en direcciones conjugadas para ir descendiendo hacia el mínimo.

Así pues, en un determinado punto del espacio de parámetros se calcula la próxima dirección conjugada[2] $\boldsymbol{s}^{(i)}$ y se procede a una búsqueda del valor mínimo a lo largo de esa línea:

$$\alpha_{opt} = \min_{\alpha} f_{err}\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)} + \alpha\, \boldsymbol{s}^{(i)}\right) \tag{3.2}$$

A continuación se actualiza el valor de los parámetros:

$$\boldsymbol{\mathcal{P}}^{(i+1)} = \boldsymbol{\mathcal{P}}^{(i)} + \alpha_{opt}\, \boldsymbol{s}^{(i)} \tag{3.3}$$

En ese nuevo punto se calcula el gradiente:

$$\boldsymbol{gr}^{(i+1)} = \nabla f_{err}\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i+1)}\right) \tag{3.4}$$

Y se computa la siguiente dirección conjugada $\boldsymbol{s}^{(i+1)}$ mediante[3]:

$$\beta_{FR}^{(i+1)} = \frac{\left(\boldsymbol{gr}^{(i+1)}\right)^T \boldsymbol{gr}^{(i+1)}}{(\boldsymbol{gr}^{(i)})^T\, \boldsymbol{gr}^{(i)}} \qquad Fletcher - Reeves \tag{3.5}$$

$$\beta_{PR}^{(i+1)} = \frac{\left(\boldsymbol{gr}^{(i+1)}\right)^T \left(\boldsymbol{gr}^{(i+1)} - \boldsymbol{gr}^{(i)}\right)}{(\boldsymbol{gr}^{(i)})^T\, \boldsymbol{gr}^{(i)}} \qquad Polak - Ribière \tag{3.6}$$

$$\boldsymbol{s}^{(i+1)} = -\boldsymbol{gr}^{(i+1)} + \beta_{xx}^{(i+1)} \boldsymbol{s}^{(i)} \tag{3.7}$$

procediéndose a realizar nuevas iteraciones hasta obtener un mínimo en la función de error, o a que se estabilicen los valores de los parámetros:

$$\left|\boldsymbol{\mathcal{P}}^{(i+1)} - \boldsymbol{\mathcal{P}}^{(i)}\right| < \varepsilon \tag{3.8}$$

En la minimización de funciones no lineales, periódicamente se procede a reinicializar la próxima dirección conjugada.

En la Figura 3.2 se ilustra el funcionamiento de este algoritmo para un paraboloide que se representa por medio de sus curvas de nivel. Se supone que se comienza desde el punto $\boldsymbol{x}_0$, y desde ese punto se procede una búsqueda en la dirección del gradiente negativo (línea s_0), hasta llegar al punto $\boldsymbol{x}_1$. Aquí se calcula el gradiente y se procede a calcular la siguiente

---

2 En la primera iteración, la dirección conjugada es la del gradiente cambiada de signo.

3 Normalmente cuando el cálculo de $\beta_{xx}^{(i+1)}$ devuelve un valor negativo, se lo convierte en cero.

dirección de movimiento (línea s_1) aplicando una de las fórmulas (3.5) o (3.6) y luego la (3.7). Se vuelve a buscar el mínimo a lo largo de la curva, hasta llegar al punto $\boldsymbol{x}_2$. Las iteraciones continúan hasta llegar al mínimo $\boldsymbol{x}_f$.

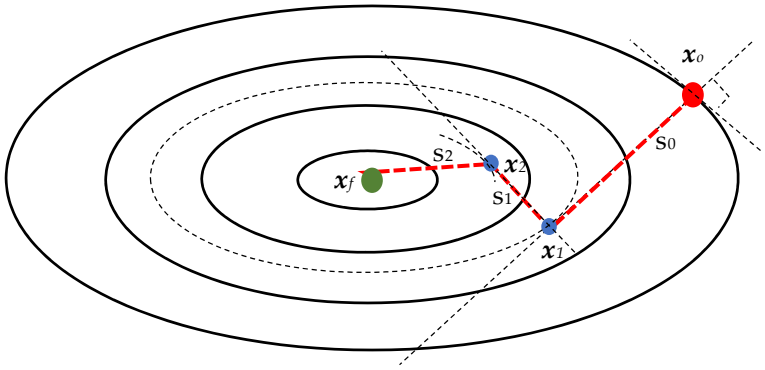


Figura 3.2.- Búsqueda del mínimo de la función por el método del gradiente conjugado.

Esta técnica de optimización se usa en la explicación del algoritmo VSM (sección 4.4.1).

3.1.1.3 Método de Newton

Es el más antiguo de los métodos de optimización basados en las derivadas de la función de error. Se basa en un proceso iterativo que emplea las derivadas primera y segunda de la función a minimizar respecto a los distintos parámetros para acelerar la búsqueda del mínimo.

$$\boldsymbol{\mathcal{P}}^{(i+1)} = \boldsymbol{\mathcal{P}}^{(i)} - \left(\nabla^2 f_{err}\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)}\right)\right)^{-1} \nabla f_{err}\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)}\right) \tag{3.9}$$

donde $\boldsymbol{\mathcal{P}}^{(i)}$ es el vector con los valores de los parámetros a optimizar (iteración i-ésima). $\nabla f_{err}(\ )$ es el gradiente de la función de error y $\nabla^2 f_{err}(\ )$ es el hessiano de dicha función.

El proceso iterativo continúa hasta que la norma del gradiente esté por debajo de un valor prefijado:

$$\left\|\nabla f_{err}\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)}\right)\right\| < \varepsilon \tag{3.10}$$

Su principal inconveniente es que hay que calcular el hessiano de la función de error (bien sea analítica o numéricamente) e invertirlo. Para evitar el cálculo del hessiano existen métodos denominados genéricamente "cuasi-Newton"[4] que van actualizando una matriz aproximada a la inversa del hessiano (de esta forma se evita tanto el cálculo de derivadas segundas como el tener que invertir una matriz, muchas veces mal condicionada).

La interpretación geométrica del método de Newton se puede ver en la Figura 3.3. Si se parte del punto $\boldsymbol{x}_0$ y en él se calcula el gradiente y el hessiano, el

[4] El más usado habitualmente es el conocido como BFGS en honor a sus autores Broyden, Fletcher, Goldfarb y Shanno.

método de Newton, considera que hay que minimizar el paraboloide que pasa por ese punto y presenta el mismo gradiente y hessiano en ese punto. Esto lleva a calcular el punto x_1, donde se repite el algoritmo hasta llegar al mínimo de la función x_f.

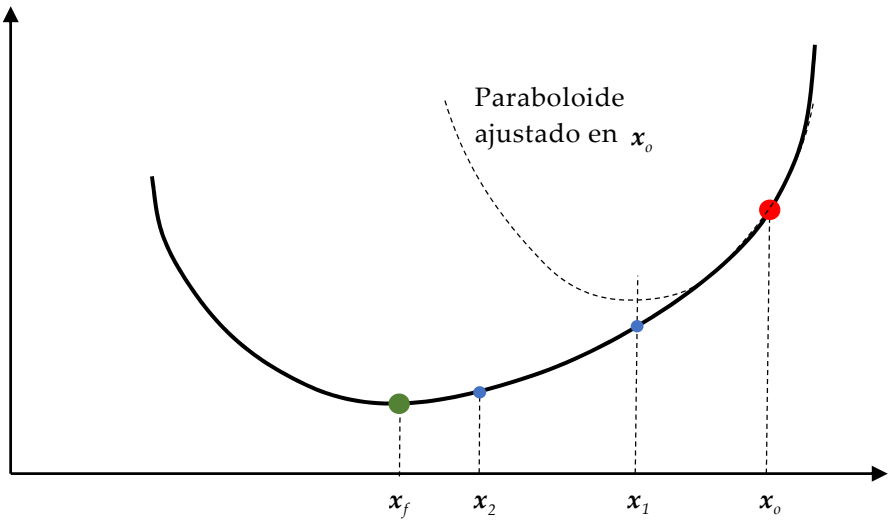


Figura 3.3.- Búsqueda del mínimo por el método de Newton.

3.1.1.4 Método de Levenberg–Marquardt

Es quizás el algoritmo de elección cuando la función de error a minimizar es una suma de cuadrados, escenario típico en un problema de clasificación en la minimización de sumas de las diferencias entre el valor real y el predicho elevado al cuadrado [37] [34].

$$f_{err}(\boldsymbol{x},\boldsymbol{\mathcal{P}}) = \sum_{i=1}^{N} \vartheta_i^2(\boldsymbol{x},\boldsymbol{\mathcal{P}}) = \boldsymbol{\vartheta}^T(\boldsymbol{x},\boldsymbol{\mathcal{P}})\boldsymbol{\vartheta}(\boldsymbol{x},\boldsymbol{\mathcal{P}}) \tag{3.11}$$

Así pues, el gradiente de esta función de error será:

$$\nabla f_{err}(\boldsymbol{x},\boldsymbol{\mathcal{P}}) = 2\sum_{i=1}^{N} \vartheta_i(\boldsymbol{x},\boldsymbol{\mathcal{P}}) \frac{\partial \vartheta_i(\boldsymbol{x},\boldsymbol{\mathcal{P}})}{\partial\, \boldsymbol{\mathcal{P}}} = 2\,\boldsymbol{\mathcal{J}}^T(\boldsymbol{x},\boldsymbol{\mathcal{P}})\,\boldsymbol{\vartheta}(\boldsymbol{x},\boldsymbol{\mathcal{P}}) \tag{3.12}$$

donde $\boldsymbol{\mathcal{J}}(\boldsymbol{x},\boldsymbol{\mathcal{P}})$ es el jacobiano del vector de funciones individuales de error (con derivadas parciales respecto a los distintos parámetros).

Partiendo del conocimiento del jacobiano, el hessiano de la función de error se puede expresar como:

$$\nabla^2 f_{err}(\boldsymbol{x},\boldsymbol{\mathcal{P}}) = 2\,\boldsymbol{\mathcal{J}}^T(\boldsymbol{x},\boldsymbol{\mathcal{P}})\,\boldsymbol{\mathcal{J}}(\boldsymbol{x},\boldsymbol{\mathcal{P}}) + 2\sum_{i=1}^{N} \vartheta_i(\boldsymbol{x},\boldsymbol{\mathcal{P}})\,\nabla^2\vartheta_i(\boldsymbol{x},\boldsymbol{\mathcal{P}}) \tag{3.13}$$

Si el segundo término es lo suficientemente pequeño, el hessiano quedaría como:

$$\nabla^2 f_{err}(\boldsymbol{x},\boldsymbol{\mathcal{P}}) = 2\,\boldsymbol{\mathcal{J}}^T(\boldsymbol{x},\boldsymbol{\mathcal{P}})\,\boldsymbol{\mathcal{J}}(\boldsymbol{x},\boldsymbol{\mathcal{P}}) \tag{3.14}$$

Si se sustituyen las ecuaciones (3.12) y (3.14) en la fórmula (3.9), que permitía iterar buscando el mínimo en el método de Newton, nos deriva al algoritmo de minimización denominado de Gauss-Newton:

$$\boldsymbol{\mathcal{P}}^{(i+1)} = \boldsymbol{\mathcal{P}}^{(i)} - \left[\boldsymbol{\mathcal{J}}^T\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)}\right) \boldsymbol{\mathcal{J}}\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)}\right)\right]^{-1} \boldsymbol{\mathcal{J}}^T\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)}\right) \boldsymbol{\vartheta}\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)}\right) \qquad (3.15)$$

Un problema habitual con el algoritmo de Gauss-Newton es la invertibilidad de la matriz $\left[\boldsymbol{\mathcal{J}}^T\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)}\right) \boldsymbol{\mathcal{J}}\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)}\right)\right]$; para facilitar esta operación se puede incluir un término de regularización[5]:

$$\left[\boldsymbol{\mathcal{J}}^T\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)}\right) \boldsymbol{\mathcal{J}}\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)}\right)\right] \rightarrow \left[\boldsymbol{\mathcal{J}}^T\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)}\right) \boldsymbol{\mathcal{J}}\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)}\right) + \mu \boldsymbol{I}\right] \qquad (3.16)$$

resultando:

$$\boldsymbol{\mathcal{P}}^{(i+1)} = \boldsymbol{\mathcal{P}}^{(i)} - \left[\boldsymbol{\mathcal{J}}^T\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)}\right) \boldsymbol{\mathcal{J}}\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)}\right) + \mu \boldsymbol{I}\right]^{-1} \boldsymbol{\mathcal{J}}^T\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)}\right) \boldsymbol{\vartheta}\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)}\right) \qquad (3.17)$$

Del análisis de esta última fórmula nace el método de Levenberg-Marquardt: si $\mu \boldsymbol{I}$ es lo suficientemente pequeño, la anterior ecuación realizará iteraciones de tipo Gauss-Newton; mientras que si dicho término predomina en la matriz a invertir, las iteraciones serán de tipo gradiente descendente.

Los autores propusieron comenzar con un pequeño valor de μ y realizar una iteración. Si el valor de la función de error con el nuevo conjunto de parámetros es menor que el anterior se dividirá μ entre un cierto factor mayor que la unidad (p.ej. 10) y se procederá a realizar otra iteración. Si por el contrario el valor de la función de error hubiese aumentado, se descartará esta iteración, se multiplicará el valor de μ por ese mismo factor y se procederá con una nueva iteración. Así hasta lograr la convergencia en un mínimo.

El método de Levenberg–Marquardt implementa un interesante compromiso entre la velocidad de convergencia de un método de tipo Newton y la convergencia lenta, pero segura, de los métodos de tipo gradiente descendente.

El método de minimización de Levenberg–Marquardt se ha utilizado en esta investigación en la primera aproximación del algoritmo BTW (que luego fue descartada).

Conclusión final

El mayor problema que presentan, en general, los algoritmos para la optimización de ecuaciones no lineales es, si la función a minimizar no es convexa, la posibilidad real de que el algoritmo se quede "atrapado" en un mínimo local[6], no llegando a encontrar el mínimo absoluto. Este problema se agudiza según aumenta el número de dimensiones y la "complejidad" de la función.

---

[5] En la literatura científica existen otros métodos de regularización para resolver este problema como pudiera ser: $\mu \,.\, \boldsymbol{diag}\left(\boldsymbol{\mathcal{J}}^T\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)}\right) \boldsymbol{\mathcal{J}}\left(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^{(i)}\right)\right)$.

[6] La técnica de exploración exhaustiva no participa de este inconveniente.

No existe una solución definitiva para este problema. Algunos algoritmos de aprendizaje en redes neuronales artificiales proponen el comenzar la minimización de la función de error con distintos juegos de valores para los parámetros de aprendizaje [38], o emplear una técnica de "enfriamiento simulado" [39] con la que se persigue en una primera fase encontrar una zona donde los valores de la función de error sean lo suficientemente pequeños (próximos al mínimo global), para luego pasar a buscar el valor más pequeño en esa área.

3.1.2 Optimización con restricciones

La optimización con restricciones intenta minimizar una función objetivo sujeta a un conjunto de restricciones en los parámetros.

$$\begin{aligned} &min \quad f(\boldsymbol{x}, \boldsymbol{\mathcal{P}}) \\ &sujeto\ a\text{:} \begin{cases} \boldsymbol{h}(\boldsymbol{\mathcal{P}}) = 0 \\ \boldsymbol{g}(\boldsymbol{\mathcal{P}}) \geq 0 \end{cases} \end{aligned} \tag{3.18}$$

donde $\boldsymbol{h}(\boldsymbol{\mathcal{P}})$ y $\boldsymbol{g}(\boldsymbol{\mathcal{P}})$ son conjuntos de funciones reales $\mathbb{R}^N \rightarrow \mathbb{R}$ y $\boldsymbol{\mathcal{P}}$ los parámetros de la función $f(\boldsymbol{x}, \boldsymbol{\mathcal{P}})$ a minimizar.

El conjunto de restricciones definen la denominada "región factible" para los parámetros. Dentro de la región factible se encontrará la "solución óptima" ($\boldsymbol{\mathcal{P}}^*$), que será el punto del espacio de los parámetros (obviamente dentro de esa región) donde la función $f(\boldsymbol{x}, \boldsymbol{\mathcal{P}})$ presenta el mínimo absoluto.

Las condiciones de Karush-Kuhn-Tucker (KKT) proporcionan las condiciones de primer orden necesarias[7] para que un punto $\boldsymbol{\mathcal{P}}^*$ sea la solución óptima de un problema de minimización de una función sujeta a un conjunto de restricciones de tipo ecuación y/o inecuación (de acuerdo a lo expresado en (3.18)).

Partiendo de las técnicas de minimización basadas en los multiplicadores de Lagrange (válidas para la optimización cuando las restricciones son de tipo ecuación), el teorema de Karush-Kuhn-Tucker propone que, siendo $\boldsymbol{\mathcal{P}}^*$ el conjunto de parámetros que minimizan $f(\boldsymbol{x}, \boldsymbol{\mathcal{P}})$, y $\boldsymbol{h}(\boldsymbol{\mathcal{P}}) = 0$ y $\boldsymbol{g}(\boldsymbol{\mathcal{P}}) \geq 0$ los conjuntos de funciones que juegan el papel de restricciones en dicha minimización, existen unos vectores $\boldsymbol{\lambda}^*$ y $\boldsymbol{\mu}^*$ tales que:

$$\begin{aligned} &1) \quad \boldsymbol{\mu}^* \geq \mathbf{0} \\ &2) \quad \boldsymbol{\nabla} f(\boldsymbol{x}, \boldsymbol{\mathcal{P}}^*) + \boldsymbol{\lambda}^{*T} \boldsymbol{\nabla} \boldsymbol{h}(\boldsymbol{\mathcal{P}}^*) - \boldsymbol{\mu}^{*T} \boldsymbol{\nabla} \boldsymbol{g}(\boldsymbol{\mathcal{P}}^*) = \mathbf{0}^{\boldsymbol{T}} \\ &3) \quad \boldsymbol{\mu}^{*T} \boldsymbol{g}(\boldsymbol{\mathcal{P}}^*) = 0 \end{aligned} \tag{3.19}$$

Y si a esto le añadimos las propias restricciones de partida:

$$\begin{aligned} &4) \quad \boldsymbol{h}(\boldsymbol{\mathcal{P}}) = 0 \\ &5) \quad \boldsymbol{g}(\boldsymbol{\mathcal{P}}) \geq 0 \end{aligned} \tag{3.20}$$

[7] La suficiencia vienen dada por condiciones de segundo orden.

el conjunto de las cinco expresiones forman las denominadas "condiciones KKT". Aplicándolas al problema en cuestión proporcionan un conjunto de ecuaciones que, una vez resueltas (normalmente mediante métodos iterativos), conducen a encontrar el conjunto de parámetros óptimos.

Las condiciones de KKT se utilizan en la exposición de las máquinas de vectores de soporte (ver sección 4.2.3).

3.1.3 Optimización con restricciones lineales

Particularizando el problema (3.18), si las restricciones en los parámetros son ecuaciones e inecuaciones lineales (situación habitual), la ecuación (3.18) quedará:

$$\begin{aligned} &min \;\; f(\boldsymbol{x}, \boldsymbol{\mathcal{P}}) \\ &sujeto\ a\text{:} \begin{cases} \boldsymbol{A} \,.\, \boldsymbol{\mathcal{P}} = \boldsymbol{a} \\ \boldsymbol{B} \,.\, \boldsymbol{\mathcal{P}} \geq \boldsymbol{b} \end{cases} \end{aligned} \tag{3.21}$$

donde las matrices $\boldsymbol{A}$ y $\boldsymbol{B}$ no tienen por qué ser cuadradas (habitualmente tienen más columnas que filas)[8]. En este caso la región factible tendrá forma poliédrica.

Dentro de este tipo de optimización con restricciones lineales se revelan como más importantes en esta tesis la programación lineal y la programación cuadrática[9].

3.1.3.1 Programación lineal

En los problemas de programación lineal ("Linear Programming" o LP) la expresión de la función a minimizar es el producto escalar de un vector de "costo" $\boldsymbol{c}$ por el vector que contiene los parámetros $\boldsymbol{\mathcal{P}}$.

$$\begin{aligned} &min \;\; \boldsymbol{c}^T \,.\, \boldsymbol{\mathcal{P}} \\ &sujeto\ a\text{:} \begin{cases} \boldsymbol{A} \,.\, \boldsymbol{\mathcal{P}} = \boldsymbol{a} \\ \boldsymbol{\mathcal{P}} \geq \boldsymbol{0} \end{cases} \end{aligned} \tag{3.22}$$

---

[8] Y en muchos casos $\boldsymbol{B}$ es una matriz unitaria, y $\boldsymbol{b}$ es el vector $\boldsymbol{0}$. En cualquier caso, en la mayoría de tratados, y mediante el uso de variables de holgura, las restricciones de tipo inecuación, recogidas en (3.21), se convierten en ecuaciones e inecuaciones simples en los parámetros. Así pues, se pueden expresar en la forma $\begin{cases} \boldsymbol{E} \,.\, \boldsymbol{\mathcal{P}} = \boldsymbol{e} \\ \boldsymbol{\mathcal{P}} \geq \boldsymbol{0} \end{cases}$.

[9] Existen más variantes, como la "programación lineal con valores enteros", la "programación no lineal"…, pero no se van a explicar porque se salen de los métodos necesarios para abordar esta tesis.

Existen métodos muy efectivos para solucionar problemas que requieran este tipo de optimización:

- El tradicional método simplex. Realiza iteraciones moviéndose de un vértice de la región factible a otro, buscando aquel en que la función objetivo presente un menor valor.
- El conocido como "método del punto interior". En este método, el punto óptimo se comienza a buscar desde dentro de la región factible. A partir de ahí, la siguiente iteración mueve dicho punto en la dirección negativa del vector de coste proyectada sobre el espacio nulo de la matriz de restricciones lineales; hasta llegar a las proximidades de una de las restricciones[10]. Se procede a realizar una transformación matricial mediante la cual el punto de partida se convierte en el "centro" de la región y, partiendo de este nuevo punto se procede con la siguiente iteración.

Para más información sobre estos métodos se puede consultar un texto de referencia como [40].

Formas primal y dual

Muchos problemas de optimización se pueden expresar (y resolver) en forma primal o dual. La programación lineal expresada en forma primal es la recogida en las ecuaciones (3.22). Para obtener la forma dual, en primer lugar se forma el Lagrangiano:

$$\mathcal{L}(\boldsymbol{\mathcal{P}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \boldsymbol{c}^T . \boldsymbol{\mathcal{P}} + \boldsymbol{\lambda}^T(\mathbf{a} - \boldsymbol{A} . \boldsymbol{\mathcal{P}}) \tag{3.23}$$

Para obtener el mínimo de $\boldsymbol{c}^T . \boldsymbol{\mathcal{P}}$ se debe obtener un punto de silla ("saddle point") de este Lagrangiano, es decir, se debe minimizar respecto a las variables de la forma primal $\boldsymbol{\mathcal{P}}$ y maximizar respecto a los multiplicadores de Lagrange $\boldsymbol{\lambda}$.

Operando:

$$\begin{aligned} \max_{\boldsymbol{\lambda}} \min_{\boldsymbol{\mathcal{P}}} \ \boldsymbol{c}^T . \boldsymbol{\mathcal{P}} + \boldsymbol{\lambda}^T(\mathbf{a} - \boldsymbol{A} . \boldsymbol{\mathcal{P}}) &= \max_{\boldsymbol{\lambda}} \min_{\boldsymbol{\mathcal{P}}} \ \boldsymbol{\lambda}^T\mathbf{a} + (\boldsymbol{c}^T - \boldsymbol{\lambda}^T\boldsymbol{A}) . \boldsymbol{\mathcal{P}} = \\ &= \max_{\boldsymbol{\lambda}} \left[\boldsymbol{\lambda}^T\mathbf{a} + \min_{\boldsymbol{\mathcal{P}}} (\boldsymbol{c}^T - \boldsymbol{\lambda}^T\boldsymbol{A}) . \boldsymbol{\mathcal{P}} \right] \end{aligned} \tag{3.24}$$

donde:

$$\min_{\boldsymbol{\mathcal{P}}} (\boldsymbol{c}^T - \boldsymbol{\lambda}^T\boldsymbol{A}) . \boldsymbol{\mathcal{P}} \ = \ \left\{ \begin{array}{ll} 0 & \text{si } \boldsymbol{c}^T - \boldsymbol{\lambda}^T\boldsymbol{A} \geq \mathbf{0}^T \\ -\infty & \text{en caso contrario.} \end{array} \right\} \tag{3.25}$$

Así pues, y descartando como solución que $\boldsymbol{\lambda}$ pueda adoptar aquellos valores que hacen la anterior expresión $-\infty$, resulta que el problema dual consistirá

[10] Este paso es similar a un movimiento en la dirección del gradiente descendente.

en maximizar la nueva función objetivo (3.24), respecto a las variables duales $\boldsymbol{\lambda}$, cumpliendo la restricción de la expresión (3.25).

$$\begin{aligned} &\max_{\boldsymbol{\lambda}} \quad \boldsymbol{\lambda}^T.\boldsymbol{a} \quad \equiv \max_{\boldsymbol{\lambda}} \quad \boldsymbol{a}^T.\boldsymbol{\lambda} \\ &sujeto\ a\text{:} \quad \boldsymbol{\lambda}^T.\boldsymbol{A} \leq \boldsymbol{c}^T \equiv \boldsymbol{A}^T.\boldsymbol{\lambda} \leq \boldsymbol{c} \end{aligned} \tag{3.26}$$

Quedando así el problema expresado en función de los multiplicadores de Lagrange de la forma primal (que ahora serán las variables "duales")[11].

El tratamiento de un problema en la forma primal o dual se utiliza en las explicaciones máquinas de vectores de soporte (ver sección 4.2.3).

3.1.3.2 Programación cuadrática

Un tipo particular de optimización con restricciones lineales es la programación cuadrática ("Quadratic Programming" o QP). En ella la función objetivo es una función de tipo cuadrático, manteniendo las restricciones un formato lineal:

$$\begin{aligned} &min \ \frac{1}{2}\boldsymbol{\mathcal{P}}^{\boldsymbol{T}}\boldsymbol{Q}(\boldsymbol{x})\boldsymbol{\mathcal{P}} + \boldsymbol{c}^T\boldsymbol{\mathcal{P}} \\ &sujeto\ a\text{:} \begin{cases} \boldsymbol{A}\,.\boldsymbol{\mathcal{P}} = \boldsymbol{a} \\ \boldsymbol{B}\,.\boldsymbol{\mathcal{P}} \geq \boldsymbol{b} \end{cases} \end{aligned} \tag{3.27}$$

Si la matriz $\boldsymbol{Q}(\boldsymbol{x})$ es semidefinida positiva en el rango de parámetros a optimizar, se hablará de optimización QP convexa. La optimización QP convexa conduce al único mínimo de la función. Lo cual no deja lugar a quedar atrapado en mínimos locales.

De forma similar a lo expuesto en la programación lineal, aquí también se pueden plantear las formas primal y dual del problema.

Este tipo de optimización se utiliza en las explicaciones de las máquinas de vectores de soporte (ver sección 4.2.3).

[11] En el caso de que las restricciones lineales del problema primal provengan de un sistema de inecuaciones (convertido en ecuaciones mediante variables de holgura) se deberá cumplir también que los elementos de $\boldsymbol{\lambda}$ correspondientes a dichas variables de holgura deben ser no negativos.

3.1.4 Programación semidefinida

Se define el cono de matrices simétricas semidefinidas positivas $\mathcal{S}_+^n$ como:

$$\mathcal{S}_+^n = \{\boldsymbol{\mathcal{P}} \in \mathcal{S}^n \mid \boldsymbol{z}^T \boldsymbol{\mathcal{P}}\, \boldsymbol{z} \geq 0,\ \ \forall\, \boldsymbol{z} \in \mathbb{R}^n\} \tag{3.28}$$

donde $\mathcal{S}^n$ es el conjunto de matrices simétricas de dimensión n x n, y $\boldsymbol{z}$ un vector cualquiera de dimensión n.

Y al tener en cuenta que el producto interno de matrices semidefinidas positivas se calcula mediante la traza del producto de ambas matrices:

$$\langle \boldsymbol{A}, \boldsymbol{B} \rangle := tr\left(\boldsymbol{A}^{\boldsymbol{T}} \boldsymbol{B}\right) = \sum_{i,j} a_{ij}\, b_{ij} \tag{3.29}$$

la programación semidefinida (SDP) intenta minimizar una función lineal definida como el producto interno de una matriz $\boldsymbol{C} \in \mathbb{R}^{nxn}$ por la matriz de los parámetros que son objeto de la optimización. Está minimización está sujeta a restricciones que son también productos internos de matrices $\boldsymbol{A}_i$ y $\boldsymbol{\mathcal{P}}$, y a que la matriz de parámetros se mantenga en el cono de matrices semidefinidas positiva:

$$\begin{aligned} &min\ \langle \boldsymbol{C}, \boldsymbol{\mathcal{P}} \rangle \\ &sujeto\ a\text{:}\ \begin{cases} \langle \boldsymbol{A}_i, \boldsymbol{\mathcal{P}} \rangle = b_i \\ \boldsymbol{\mathcal{P}} \in \mathcal{S}_+^n \end{cases} \end{aligned} \tag{3.30}$$

Puede considerarse como una extensión de la programación lineal, donde las restricciones basadas en productos internos entre vectores son reemplazadas por productos internos entre matrices.

La programación semidefinida (y en general la optimización convexa) han sido activos campos de investigación durante la última década del siglo pasado y en la primera de este. Habiéndose establecido en estas dos décadas las bases teóricas y los algoritmos que permiten resolver problemas prácticos. Este tipo de optimización comienza a ser aplicado en numerosos campos de la ciencia [41]: dimensionamiento de componentes microelectrónicos, diseño de experimentos, optimización para estructuras de edificios, localización de sensores [42], investigación operativa [43], control de equipos industriales [44] [45]…

Dentro del ámbito de esta investigación, la programación semidefinida ha tenido que ser abordada para comprender algunos de los trabajos relacionados con la optimización de "*kernels*" en las SVM y el algoritmo LMNN (ver sección 4.4.5).

3.2 Las métricas riemannianas

Desde que somos niños nos hemos acostumbrado a vivir en un mundo euclídeo; que la distancia más corta entre dos puntos no sea la línea recta no se pone en duda por la mayor parte de la población. Pero a principios del siglo XIX insignes científicos comenzaron a estudiar y cuestionar seriamente los cinco postulados de Euclides:

- Dos puntos determinan un segmento de recta.
- Un segmento de recta se puede extender indefinidamente en forma de línea recta.
- Dados un centro y un radio se puede trazar una circunferencia.
- Todos los ángulos rectos son iguales entre sí.
- Por un punto exterior a una recta se puede trazar solo una paralela.

En 1854, Georg Friedrich Bernhard Riemann dio su famosa conferencia "Über die Hypothesen, welche der Geometrie zu Grunde liegen" [46] en el Coloquio de la Facultad de Filosofía de Göttingen. En ella puso las bases para una generalización de la geometría de las superficies, campo previamente estudiado por otros grandes científicos como Gauss, Bolyai y Lobachevsky.

Aunque en un comienzo se consideró como un conjunto de artificios matemáticos solo al alcance de unos pocos científicos, y sin relevancia práctica alguna, con el devenir del siglo XX se ha convertido en una de las más importantes herramientas matemáticas modernas[12].

Quizás el abandonar la métrica euclídea en favor de una métrica de Riemann sea la mejor manera de formalizar, matemáticamente, un concepto intuitivo. Recuérdese la transformación de la geometría del espacio-tiempo propuesta por Einstein para tratar de que las fórmulas de la física siguiesen siendo simples. En la clasificación de casos por medio de las distancias que los separan se puede intentar una aproximación similar. ¿Por qué no mantener un modelo clasificador sencillo, como el *k*-NN, pero mejorar sus prestaciones modificando la forma en la que se miden las distancias?

En esta sección se van a exponer algunos conceptos básicos de estas métricas riemannianas. Una aproximación detallada a esta rama de las matemáticas se puede realizar mediante la siguiente bibliografía: [47], [48], [49].

[12] En esta sección, se sigue la notación empleada habitualmente en el cálculo tensorial. Los vectores contravariantes se identifican porque el índice se encuentra en la parte superior, y en los covariantes el índice aparece en la parte inferior.

3.2.1 El tensor métrico

Un tensor métrico, en un punto $\boldsymbol{x}$ de una variedad M, es un tipo de función $\boldsymbol{G_x}(\boldsymbol{v},\boldsymbol{w}) = \{g_{ij}\}$, la cual, tomando como argumentos un par de vectores $\boldsymbol{v}$, $\boldsymbol{w} \in T_P M$, pertenecientes al espacio métrico e incluidos en el plano tangente[13] a dicha variedad, devuelve un valor escalar $g_{ij}\, v^i\, w^j$. [14]

Debe cumplir con las siguientes propiedades[15]:

- Ser bilineal (linealmente separable en cada argumento).
- Todas las derivadas parciales de segundo orden de g_{ij} existen y son continuas.
- $\boldsymbol{G}$ es simétrica.
- $\boldsymbol{G}$ es no singular.
- Un elemento diferencial lineal elevado al cuadrado[16] se expresa como:

$$ds^2 = g_{ij}\, dx^i\, dx^j$$

  y debe ser invariante respecto a un cambio de coordenadas.

Un espacio métrico $\boldsymbol{G} = \{g_{ij}\}$, expresado en un sistema coordenado $\boldsymbol{x}^i$ es euclídeo si existe una cierta transformación de coordenadas bajo la cual $\bar{g}_{ij} = \delta_{ij}$ [17]. No hay que confundir esto con que el tensor métrico sea una matriz identidad (o diagonal), o que todos los símbolos de Christoffel sean nulos. Métricas euclídeas expresadas en sistemas de coordenadas no cartesianos (polares, esféricos,...) pueden presentar elementos del tensor métrico que no sean constantes y símbolos de Christoffel no nulos, pero siguen siendo métricas euclídeas.

3.2.2 El tensor métrico definido positivo

Un tensor métrico se dice que es definido positivo en un punto si considerando un vector cualquiera distinto de cero perteneciente al espacio tangente de la variedad en ese punto, el resultado que devuelve el producto interno (ver sección siguiente) de ese vector por sí mismo es siempre mayor que cero. [18]

---

[13] Denominado "espacio tangente".

[14] Para interpretar las expresiones en las que intervienen tensores se sigue la notación de Einstein, donde la presencia de índices repetidos indican la suma de los productos sobre todos los posibles valores de ese índice, es decir: $g_{ij}\, v^i\, w^j = \sum_i \sum_j g_{ij}\, v^i\, w^j$.

[15] Desde un punto de vista matemático, el tensor métrico es un tensor covariante de segundo orden.

[16] También se le conoce como primera forma fundamental.

[17] Otra definición es que un espacio métrico es euclídeo si el tensor de curvatura $\boldsymbol{K}$ es idénticamente nulo en todo el espacio.

[18] Otra forma de expresarlo es decir que $\boldsymbol{G}$ tiene una signatura (N,0).

A partir de este punto de la explicación, y excepto cuando se indique explícitamente lo contrario, se considerará siempre que el tensor métrico que se está empleando es definido positivo.

3.2.3 Producto interno de dos vectores, norma y ángulo entre vectores

El uso de tensores métricos permiten, en cierta forma, generalizar la noción del producto escalar en el espacio euclídeo. Si se supone la existencia de un par de vectores $\boldsymbol{v}, \boldsymbol{w}$ en el espacio tangente a la variedad M, y un tensor métrico $\boldsymbol{G}$, se define el producto interno entre esos dos vectores como:

$$\langle \boldsymbol{v}, \boldsymbol{w} \rangle = g_{ij}\, v^i\, w^j \tag{3.31}$$

De forma similar, y para tensores métricos definido positivos, la norma de un vector se define como:

$$\|\boldsymbol{v}\| = \sqrt{\langle \boldsymbol{v}, \boldsymbol{v} \rangle} \tag{3.32}$$

Y el ángulo entre dos vectores $\boldsymbol{v}$, $\boldsymbol{w}$ como:

$$\cos\theta = \frac{\langle \boldsymbol{v}, \boldsymbol{w} \rangle}{\|\boldsymbol{v}\|\, \|\boldsymbol{w}\|} \qquad 0 \leq \theta \leq \pi \tag{3.33}$$

3.2.4 Distancia entre dos puntos

Apoyándose en la noción del tensor métrico, y por medio de integración, es posible calcular la distancia L entre dos puntos de ese espacio métrico siguiendo una determinada curva paramétrica $\gamma = \gamma(t)$:

$$L = \int_{\gamma} \sqrt{\left| g_{ij} \frac{dx^i}{dt} \frac{dx^j}{dt} \right|}\, dt \tag{3.34}$$

Si el tensor métrico es definido positivo:

- No será necesario explicitar el valor absoluto del argumento del radical que se aprecia en la ecuación (3.34), ya que para todo vector distinto del nulo su producto interno es positivo.
- No existirán arcos (segmentos) de esta curva cuya longitud sea 0 (las denominadas “null curves”).

3.2.5 Elemento de volumen

El cálculo de un volumen se puede obtener mediante la integración de la raíz cuadrada del determinante del tensor métrico, extendida a todo el recinto.

$$V_R = \int_R \sqrt{|\boldsymbol{G}|}\; dx^1\, dx^2 \ldots dx^N \tag{3.35}$$

En el caso de la norma euclídea el valor de $|\boldsymbol{G}|$ es 1, quizás por esta razón al término $\sqrt{|\boldsymbol{G}|}$ se le denomina factor de magnificación de la métrica.

3.2.6 Métrica riemanniana

Una variedad M dotada de un tensor métrico definido positivo para todos los puntos de dicha variedad se denomina riemanniana.
Así pues, una métrica riemanniana en M nos permite calcular aquellos productos internos que permitirán evaluar las distancias entre dos puntos.

La definición de distancia entre dos puntos en una variedad riemanniana se define como la menor de las longitudes calculadas (mediante la ecuación (3.34)) para cualquiera de las curvas $\gamma(t)$ diferenciables[19] de la variedad que unen dichos puntos.
O sea, la distancia entre dos puntos a y b de una variedad riemanniana será:

$$d(a,b) = \underset{\gamma(t)}{inf}\ L(a,b)$$
$$L(a,b) = \sum_{i=1}^{m-1} \int_{t_i}^{t_{i+1}} \|\dot{\gamma}(t)\|\ dt \qquad a = \gamma(t_1),\ \ b = \gamma(t_m) \tag{3.36}$$

donde $L(a,b)$ es la suma de los arcos infinitesimales para los distintos segmentos diferenciables $[t_i, t_{i+1}]$ que, a lo largo de la curva $\gamma(t)$, unen los puntos a y b.
Esta definición de distancia entre puntos para la métrica riemanniana asegura el cumplimiento del axioma de la desigualdad triangular. La simetría del tensor y la no existencia de "null curves" completan los tres axiomas exigibles a una métrica.

3.2.7 Concepto de geodésica

El espacio métrico de una variedad riemanniana M es completo; por tanto, para cualquier pareja de puntos $a, b \in M$, existen una o más curvas que los unen y, de ellas, una en la cual la longitud de ese segmento es la más pequeña. A esta curva se la conoce como la geodésica que une dichos puntos. [20]

[19] No es necesario que la curva sea diferenciable en todo su recorrido, vale con que lo sea por trozos.

[20] Otra definición de geodésica es: "de todas las curvas que unen dos puntos suficientemente próximos en una variedad Riemanniana, aquella que proporciona la mínima distancia es una geodésica".

Por métodos variacionales[21] se puede obtener la ecuación diferencial que rige a una geodésica.

$$\begin{gathered} \ddot{x}^k + \sum_{i,j} \Gamma_{ij}^k \dot{x}^i \dot{x}^j = 0 \\ x(0) = x_0 \\ \dot{x}|_{x_0} = \boldsymbol{v} \end{gathered} \tag{3.37}$$

donde:

Las derivadas son respecto al parámetro de integración (t).

Γ_{ij}^k Son los símbolos de Christoffel de segunda especie de la métrica.

$$\Gamma_{ij}^k = \frac{1}{2} \sum_{l=1}^{N} g^{kl} \left(\frac{\partial g_{jl}}{\partial x_i} + \frac{\partial g_{li}}{\partial x_j} - \frac{\partial g_{ij}}{\partial x_l} \right) \tag{3.38}$$

donde:

g_{ij} son los elementos del tensor métrico covariante.

g^{ij} son los elementos de tensor contravariante conocido como métrica inversa de $\boldsymbol{G}$. Este tensor se calcula invirtiendo la matriz $\boldsymbol{G}$ (lo cual es siempre posible ya que $\boldsymbol{G}$ es no singular).

Así pues, si para calcular la distancia entre dos puntos en una variedad riemanniana, se emplea una curva geodésica, esta distancia cumplirá con los tres axiomas que caracterizan a una métrica.

[21] Una completa demostración de la derivación de esta fórmula se puede obtener en [153]; tanto por el método de minimización de la longitud de la curva, como por el método de anular la aceleración de un punto con velocidad unitaria en el plano tangente a la superficie.

4 Estado del arte

El propósito del aprendizaje de una métrica es mejorar la solución de un problema específico de clasificación. Y a lo largo de esta tesis el estudio experimental está centrado en problemas de clasificación supervisada y su mejora.

Tal como se ha expuesto anteriormente, los problemas de clasificación se resuelven generalmente en dos etapas:

- Entrenamiento del clasificador.
- Predicción de la etiqueta de la clase de un caso nuevo a clasificar.

La etapa de entrenamiento consiste en desarrollar, a partir de los datos empíricos, una función de decisión basándose en la similitud y separación de con casos de la misma y distintas clases respectivamente. Dicha función es el resultado del modelo clasificador, y puede ser lineal o no lineal.

Una métrica es una función de distancia (o abusando de la notación, de similitud), que cumple los axiomas indicados en la sección 4.1.3.1, y que se usa para medir la separación o similitud de dos casos. El aprendizaje de una métrica consiste en desarrollar un método para parametrizar la función de distancia o similitud considerada.

El objetivo de este amplio capítulo es realizar un estudio del arte sobre el aprendizaje de estrategias para la clasificación supervisada basada en métricas.

Para una mejor comprensión, este capítulo se ha dividido en cinco secciones. En la primera se estudian las medidas de separación y similitud, presentando las familias de funciones de distancia que se usan en el mundo de la clasificación. En esta primera sección se recoge también las definiciones de los espacios métricos, normados y de productos internos. Su necesidad se justifica por el hecho de que los casos se representan mediante vectores de atributos.

En la segunda sección se estudian brevemente los algoritmos clásicos de clasificación *k*-NN, LDA y las máquinas de vectores de soporte (SVM) porque los algoritmos de aprendizaje de modelos para la clasificación se desarrollan para mejorar alguna de las características de los clasificadores *k*-NN y SVM, y la lógica del LDA se utilizará como mecanismo de orientación de la métrica BTW.

En la tercera sección se presentan los enfoques del estado del arte de aprendizaje para los modelos de clasificación. En la cuarta sección se estudiarán detalladamente los algoritmos de aprendizaje de referencia para esta tesis y que han servido de base para la concepción de la métrica global BTW y la local LOM.

Por último, se proporcionará una aportación novedosa de esta tesis sobre la magnificación de los elementos diferenciales de volumen y su influencia en la mejora de la separabilidad de los casos.

4.1 Medidas de separación y de similitud

En multitud de aplicaciones, especialmente en el campo del aprendizaje automático ("machine learning", ML), dado un conjunto de objetos es muy importante poder medir la separación y similitud de cualquier pareja de sus elementos. Intuitivamente dos objetos estarán más separados cuanto mayor sea la "diferencia" entre los dos. Ambos conceptos, separación y similitud se pueden medir aplicando una función de distancia. Desde la perspectiva de procedimiento matemático, interesa más que la función de distancia elegida sea una métrica.

Antes de exponer las propiedades de las principales medidas de separación, como son las métricas, es conveniente reflexionar sobre la forma de representar los atributos que conforman uno de estos objetos (un caso empírico). En la mayor parte de las representaciones matemáticas, los valores de los atributos se almacenan en vectores de números reales, y esto lleva a tener que definir en primer lugar qué es un espacio vectorial[1].

4.1.1 Espacio vectorial

Un conjunto $\mathcal{V}$ es un espacio vectorial sobre $\mathbb{R}$ si sobre él se han definido las operaciones de adicción (+) y multiplicación por un escalar (.), tal que para dos elementos del espacio vectorial $\boldsymbol{v}_i$, $\boldsymbol{v}_j \in \mathcal{V}$, y otros dos valores reales c , $d \in \mathbb{R}$ se cumple:

- $\boldsymbol{v}_i + \boldsymbol{v}_j \in \mathcal{V}$
- $c \,.\, \boldsymbol{v}_i \in \mathcal{V}$
- $c \,.\, (d \,.\, \boldsymbol{v}_i) = (c \,.\, d) \,.\, \boldsymbol{v}_i$
- $\mathbf{1} \,.\, \boldsymbol{v}_i = \boldsymbol{v}_i$
- $\mathbf{0} \,.\, \boldsymbol{v}_i = \mathbf{0}$

Además, la operación de suma tiene que satisfacer[2] para todo $\boldsymbol{v}_i, \boldsymbol{v}_j, \boldsymbol{v}_k \in \mathcal{V}$:

- $\boldsymbol{v}_i + \boldsymbol{v}_j = \boldsymbol{v}_j + \boldsymbol{v}_i$
- $(\boldsymbol{v}_i + \boldsymbol{v}_j) + \boldsymbol{v}_k = \boldsymbol{v}_i + (\boldsymbol{v}_j + \boldsymbol{v}_k)$
- $\exists\ \mathbf{0} \in \mathcal{V} : \quad \boldsymbol{v}_i + \mathbf{0} = \boldsymbol{v}_i$
- $\exists\ -\boldsymbol{v}_i \in \mathcal{V} : \quad \boldsymbol{v}_i + (-\boldsymbol{v}_i) = \mathbf{0}$

Y también tiene que cumplir las dos propiedades distributivas:

- $c \,.\, (\boldsymbol{v}_i + \boldsymbol{v}_j) = \ c \,.\, \boldsymbol{v}_i + c \,.\, \boldsymbol{v}_j$
- $(c + d) \,.\, \boldsymbol{v}_i = \ c \,.\, \boldsymbol{v}_i + d \,.\, \boldsymbol{v}_i$

---

[1] Una métrica no necesita obligatoriamente estar definida sobre un espacio vectorial (solo lo tiene que estar sobre un conjunto de puntos), pero para los desarrollos de esta tesis es preferible asociarla a un espacio de vectores.

[2] Estas cuatro condiciones lo definen además como grupo conmutativo.

4.1.2 Función de distancia y función de similitud

4.1.2.1 Función y medida de distancia[3]

Dado un espacio vectorial $\mathcal{V}$ y sea $\mathbb{R}^+$ el conjunto de todos los números no negativos, una función $d : \mathcal{V} \longrightarrow \mathbb{R}^+$ es una función de distancia si verifica los siguientes axiomas:

- $d(\boldsymbol{v}) = 0$ si $\boldsymbol{v} = \boldsymbol{0}$ (reflexiva).
- $d(\boldsymbol{v}) = d(-\boldsymbol{v})$ (simetría).

En las aplicaciones de clasificación, este vector estará definido por la diferencia entre dos objetos (casos) los cuales estarán también representados como vectores en el espacio de atributos: $\boldsymbol{x}_i, \boldsymbol{x}_j$. Por lo tanto se hablará de distancia $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

4.1.2.2 Función y medida de similitud

Las distintas funciones de distancia, en cualquiera de sus variedades, evalúan la disimilitud entre casos. En algunos escenarios es preferible expresar esa relación como una medida de similitud.

Una función de similitud es una función que se aplica sobre dos objetos y que presenta los siguientes axiomas:

- $sim(\boldsymbol{x}_i, \boldsymbol{x}_i) = 1$ (reflexiva).
- $sim(\boldsymbol{x}_i, \boldsymbol{x}_j) = sim(\boldsymbol{x}_j, \boldsymbol{x}_i)$ (simetría).

Basándose solo en esas propiedades, a una función de similitud se le puede asociar una "función de distancia", la cual es una medida que toma valores no negativos.

Se dice que una medida de similitud y una de distancia se corresponden si puede definirse una función *f* que relacione ambas [50] [51]:

$$rango(d) \xrightarrow{f} rango(sim)$$

Esto es:

$$sim(\boldsymbol{x}_i, \boldsymbol{x}_j) = f\left(d(\boldsymbol{x}_i, \boldsymbol{x}_j)\right)$$

Aparte de la inversa de la distancia[4], algunas funciones monótonas decrecientes que pueden servir para implementar esta relación son:

$$f(x) = 1 - \frac{x}{1+x}\,; \qquad f(x) = 1 - \frac{x}{\max(rango\ d)}; \qquad f(x) = e^{-cx} \tag{4.1}$$

En la sección 5.2.4.2 se explicará el empleo de la tercera de estas fórmulas en el algoritmo BTW.

---

[3] No hay un consenso en la comunidad científica sobre la definición de "función de distancia" o "medida de distancia".

[4] Que presenta el problema de que para una distancia igual a cero da una similitud de valor infinito, lo cual puede causar problemas en muchos programas informáticos.

4.1.3 Espacios métricos, espacios normados y de productos internos

Sobre un espacio vectorial es posible definir una métrica, obteniéndose así un espacio métrico.

4.1.3.1 Espacio métrico

Un espacio métrico se define como la pareja $(\mathcal{X},\ d)$ donde d es una función de medida de distancias $d\colon \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}^+$ que tiene que cumplir:

- $d(\boldsymbol{x}_i,\ \boldsymbol{x}_j) \geq 0$ (no negatividad).
- $d(\boldsymbol{x}_i,\ \boldsymbol{x}_j) = d(\boldsymbol{x}_j,\ \boldsymbol{x}_i)$ (simetría).
- $\forall\, \boldsymbol{x}_i = \boldsymbol{x}_j \ \leftrightarrow\ d(\boldsymbol{x}_i,\ \boldsymbol{x}_j) = 0$ (identidad).
- $d(\boldsymbol{x}_i,\ \boldsymbol{x}_j) \leq d(\boldsymbol{x}_i,\ \boldsymbol{x}_k) + d(\boldsymbol{x}_k,\ \boldsymbol{x}_j)$ (desigualdad triangular).

La desigualdad triangular expresa que la distancia desde *x* hasta *z* es más corta que la suma de las distancias a través de otro punto *y*.

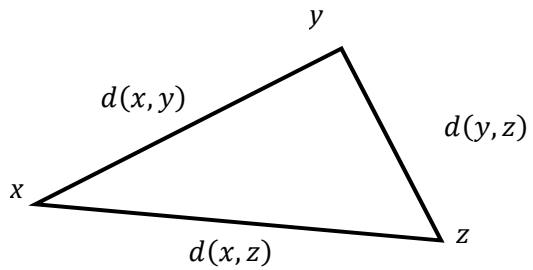


Figura 4.1.- Representación de la desigualdad triangular.

En algunos libros, la tercera de las anteriores condiciones se desdobla en dos:

- $d(\boldsymbol{x}_i,\ \boldsymbol{x}_i) = 0$ (reflexiva).
- $\forall\, \boldsymbol{x}_i \neq \boldsymbol{x}_j \ \leftrightarrow\ d(\boldsymbol{x}_i,\ \boldsymbol{x}_j) > 0$ (positividad).

Existen otras medidas de distancias que no cumplen alguno de los postulados anteriores[5]:

- Si no se satisface la condición de "positividad" a la medida de distancia se la llama pseudométrica.
- Si lo que no se satisface es la "simetría" se llama cuasimétrica.
- Si carece de la propiedad de "desigualdad triangular" se la denominará semimétrica.

Existen múltiples estudios que sugieren que medidas de distancias que no cumplen todas las propiedades de las métricas proporcionan un rendimiento

[5] Aparte de las citadas en estos párrafos, en función de los axiomas que cumplan también existen otras medidas de distancia, pero raramente se utilizan en los textos: hemimétricas, premétricas,... Además, sobre la notación para las distancias que no cumplen todos los axiomas de una métrica no hay un consenso universal.

comparable (e incluso superior en algunos casos) al ser usadas en los algoritmos habituales de la inteligencia artificial [52].

También hay autores que a las propiedades de una métrica le añaden una condición más (que es más restrictiva que la "desigualdad triangular") y se denomina "desigualdad triangular reforzada":

- $d(\boldsymbol{x}_i,\ \boldsymbol{x}_j) \leq max\{d(\boldsymbol{x}_i,\ \boldsymbol{x}_k),\ d(\boldsymbol{x}_k,\ \boldsymbol{x}_j)\}$

A estas distancias se les llaman supermétricas o ultramétricas[6].

4.1.3.2 Espacios normados

Dado un espacio vectorial $\mathcal{X}$, se denomina norma a una función $\|\,.\,\| : \mathcal{X} \longrightarrow \mathbb{R}^+$ que verifica las siguientes condiciones:

- $\|\boldsymbol{x}_i\| \geq 0$ (no negatividad).
- $\|\boldsymbol{x}_i\| = 0 \Leftrightarrow \boldsymbol{x}_i = \boldsymbol{0}$ (no degeneración).
- $\|c\ .\ \boldsymbol{x}_i\| = |c|\ .\ \|\boldsymbol{x}_i\|\ ,\ \forall\, c \in \mathbb{R}$ (homogeneidad).
- $\|\boldsymbol{x}_i + \boldsymbol{x}_j\| \leq \|\boldsymbol{x}_i\| + \|\boldsymbol{x}_j\|$ (desigualdad triangular).

Si $\|\,.\,\|$ es una norma, el espacio $\mathcal{X}$ se denomina espacio normado $(\mathcal{X}, \|\,.\,\|)$.

Dado un espacio normado $(\mathcal{X}, \|\,.\,\|)$, $\mathcal{X}$ es también un espacio métrico cuya medida de distancia es $d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|$.

A este tipo de métricas, basadas en la diferencia entre vectores, se las conoce habitualmente como "distancias normadas" o "métricas homogéneas e invariantes en una translación".

4.1.3.3 Espacios de productos internos

Producto interno

Sea $\mathcal{X}$ un espacio vectorial, se llama producto interno $\langle\cdot\,,\cdot\rangle$ a una aplicación bilinear $\mathcal{X}\,\mathrm{x}\,\mathcal{X} \to \mathbb{R}$, que para $\boldsymbol{x},\ \boldsymbol{y},\ \boldsymbol{z} \in \mathcal{X}$ y $\alpha,\ \beta \in \mathbb{R}$ cumple con los siguientes axiomas:

- $\langle \boldsymbol{x},\ \boldsymbol{y}\rangle \geq 0$ (no negatividad).
- $\langle \boldsymbol{x},\ \boldsymbol{y}\rangle = \langle \boldsymbol{x},\ \boldsymbol{y}\rangle$ (simetría).
- $\forall\, \boldsymbol{x} = \boldsymbol{0} \ \leftrightarrow\ \langle \boldsymbol{x},\ \boldsymbol{x}\rangle = 0$ (no degeneración).
- $\langle \alpha\boldsymbol{x} + \beta\boldsymbol{y},\ \boldsymbol{z}\rangle = \alpha\,\langle \boldsymbol{x},\ \mathbf{z}\rangle + \beta\,\langle \boldsymbol{y},\ \boldsymbol{z}\rangle$ (linealidad).

Si $\langle\cdot\,,\cdot\rangle$ es un producto interno, $(\mathcal{X},\ \langle\cdot\,,\cdot\rangle)$ se denomina espacio de producto interno. Todo espacio de producto interno es un espacio normado con la norma L_2: $\|\boldsymbol{x}\|^2 = \langle \boldsymbol{x}, \boldsymbol{x}\rangle$ y en este espacio se verifican:

- Regla del paralelogramo: $\|\boldsymbol{x} + \boldsymbol{y}\|^2 + \|\boldsymbol{x} - \boldsymbol{y}\|^2 = 2\|\boldsymbol{x}\|^2 + 2\|\boldsymbol{y}\|^2$ para L_2
- Identidad de polarización: $\langle \boldsymbol{x}, \boldsymbol{y}\rangle = \frac{1}{2}\|\boldsymbol{x} + \boldsymbol{y}\|^2 - \|\boldsymbol{x}\|^2 - \|\boldsymbol{y}\|^2$

---

[6] Su uso está relacionado con estudios de biología evolutiva (comparación de ADN entre especies…).

Espacio de Hilbert y espacio pre-Hilbert

Se dice que un espacio de producto interno es un espacio de Hilbert cuando la norma inducida conduce a un espacio métrico completo[7]; en caso contrario se dice que es un espacio pre-Hilbert (espacio de producto interno no completo).

Algunos ejemplos de espacios de Hilbert:

- $(\mathbb{R}^A, \langle \cdot, \cdot \rangle)$ con $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \sum_{i=1}^{A} x_i \, y_i$.
- L_2^{∞} con $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \sum_{i=1}^{\infty} x_i \, y_i$.
  La métrica será $d(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\| = \sqrt{\sum_{i=1}^{\infty} (x_i - y_i)^2}$

Espacio de Hilbert del "*kernel*" de reproducción

Son espacios de Hilbert asociados con una función de tipo "*kernel*" que reproduce todas las funciones de ese espacio, o equivalentemente, donde cada funcional de evaluación es acotado.

Si $\mathcal{X}$ es un conjunto arbitrario y $\mathcal{H}$ un espacio de Hilbert de funciones reales sobre $\mathcal{X}$. El funcional de evaluación sobre el espacio de funciones de Hilbert L_x es un funcional lineal que evalúa cada función en un punto $x \in \mathcal{X}$. $L_x: f \mapsto f(x) \;\; \forall \; f \in \mathcal{H}$.

Se dice que $\mathcal{H}$ es un espacio de Hilbert del "*kernel*" de reproducción si L_x es una función continua para cualquier función f de $\mathcal{H}$, y existe una $M \in \mathbb{R}^+$ tal que $L_x[f] = f(x) \leq M \|f\|_{\mathcal{H}} \;\; \forall \; f \in \mathcal{H}$.

Los "*kernels*" de reproducción (abreviados como simplemente "*kernels*") se utilizan en la teoría de aprendizaje para construir las máquinas de vectores de soporte.

4.1.4 Algunas funciones de distancia y de similitud

En la literatura científica existe una gran variedad de distancias, hecho confirmado por recopilatorios como [53], que se actualiza según los avances científicos y aplicaciones tecnológicas.

Las distancias se pueden agrupar según varios criterios, algunos de ellos son:

- Áreas de aplicación. Existen por ejemplo distancias específicas en las ciencias sociales, naturales, visión artificial, teoría de información, estadística, etc.
- El tipo de los atributos de los casos en el aprendizaje automático. Así se disponen de distancias específicas para variables binarias, cuantitativas, cadenas de texto, etc. [54].

[7] Se dice que un espacio es completo si toda secuencia de Cauchy definida en él converge en un elemento de dicho espacio. Se dice que una secuencia es de tipo Cauchy si $\exists \, K, \;\; \forall \, i, j > K \;\;\; \|\boldsymbol{x}_i - \boldsymbol{x}_j\| < \epsilon$.

- El punto de vista sistemático. Desde esta perspectiva, las distancias se suelen agrupar por familias definidas por las formas de las expresiones. Por su relación con la tesis se destacan las siguientes: las inducidas por la norma L_p: L_2 y L_1, las distancias de tipo Mahalanobis, las distancias derivadas del producto interno y las de la familia χ^2.
- Desde el punto de vista de rango de su aplicabilidad a lo largo del espacio de atributos. Bajo este criterio se pueden clasificar en globales y locales.

4.1.4.1 *Distancias inducidas por la norma L_p (de Minkowsky)*

La más conocida y habitual de las métricas, con diferencia, es la métrica euclídea (L_2); pero a esta distancia se la puede englobar en un grupo mucho más amplio conocido como las distancias de norma L_p o de Minkowsky cuya expresión general responde a:

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(\sum_{a=1}^{A} |x_{i,a} - x_{j,a}|^p \right)^{1/p} \tag{4.2}$$

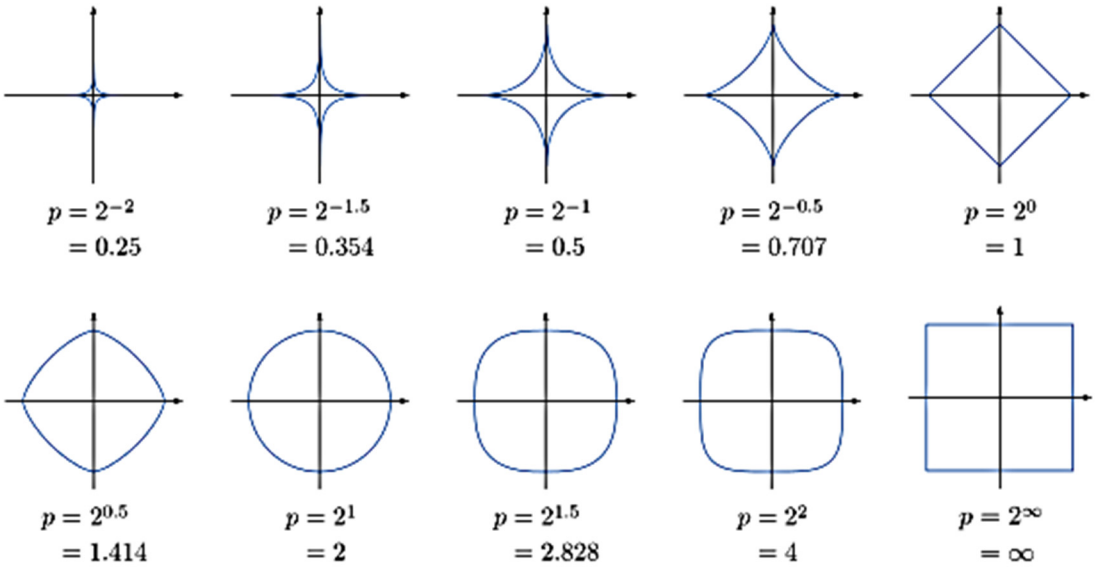


Figura 4.2.- Conjuntos de puntos que distan una unidad desde un punto central de acuerdo a distintas distancias de Minkowski (fuente: [16]).

Generalmente se las conoce como distancias de Minkowski, aunque estrictamente solo son métricas las que verifican que $p \geq 1$, ya que el resto no cumplen con la propiedad de la desigualdad triangular; y serán, por tanto, semimétricas.

Obsérvese la variabilidad existente en los conjuntos de puntos próximos a un punto determinado en función de la elección del parámetro p.[8]

En la siguiente tabla se indican las principales características de algunas de las distancias de Minkowsky más habituales.

Distancias de Minkowsky

p	Expresión	Nombres comunes	Norma	
1	$\sum_{a=1}^{N} \lvert x_{i,a} - x_{j,a} \rvert$	Distancia de Manhattan. Distancia "*City-block*".	L_1	$\lVert \boldsymbol{x}_i - \boldsymbol{x}_j \rVert_1$
2	$\sqrt{\sum_{a=1}^{N} (x_{i,a} - x_{j,a})^2}$	Distancia euclídea. Distancia pitagórica.	L_2	$\lVert \boldsymbol{x}_i - \boldsymbol{x}_j \rVert_2$
∞	$\max_a \lvert x_{i,a} - x_{j,a} \rvert$	Distancia de Chebyshev. Distancia máxima. Distancia infinita. Distancia "chessboard".	L_∞	$\lVert \boldsymbol{x}_i - \boldsymbol{x}_j \rVert_\infty$

Tabla 4.1.- Distancias de Minkowski más habituales.

4.1.4.2 Distancias de tipo Mahalanobis

El otro gran conjunto de distancias son las conocidas como distancias de tipo Mahalanobis, responden a una expresión del tipo:

$$d(\boldsymbol{x}_\mathrm{i}, \boldsymbol{x}_\mathrm{j}) = \left((\boldsymbol{x}_\mathrm{i} - \boldsymbol{x}_\mathrm{j})^T \, \boldsymbol{M} \, (\boldsymbol{x}_\mathrm{i} - \boldsymbol{x}_\mathrm{j}) \right)^{1/2} = \lVert \boldsymbol{x}_\mathrm{i} - \boldsymbol{x}_\mathrm{j} \rVert_M \qquad (4.3)$$

donde $\boldsymbol{M}$ es una matriz de dimensión A x A. En general estas medidas de distancias no serán métricas, excepto si $\boldsymbol{M}$ es una matriz simétrica definida positiva. Los elementos m_{ij} de esta matriz indican la "relevancia de la conexión" entre cada pareja de atributos.

[8] Esta es una prueba más de la importancia de seleccionar la métrica adecuada para un problema práctico (u optimizarla).

En función del tipo de la matriz $\boldsymbol{M}$, se distinguen:

- Si $\boldsymbol{M}$ es la matriz unitaria la anterior métrica se corresponde con la métrica euclídea.
- Si $\boldsymbol{M}$ es una matriz diagonal con todos sus valores no negativos se hablará de una métrica euclídea normalizada (estandarizada o ponderada). Los pesos que se aplican a cada atributo son la raíz cuadrada de su correspondiente valor en la diagonal [55].
- En el caso de que $\boldsymbol{M}$ sea la inversa de la matriz de covarianzas de los datos ($\boldsymbol{\Sigma}$), a la métrica resultante se la denomina propiamente como métrica de Mahalanobis [56].

Una distancia de tipo Mahalanobis difiere de la euclídea en que para esta última todos los atributos de los casos $\boldsymbol{x}_i$ y $\boldsymbol{x}_j$ contribuyen por igual al valor de la distancia, mientras que en las de tipo Mahalanobis los atributos contribuyen de acuerdo a sus términos de ponderación (expresados por medio de los valores de los elementos de la matriz $\boldsymbol{M}$)[9].

Ventajas e inconvenientes

Las distancias de Mahalanobis se usan preferentemente cuando existen importantes diferencias e interacciones cruzadas entre los valores de los atributos y se dispone de información sobre sus correlaciones.

Uno de los objetivos más habituales de este tipo de métrica es evitar que los valores que toman ciertos atributos (o combinaciones de ellos) les hagan preponderar en el cálculo de las distancias. Por ejemplo, si en un problema de clasificación de coches, la longitud de los vehículos se mide en metros, en la base de casos aparecerán valores del orden de 3, 4 o 5 metros; si para los mismos casos se mide esa longitud en milímetros, los valores serán 3.000, 4.000, 5.000,… Es evidente que, en este segundo escenario, cualquier pequeña variación en la longitud de un automóvil hará que el término asociado a su diferencia de longitud con otro se eleve enormemente, y predomine totalmente sobre los términos aportados por los otros atributos en el cómputo total de la distancia.

Aunque las distancias de tipo Mahalanobis tienen cierto predicamento en el mundo de la investigación, ya que poseen ciertas propiedades interesantes (como normalizar los valores en que están medidos los distintos atributos y tener en cuenta sus correlaciones), no es siempre una métrica que esté orientada a mejorar la eficiencia de la clasificación, sobre todo cuando se ha intentado optimizar la matriz completa $\boldsymbol{M}$ sin poner ninguna restricción (ver la teoría de la minimización del riesgo estructural de Vapnik en la sección 2.6.4).

[9] Donde también se pueden incluir las correlaciones entre atributos, que pueden dar lugar a términos aditivos o sustractivos para el total de la distancia.

En el ejemplo de la Figura 4.3 se muestra un problema sintético con dos clases (cuyos casos se distribuyen aproximadamente de acuerdo a distribuciones gaussianas bidimensionales) y en el que la frontera de separación ideal sería una línea paralela al eje de abscisas (línea de puntos). De acuerdo a la lógica que se explicará en las próximas secciones del estudio del estado del arte y en los algoritmos aportados por esta tesis (y que aquí se puede intuir con facilidad), la dirección más relevante (más significativa) para detectar si un caso pertenece a una u otra clase es la vertical (de hecho el atributo 1 no aporta información alguna para la clasificación).

Pues bien, en el gráfico se ve claramente que la métrica de Mahalanobis para este problema, "alarga" en el eje vertical las curvas de los puntos que se encuentran a la misma distancia de uno dado, lo cual hace que esa dirección sea menos significativa y, por tanto, se vaya en contra de lo que indica la lógica de la clasificación.

Se puede apreciar que, aunque el "*kernel*" derivado de esta métrica tiende a alargarse en la dirección de máxima variabilidad del conjunto de datos, esta dirección no se tiene que corresponder con la dirección de la frontera óptima; de hecho, en el ejemplo es justamente la opuesta.

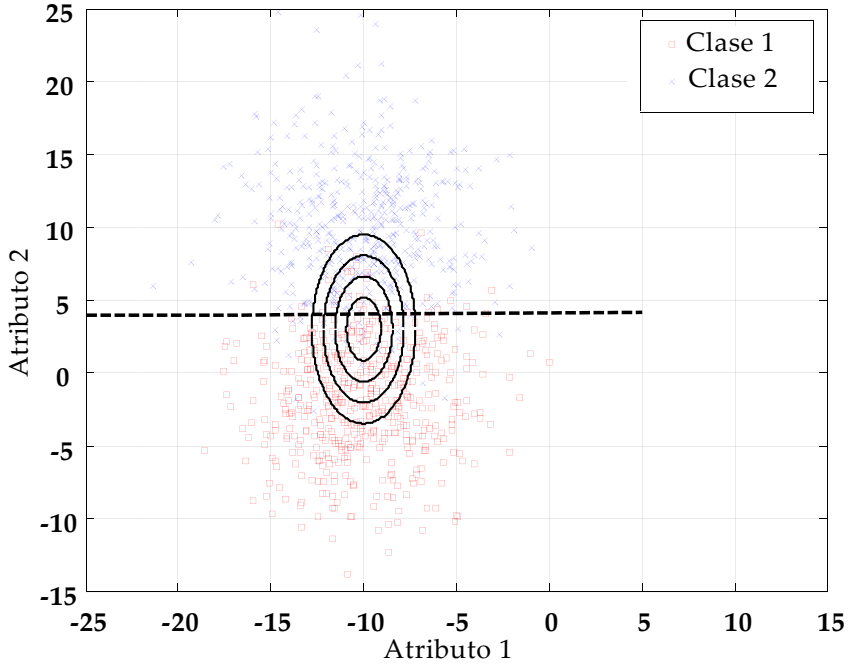


Figura 4.3.- Contraejemplo que ilustra la no idoneidad de la métrica de Mahalanobis para las tareas de clasificación. Se supone que el caso a clasificar está en la posición (-10,2).

En esta línea de argumentación, y como ejemplo a favor de alguna de las distancias de tipo Mahalanobis, se podría citar la distancia euclídea normalizada (ver segundo punto de la enumeración de la página 76), donde los valores de los atributos son divididos entre la desviación estándar para ese atributo. El "*kernel*" resultante de esta métrica estará orientado en la

dirección de los ejes coordenados y alargará sus curvas de equidistancia en función de la mayor varianza de los atributos (lo cual es correcto, pero aun así, y como se comentará más adelante, no es óptimo porque no tiene en cuenta las correlaciones entre los atributos). Una más amplia discusión sobre este tipo de distancia y su optimización se recoge en la sección 4.3.

En general, se discutirá más detalladamente sobre las ventajas e inconvenientes de este tipo de métricas en el estudio de los trabajos de referencia en el estado del arte (sección 4.4)

4.1.4.3 Distancias derivadas de productos internos

Más que distancias son medidas de similitud (aunque también existen sus correspondientes distancias). Las más importantes son: la que se deriva directamente de la definición de producto interno (que es en la que se basan las máquinas de vectores de soporte) y la similitud del coseno.

Medidas de similitud derivadas de productos internos

Nombre	Expresión
Producto interno	$Sim_{pi}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_{a=1}^{N} x_{i,a} \cdot x_{j,a} = \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle$
Coseno	$Sim_{cos}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{\sum_{a=1}^{N} x_{i,a} \cdot x_{j,a}}{\sqrt{\sum_{a=1}^{N} x_{i,a}^2} \sqrt{\sum_{a=1}^{N} x_{j,a}^2}} = \frac{\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle}{\|\boldsymbol{x}_i\| \, \|\boldsymbol{x}_j\|}$

Tabla 4.2.- Similitudes derivadas de productos internos.

La similitud del coseno resulta del cómputo del producto escalar normalizado de dos vectores (formados por los atributos de los casos) dividido entre el producto de sus normas.

Esta medida está basada en el algoritmo para calcular el coseno del ángulo formado por los dos vectores y por lo tanto debe ser interpretada como una medida de "orientación" de dichos vectores.

En algunos textos, $1 - Sim_{cos}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ aplicada en la parte positiva del espacio aparece con el nombre de "Distancia del coseno", pero hay que tener en cuenta que esta no cumple con el axioma de la desigualdad triangular.

Presenta múltiples usos, como herramienta para dividir datos experimentales formando "clusters" [57], minería de textos [58]…

4.1.4.4 Distancias de tipo χ^2

Si los distintos atributos de un problema están orientados a contener datos de recuentos (p.ej. en histogramas, funciones de densidad de probabilidad…), la distancia χ^2 suele ser la métrica a considerar.

Medidas de distancia de tipo χ^2	
Nombre	Expresión
Cuadrado de la euclídea	$d_{sqr}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_{a=1}^{N} (x_{i,a} - x_{j,a})^2$
Pearson χ^2	$d_P(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_{a=1}^{N} \frac{(x_{i,a} - x_{j,a})^2}{x_{j,a}}$
Neyman χ^2	$d_N(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_{a=1}^{N} \frac{(x_{i,a} - x_{j,a})^2}{x_{i,a}}$
χ^2	$d_{\chi^2}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_{a=1}^{N} \frac{(x_{i,a} - x_{j,a})^2}{(x_{i,a} + x_{j,a})}$

Tabla 4.3.- Distancias de tipo χ^2.

La distancia χ^2 se usa en la comparación de pdf en diversos campos científicos como la ecología, medicina [59],... Recientemente ha recibido mayor atención al ser usada en el denominado "kernel χ^2" [60]; también se hace referencia a ella en la métrica DANN de Hastie y Tibshirani [61] o en los trabajos de Domeniconi en su algoritmo ADAMENN [62] y LaMaNNa [63].

4.1.4.5 Medidas de distancia y similitud específicas propuestas en distintas investigaciones

Aparte de las medidas de distancia "básicas" que se reflejan en la sección 4.1.4.1, en esta se presentan aquellas medidas que han resultado especialmente útiles en campos de aplicación específicos. Sin pretender realizar una exposición exhaustiva[10], a continuación se pasará revista a algunas de las más relevantes.

Medidas para datos numéricos:

- Medida de disimilitud de Bray-Curtis.
- Distancia de Canberra.
- Distancia correlación.

Medidas para datos booleanos:

- Distancia de Hamming.
- Contraste de similitud de Tversky.
- Medida de similitud de PATDEX.

[10] Se puede encontrar una relación de los cientos de distancias que han sido definidas en el ámbito de distintas investigaciones en la referencia [53].

- Medidas de similitud de Jaccard y de Rogers-Tanimoto.
- Medida de similitud de Rusell-Rao.
- Índice de Sørensen-Dice.

Medidas para textos:

- Distancia de Hamming.
- Distancia de Damerau-Levenshtein.
- Medida de similitud de Smith-Waterman y de Needleman-Wunsch.

Medida de disimilitud de Bray-Curtis

Es una medida que sirve para medir la diferencia tanto cuantitativa como cualitativa entre los atributos que forman los casos [64]. Su cálculo se basa en sumar en valor absoluto las diferencias entre los distintos atributos y dividirla entre la suma de todos ellos (los de ambos casos):

$$d_{BrCu}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{\sum_a |x_{i,a} - x_{j,a}|}{\sum_a (x_{i,a} + x_{j,a})}$$

El resultado de esta medida varía entre 0 (fuerte relación) y 1 (ausencia de relación).

No es una distancia ya que no cumple con el axioma de la desigualdad triangular.

Tiene una medida de similitud asociada denominada "índice de Bray-Curtis" [65], que se obtiene restando de 1 la anterior medida de disimilitud.

Su uso está extendido en biología para la medida de las relaciones entre poblaciones de distintos nichos ecológicos [66].

Distancia de Canberra

Es una métrica similar a la distancia Manhattan[11] con la modificación de que la diferencia entre atributos es dividida entre la suma (en valor absoluto) de ambos atributos [67]:

$$d_{Can}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_a \frac{|x_{i,a} - x_{j,a}|}{|x_{i,a}| + |x_{j,a}|}$$

Tiene como mérito que es muy sensible[12] para analizar diferencias de atributos cuyos valores son próximos a cero frente a otros atributos con valores más grandes.

Se ha utilizado en informática, entre otras aplicaciones, para comparar listas ordenadas [68] y detección de intrusos en redes [69].

---

[11] En algunos estudios la consideran como una métrica de Manhattan ponderada.

[12] A diferencia de la mayoría de las métricas, esta "amplifica" las diferencias de aquellos atributos cuyo valor es muy pequeño.

Distancia correlación

En general, la correlación es una medida de dependencia entre dos variables aleatorias.

Tal como la concretó Székely [70] en 2007, se define la distancia correlación como el cociente entre la "distancia covarianza" de los dos vectores que representan sendos casos y el producto de las "distancias desviación estándar"[13] de cada uno de esos vectores:

$$d_{Corr}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{d_{Cov}(\boldsymbol{x}_i, \boldsymbol{x}_j)}{\sqrt{d_{Var}(\boldsymbol{x}_i).\, d_{Var}(\boldsymbol{x}_j)}}$$

siendo la distancia covarianza:

$$d_{Cov}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{1}{A^2}\sum_{a=1}^{A}(x_{i,a} - \overline{\boldsymbol{x}}_i).(x_{j,a} - \overline{\boldsymbol{x}}_j)$$

Y la distancia varianza:

$$d_{Var}(\boldsymbol{x}_i) = \frac{1}{A^2}\sum_{a=1}^{A}(x_{i,a} - \overline{\boldsymbol{x}}_i)^2$$

Su objetivo es conseguir que, cuando ambos vectores sean auténticamente independientes, la distancia correlación sea igual a cero, siendo uno cuando sean linealmente dependientes.

Se ha utilizado para medir la asociación de ciertos genes [71] [72].

Distancia de Hamming

Mide el número de sustituciones a realizar para convertir una cadena de códigos en otra [73]. A diferencia de la distancia de Damerau-Levenshtein, las dos cadenas a comparar deben tener la misma longitud. Su uso más habitual es con cadenas de bits, aunque también existen aplicaciones para cadenas de letras.

Se usa en telecomunicaciones para medir el número de permutas que ha sufrido un cierto mensaje al ser transmitido por un canal de comunicación y también en otras ramas de la ciencia como criptografía, genética y, en general, en aquellos problemas relacionados con la teoría de la información.

Contraste de similitud de Tversky

Tversky [74] fue uno de los primeros investigadores que apuntaron que los axiomas que definen una métrica no se reflejan habitualmente en el comportamiento de los seres humanos a la hora de clasificar elementos.

[13] Que a su vez son las raíces cuadradas de las correspondientes "distancias varianza" de cada vector.

El modelo de contraste de Tversky intenta plasmar las anteriores ideas tomando en cuenta y ponderando:

- Las características comunes de dos casos.
- Aquellas características que figuran en el primer caso pero no en el segundo.
- Aquellas que están presentes en el segundo caso pero no en el primero.

La expresión matemática de su definición se similitud responde a:

$$Sim_{Tve}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \alpha \,.\, f(A) - \beta \,.\, f(B) - \gamma \,.\, f(C)$$

donde:

- α, β, γ : son parámetros no negativos a elegir para cada problema.
- A: es el conjunto de atributos que tienen igual valor para $\boldsymbol{x}_i$ y para $\boldsymbol{x}_j$
- B: es el conjunto de atributos que están presentes en $\boldsymbol{x}_i$ y no en $\boldsymbol{x}_j$.
- C: es el conjunto de atributos que están presentes en $\boldsymbol{x}_j$ y no en $\boldsymbol{x}_i$.
- f ( ): son funciones monótonas no decrecientes que se aplican sobre los conjuntos de atributos (en muchos casos consisten simplemente en el cálculo de la cardinalidad de dichos conjuntos).

Su uso fue pionero en estudios psicológicos sobre la percepción y cognición humanas, creando la "teoría de las perspectivas". Más recientemente (en 2002), el colaborador de Tversky, Daniel Kahneman [75] recibió el premio Nobel en Economía por su aplicación en el campo de la toma de decisiones bajo incertidumbre y la percepción del riesgo.

Medida de similitud de PATDEX

PATDEX [76] fue un sistema experto que tuvo un importante predicamento en la última década del pasado siglo. Desarrollado en la Universidad de Kaiserslautern, su rol era sustentar el núcleo de razonamiento de MOLTKE, un entorno de diagnóstico inteligente para máquinas-herramienta y sistemas productivos.

Para calcular la distancia entre dos casos formados por atributos binarios (muy habitual en las máquinas y su automatización) empleaba una versión más avanzada de la medida de similitud de Tversky.

$$d_{Pat}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{\alpha \,.\, |E| + \beta \,.\, |C| + \gamma \,.\, |U| + \eta \,.\, |R|}{\alpha \,.\, |E \cup C \cup U \cup R|}$$

donde:

- $\alpha, \beta, \gamma, \eta$: son parámetros a elegir para cada problema.
- E: es el conjunto de atributos que tienen igual valor para $\boldsymbol{x}_i$ y para $\boldsymbol{x}_j$.
- C: es el conjunto de atributos que tienen distinto valor para $\boldsymbol{x}_i$ y para $\boldsymbol{x}_j$.

- U : es el conjunto de atributos con valor desconocido para el caso a clasificar.
- R : es el conjunto de atributos, para el caso a clasificar, con valor redundante.
- $|\ |$: representa la cardinalidad del conjunto.

Medidas de similitud de Jaccard y de Rogers-Tanimoto

Es una estadística empleada para comparar la similitud de dos conjuntos de datos [77]. En el caso de casos representados mediante atributos binarios, un 1 indicaría que ese "elemento del conjunto" está presente y un 0 que está ausente. En algunos textos también se lo conoce como "índice de Jaccard".

Su cálculo viene dado por el cociente de la cardinalidad de la intersección de dichos conjuntos dividida entre la cardinalidad de su unión (es decir, en el denominador no cuentan los elementos que no pertenecen a ninguno de los dos conjuntos):

$$Sim_{Jacc}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(\frac{|\boldsymbol{x}_i \cap \boldsymbol{x}_j|}{|\boldsymbol{x}_i \cup \boldsymbol{x}_j|}\right)$$

Obviamente el rango de valores que puede adoptar está comprendido entre cero y uno.

Presenta un concepto asociado denominado "distancia de Jaccard" que se obtiene restado de uno la similitud de Jaccard:

$$d_{Jacc}(\boldsymbol{x}_i, \boldsymbol{x}_j) = 1 - Sim_{Jacc}(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

y que cumple todos los axiomas de una métrica.

La "medida de similitud de Rogers-Tanimoto" [78] es idéntica en cuanto a formulación a la anteriormente citada[14]:

$$Sim_{RoTa}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{\sum_a (x_{i,a} \wedge x_{j,a})}{\sum_a (x_{i,a} \vee x_{j,a})}$$

Pero, por el contrario, la "distancia de Rogers-Tanimoto" viene dada por[15]:

$$d_{RoTa}(\boldsymbol{x}_i, \boldsymbol{x}_j) = -\log_2 Sim_{RoTa}(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

La denominada distancia de Rogers-Tanimoto no es realmente una métrica ya que no cumple con el axioma de la desigualdad triangular.

En su comienzo, ambas medidas de similitud fueron usadas para estudios botánicos, hoy en día son usadas para la identificación de una huella dactilar [79] o para la selección de compuestos químicos [80] y drogas.

---

[14] En esta fórmula se ha preferido usar los operadores de conjunción y disyunción lógica en vez de los operadores de intersección y unión típicos de la teoría de conjuntos.

[15] Esta medida de distancia se aplica solamente cuando la similitud de Rogers-Tanimoto no sea igual a cero.

Medida de similitud de Rusell-Rao

Su formulación es similar a las métricas de Jaccard y Rogers-Tanimoto, en este caso la similitud de Rusell-Rao viene dada por [81]:

$$Sim_{RuRa}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(\frac{|\boldsymbol{x}_i \cap \boldsymbol{x}_j|}{A}\right)$$

donde A es el número total de elementos del universo (en el caso de vectores, la dimensión del vector). Del análisis de la fórmula se comprueba que en esta medida también se incluye en la cardinalidad del denominador a aquellos elementos que no están presentes en ninguno de los dos conjuntos.

Se usa en genética [82], meteorología [83]…

Índice de Sørensen-Dice

Su formulación también comparte muchos puntos en común con las medidas de similitud de Jaccard y de Rusell-Rao. Su diferencia radica en que la no presencia de un elemento en ambos conjuntos es ignorada en el denominador y las concordancias en el mismo elemento ponderan el doble en el numerador [84]:

$$d_{SoDi}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(\frac{2.|\boldsymbol{x}_i \cap \boldsymbol{x}_j|}{|\boldsymbol{x}_i| + |\boldsymbol{x}_j|}\right)$$

Este índice es también conocido como "índice o medida de Czekanowski".

Nació en el ámbito de la ecología y hoy en día sigue siendo su principal campo de utilización, aunque también se ha extendido a otros campos de la biología [85].

Distancia de Levenshtein

Probablemente es el método más usado para medir la proximidad entre dos secuencias de símbolos (habitualmente letras) [86]. Se calcula como el mínimo número de ediciones atómicas (por ediciones atómicas se entiende inserciones, borrados y reemplazos) necesarias para convertir una cadena de símbolos $\boldsymbol{x}_1$ en otra cadena $\boldsymbol{x}_2$.

- Inserción: $ins(\boldsymbol{x}_1, i, c) = x_{1,1}, x_{1,2}, \dots, x_{1,i}, c, x_{1,i+1}, \dots$
- Borrado: $borr(\boldsymbol{x}_1, i) = x_{1,1}, x_{1,2}, \dots, x_{1,i-1}, x_{1,i+1}, \dots$
- Reemplazo: $reempl(\boldsymbol{x}_1, i, c) = x_{1,1}, x_{1,2}, \dots, x_{1,i-1}, c, x_{1,i+1}, \dots$

Se usa habitualmente en los correctores ortográficos para detectar palabras mal escritas y sugerir cuáles son las posibles candidatas para sustituirla [87] [88] [89] o para variaciones en las cadenas de ADN [90].

Se demuestra que cumple con las tres condiciones necesarias para considerarla una métrica.

Existen variantes de esta medida como son la distancia de Damerau-Levenshtein [91] en las que el intercambio de dos caracteres adyacentes se considera también una operación atómica:

- Intercambio: $interc(\boldsymbol{x}_1, i) = x_{1,1}, x_{1,2}, \dots, x_{1,i-1}, x_{1,i+1}, x_{1,i}, x_{1,i+2} \dots$

Medida de similitud de Needleman-Wunsch

Intenta determinar las regiones similares de dos cadenas de códigos por medio del alineamiento global de sus secuencias [92].

Para recrear el algoritmo se crea una matriz bidimensional de resultados intermedios donde el encabezado de las filas representa uno de los códigos y el encabezado de las columnas el otro; la primera fila y la primera columna se rellenan con valores que empiezan en cero y van incrementándose en un valor d al avanzar en dicha fila o columna (-1 de acuerdo al algoritmo original). Posteriormente, cada celda se rellena con el máximo valor obtenido al evaluar cada uno de estos tres resultados:

- El valor del elemento anterior en su misma diagonal más un valor positivo (+1 en el trabajo original) si el elemento que encabeza esa fila y el que encabeza la columna coinciden; ese valor será negativo (-1) si difieren[16].
- El valor del elemento que se encuentra a su izquierda menos un valor de penalización d por tener que introducir un hueco en la secuencia colocada en la columna inicial (valor -1 en el trabajo original).
- El valor del elemento que se encuentra encima menos un valor de penalización d por tener que introducir un hueco en la secuencia colocada en la fila inicial (valor -1 en el trabajo original).

Al mismo tiempo, en cada "casilla" de dicha matriz se apunta cuál, o cuáles, han sido los caminos por los que se ha llegado a ese valor máximo.

Cuando se termina de rellenar la matriz, su último elemento tendrá la puntuación total para la similitud entre ambas secuencias completas de códigos. Para elegir la combinación de coincidencias/inserciones/borrados para conseguir este resultado se procederá a "viajar hacia atrás" por las indicaciones que se han añadido en la matriz, hasta llegar a la primera de sus celdas.

De esta medida de similitud existe una variación a reseñar que es el "algoritmo de Smith-Waterman". Al contrario que el anterior, el alineamiento que emplea este algoritmo es de tipo local [93]. Emplea una estrategia similar al algoritmo de Needleman-Wunsch, pero para cada elemento de la matriz no se admiten valores negativos (lo que es equivalente a indicar que cuando dos secuencias difieren mucho se comienza de cero "localmente" a buscar otro patrón de similitud). Además, en el "viaje hacia atrás", se comienza por la casilla (o casillas) de máxima puntuación hasta llegar a una casilla de valor 0. Los dos fragmentos que así se obtienen se corresponderán con las secuencias locales de ambas cadenas de códigos que presentan una fuerte relación.

[16] Existen múltiples trabajos donde se recogen distintas formas de ponderar estos "encuentros" y "desencuentros".

Estos dos algoritmos han sido objeto de múltiples trabajos de investigación para disminuir su tiempo de cálculo y los requerimientos de memoria [94] (también se han realizado implementaciones en FPGA, GPU…).

Más que en textos escritos, el uso de estos algoritmos se centra en el mundo de la biología para encontrar secuencias semejantes de nucleótidos en genes, aminoácidos en proteínas [95]; pero también en visión estereoespacial [96], detección de plagios [97]…

4.1.4.6 Clasificación de las métricas de acuerdo a su rango de aplicabilidad

De acuerdo a si la función presenta o no la misma expresión para medir distancias (o si los parámetros son los mismos o distintos) en función del punto a clasificar, las medidas de similitud/disimilitud entre casos basadas en distancias se pueden dividir en dos grandes grupos:

- Globales.
- Locales.

En esta tesis se han abordado investigaciones en ambos campos.

Métricas globales

Para medir distancias, este tipo de métricas emplean la misma función (o conjunto de parámetros) en todos los puntos del espacio de atributos.

Así pues, las técnicas de aprendizaje usadas emplean todos los datos empíricos disponibles y buscan un compromiso para que los parámetros minimicen una determinada función objetivo cuando se aplica al conjunto total de información disponible. Esto significa que la optimización de la medida de distancias en unas zonas del espacio de atributos puede entrar en conflicto con esa misma optimización en otras áreas.

En este grupo de métricas se engloban la mayoría de las que se han utilizado tradicionalmente en los algoritmos de clasificación. Dentro de los muchos algoritmos que optimizan una métrica o medida de distancia global se encuentran:

- Algoritmo VSM: sección 4.4.1.
- Algoritmo RCA [19] [98] [99].
- Algoritmo NCA [100].
- Algoritmo LMNN: sección 4.4.5.

Métricas locales

En este caso se busca que la medida de distancia presente distinto comportamiento en las distintas partes del espacio de atributos [101] [61].

La optimización de estas métricas se puede llevar a cabo mediante la minimización de su error de clasificación (cumpliendo además con un conjunto de restricciones), pero ajustando su evaluación a entornos locales.

Para construir métricas locales, habitualmente se utiliza un enfoque que permite que los valores de sus parámetros varíen de un punto a otro; también se puede emplear un conocimiento "a priori", que actúe como parámetro, y que sea dependiente del punto del espacio en el que se encuentra el caso a clasificar [102] [103] [104].

El diseño y uso de este tipo de métricas es un tema de más reciente investigación, y dentro de los algoritmos que optimizan una métrica o medida de distancia local se encuentran:

- Algoritmo DANN: sección 4.4.2.
- Algoritmo LFM-SVM: sección 4.4.3.
- Algoritmo LDAW: versión LDA del algoritmo "*LDA/SVM Driven Nearest Neighbor Classification*" [105].
- Algoritmo MORF: versión SVM del algoritmo "*LDA/SVM Driven Nearest Neighbor Classification*" [105].
- Algoritmo LaMaNNa: sección 4.4.4.

4.2 Métodos de clasificación relacionados con esta tesis

En esta sección se van a revisar aquellos métodos de clasificación, presentes en la Tabla 2.1, y que han sido utilizados con más intensidad en esta investigación. Estos son:

- Búsqueda de vecinos próximos.
- Análisis del discriminante.
- Máquinas de vectores de soporte.

En cada una de estas subsecciones se explica con cierta profundidad el objetivo básico del método y la tecnología en que se basan, las expresiones matemáticas (que se convierten en las fórmulas que posteriormente se programarán) y una pequeña reflexión final sobre las ventajas e inconvenientes de cada uno de ellos.

4.2.1 Búsqueda de vecinos próximos

Los métodos que juzgan la clase de un caso atendiendo a las clases de sus vecinos más próximos han tenido gran predicamento en el mundo de la IA [106] [14] [16].

El método 1-NN fue introducido formalmente por Fix y Hodges en 1951 [107] y su extensión, el método *k*-NN, hoy en día sigue siendo uno de los métodos no paramétricos usados habitualmente en la clasificación de casos. Estos métodos son muy flexibles y no exigen asunciones especiales en los datos que manejan. A pesar de su simplicidad, y de los años que han transcurrido desde su introducción, su rendimiento en los problemas de clasificación sigue siendo sorprendentemente bueno [108] [109] [110].

Una magnifica compilación de artículos científicos, sobre la evolución de este método de clasificación y los avances hasta 1990, se recoge en una colección especial de la IEEE [111]. Liangxiao et al. [112] presentaron en 2007 una comunicación con una recopilación de mejoras para los algoritmos de clasificación basados en vecinos próximos. En el año 2010 Bathia [113] realizó otra importante recopilación donde se recogía el estado del arte de las modalidades de este algoritmo; en su artículo se identifican hasta 17 variaciones principales.

Por último y relacionándolo con la métrica para calcular la distancia, se citará que Cover y Hart [106] mostraron que, asintóticamente, el error máximo cometido por el método 1-NN nunca será mayor que dos veces el que se obtiene mediante el método óptimo de Bayes.

$$\text{si } \lim_{N\to\infty} error_{1-NN}(\boldsymbol{x}) = \mathrm{e}, \quad \mathrm{e}_{bayes} \leq \mathrm{e} \leq \mathrm{e}_{bayes}(2-\mathrm{e}) \leq 2\,\mathrm{e}_{bayes} \qquad (4.4)$$

En la práctica, y aunque no se llegue a la densidad de casos requerida por el criterio asintótico, el error suele aproximarse mucho al óptimo obtenible.

4.2.1.1 *Objetivo*

k-NN es un método de clasificación supervisado que basa su funcionamiento en estimar la función de densidad de probabilidad de pertenecer a una determinada clase en función de los casos que se encuentran en el entorno de un caso. Adaptando el parámetro *k*, para fijar el tamaño del entorno, este algoritmo establece localmente una función de decisión no lineal que permite clasificar otros casos.

4.2.1.2 *Principio de funcionamiento*

El método de clasificación 1-NN asigna al caso actual la misma clase que la que posee su vecino más próximo, de acuerdo a una métrica definida en el espacio de los atributos (habitualmente euclídea). En la Figura 4.4 se puede apreciar una interpretación geométrica del funcionamiento de este método.

El método *k*-NN es una extensión del 1-NN, difiere en que elige la clase examinando cuál es la clase que más se repite entre los *k* casos más cercanos al que se desea clasificar (en caso de empate se decide aleatoriamente el resultado).

El valor óptimo de *k* depende de cada problema y hay que ajustarlo individualmente. Como no es posible establecer una expresión analítica que relacione el error cometido con el valor de *k*, la mayor parte de los investigadores establecen un procedimiento de optimización mediante la exploración exhaustiva para fijar cuál es el valor óptimo de *k* para un determinado problema[17].

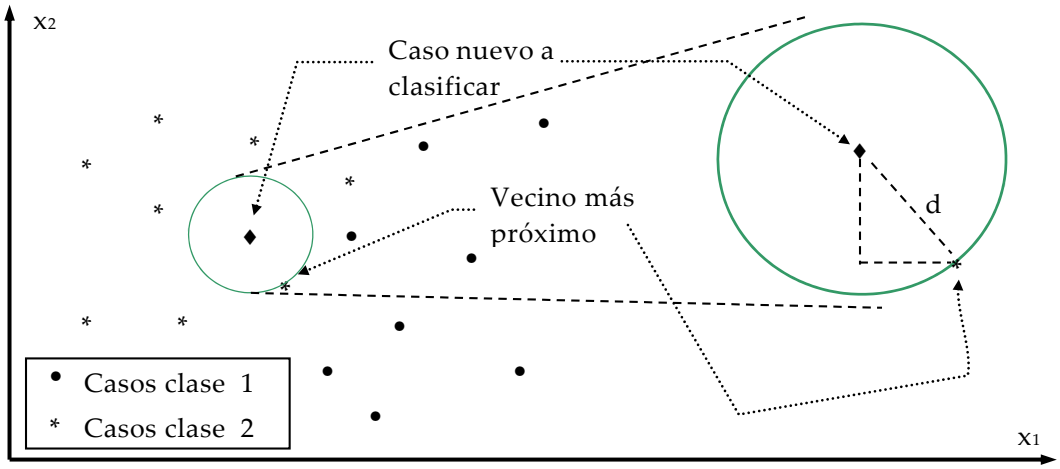


Figura 4.4.- Clasificación de un caso en función del vecino más próximo (1-NN).

No obstante, se ha constatado que el valor óptimo de *k* se extiende en un rango amplio de valores y que, por lo tanto, calcular su valor exacto no es tan importante. En la Figura 4.5 se puede apreciar este comportamiento para el problema de las flores de Iris. Entre los valores 10 y 20 para *k*, el rango de

[17] Es bastante económico realizar una exploración exhaustiva ya que k solo puede tomar valores enteros positivos.

aciertos en la predicción varía entre el 95,3% y 96,7% (que se corresponden con 143 y 145 aciertos sobre 150 casos).

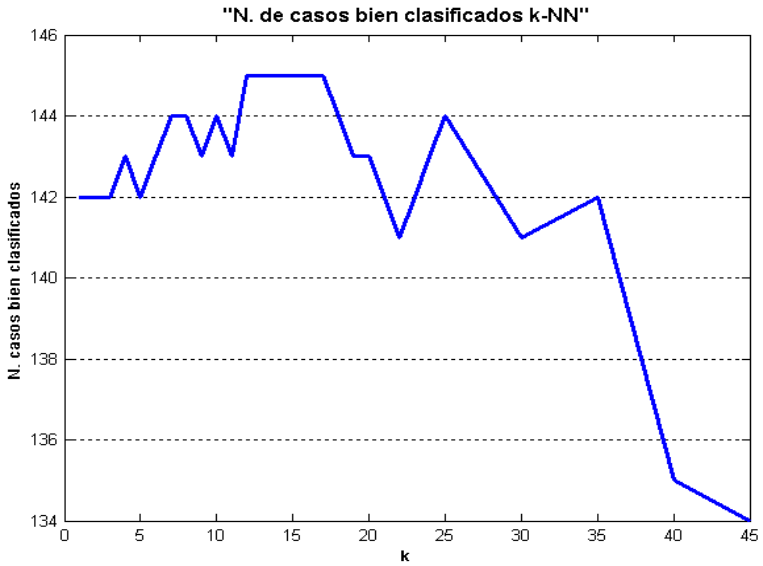


Figura 4.5.- Elección del valor óptimo de k para el problema de las flores de Iris.

Las fronteras entre clases obtenidas mediante el método *k*-NN son menos irregulares que las del método 1-NN, ya que al promediar sobre un conjunto de casos, existen menos posibilidades de que un caso "extraño" imponga su criterio clasificatorio erróneo en una zona del espacio de atributos.

4.2.1.3 Métrica usada y su aprendizaje

La versión tradicional del algoritmo *k*-NN, utiliza casi en exclusividad la métrica euclídea[18].

En la mayor parte de las aplicaciones, antes de calcular las distancias entre casos se "normalizan" los valores de los atributos, haciendo que sus distribuciones tengan media 0 y desviación 1. Esto hace que un atributo cuyos valores tengan un gran recorrido no monopolice la decisión de la clasificación[19].

En 1990, la recopilación de artículos realizada por Dasarathy [111] reconocía que había pocos artículos relacionados con *k*-NN y las métricas. En uno de los estudios pioneros, Short y Fukunaga [114] planteaban una métrica local en las proximidades del caso a clasificar; poco después el propio Fukunaga [115] proponía emplear una métrica global de tipo Mahalanobis y optimizarla.

Más recientemente, los trabajos de Lowe [55], Domeniconi [63] y otros abordaron con mayor detalle las métricas a emplear en *k*-NN y su

[18] Es la métrica que se implementa en los programas informáticos habituales de clasificación, "Data Mining"…

[19] Ahora bien, existen problemas (como el de las "waveforms") en los que se obtienen mejores resultados si no se procede a dicha normalización.

optimización (estos enfoques se exponen con mayor detalle en el estudio del estado del arte, sección 4.4.1 y ss.).

4.2.1.4 Ventajas e inconvenientes

Estos métodos de clasificación son de tipo no paramétrico y, por lo tanto, el investigador no necesita adoptar un modelo previo sobre el problema a clasificar. Es más, se acomodan muy bien a clasificaciones con fronteras irregulares (no lineales).

A lo largo de distintas investigaciones, se ha revelado entre los métodos más simples y que ofrece mayores tasas de aciertos en problemas reales [108].

Su comportamiento asemeja a una caja negra, relacionando valores de entrada con los de salida, pero sin proporcionar explicaciones sobre los porqués.

Sufre fuertemente del problema conocido como "maldición de la dimensionalidad" (ver sección 2.6.2.2). La medida de la distancia se degrada si aparecen muchas variables que no están relacionadas con el resultado de la clasificación, ya que estas incrementan aleatoriamente la distancia entre dos casos, sin que dicho incremento esté relacionado con la distancia realmente "útil" entre casos.

A la hora de establecer la clasificación, la relevancia de los distintos atributos no tiene por qué ser idéntica. En los casos más complejos, dicha relevancia depende además de la "localización" del caso a clasificar en el espacio de los atributos. Es necesario que los casos prototipo representen bien al conjunto de las poblaciones de las distintas clases, sobre todo en las fronteras entre ellas.

Las técnicas 1-NN o *k*-NN no proporcionan un valor de confianza para la predicción. Puede que se hayan encontrado muchos vecinos de la misma clase muy cercanos, y por lo tanto la predicción sea fiable; o que los casos más próximos se encuentren lejos y pertenezcan a distintas clases, preponderando solo muy ligeramente aquella que se ha ofrecido como resultado de la clasificación.

4.2.1.5 Trabajo relacionado en la tesis

Se usa el método de clasificación *k*-NN en los dos algoritmos desarrollados en esta tesis: BTW y LOM. También se utiliza la versión que utiliza una métrica euclídea como algoritmo de comparación.

4.2.2 Análisis del discriminante

Tanto el análisis discriminante lineal (LDA: "Linear Discriminant Analysis") como el cuadrático (QDA: "Quadratic Discriminant Analysis") son técnicas de clasificación que tratan de establecer las fronteras entre clases determinando el lugar geométrico de los puntos que tienen la misma probabilidad de pertenecer a cada una de ellas. Aparte de esta interpretación probabilística, el LDA se puede contemplar desde el punto de vista de una métrica, y es este el enfoque que se va a utilizar en esta exposición.

4.2.2.1 Objetivo

LDA es un método de clasificación supervisado que se basa en la reducción de la dimensionalidad del espacio de atributos. Su propósito general es hallar una transformación lineal óptima que maximice la separabilidad entre clases en el espacio transformado.

4.2.2.2 Enfoque

Incrementar la separabilidad entre clases se logra minimizando la distancia intraclase a la vez que se aumenta la interclases. Dicha separabilidad será óptima cuando resulte máximo el ratio entre la distancia interclase y la intraclase de los datos.

Si indicamos como $f(\boldsymbol{L})$ la función objetivo a optimizar y $\boldsymbol{B}$ y $\boldsymbol{W}$ las matrices de covarianza interclase e intraclase respectivamente, el problema de optimización a resolver será:

$$f(\boldsymbol{L}) = \underset{\boldsymbol{L}}{arg\,max}\ \ tr\{\,(\boldsymbol{L}^T\boldsymbol{B}\,\boldsymbol{L})\cdot(\boldsymbol{L}^T\boldsymbol{W}\,\boldsymbol{L})^{-1}\,\}$$

o (4.5)

$$f(\boldsymbol{L}) = \underset{\boldsymbol{L}}{arg\,min}\ \ tr\{\,(\boldsymbol{L}^T\boldsymbol{B}\,\boldsymbol{L})^{-1}\cdot(\boldsymbol{L}^T\boldsymbol{W}\,\boldsymbol{L})\,\}$$

con:

$$\boldsymbol{W} = \frac{1}{N}\sum_{j=1}^{J}\sum_{n=1}^{N_j}(\boldsymbol{x}_n - \boldsymbol{\mu}_j)\,(\boldsymbol{x}_n - \boldsymbol{\mu}_j)^T \qquad \boldsymbol{B} = \frac{1}{N}\sum_{j=1}^{J} N_j\,(\boldsymbol{\mu}_j - \overline{\boldsymbol{\mu}})(\boldsymbol{\mu}_j - \overline{\boldsymbol{\mu}})^T \qquad (4.6)$$

donde:

$\boldsymbol{\mu}_j = \frac{1}{N_j}\sum_{n=1}^{N_j}\boldsymbol{x}_n$ es el vector media de la clase *j*.

$\overline{\boldsymbol{\mu}} = \frac{1}{N}\sum_{n=1}^{N}\boldsymbol{x}_n$ es el vector media de todos los datos.

El problema de optimización planteado en (4.5) tiene una solución analítica desde el punto de vista de valores y vectores propios. Esto es:

$$\boldsymbol{B}\,\boldsymbol{v} = \lambda\,\boldsymbol{W}\,\boldsymbol{v} \qquad (4.7)$$

donde $\boldsymbol{v}$ representa el vector propio asociado al valor propio λ del problema generalizado de vectores y valores propios de las matrices $\boldsymbol{B}$ y $\boldsymbol{W}$.
Por lo que la solución al problema es:

$$\boldsymbol{W}^{-1}\boldsymbol{B}\,\boldsymbol{v} = \lambda\,\boldsymbol{v} \tag{4.8}$$

Para reducir la dimensión de A a A_r ($A_r < A$) se formará una matriz $\boldsymbol{L}$ con los primeros A_r vectores propios de $\boldsymbol{W}^{-1}\boldsymbol{B}$. Es decir:

$$\boldsymbol{L} = \left[\boldsymbol{l}_1, \boldsymbol{l}_2, \ldots, \boldsymbol{l}_{A_r}\right] \quad \in \quad \mathbb{R}^{A \times A_r} \tag{4.9}$$

Así pues, para transformar un caso formado por los valores de sus atributos $\boldsymbol{x}_i$ en el espacio $\mathcal{X}$ a sus nuevas coordenadas $\boldsymbol{z}_i$ en el espacio $\mathcal{Z}$:

$$\boldsymbol{z}_i = \boldsymbol{L}^T\boldsymbol{x}_i \tag{4.10}$$

Para la clasificación de un nuevo caso $\boldsymbol{x}_o$ se calcula la distancia entre $\boldsymbol{z}_o$ y los valores de los centros de las clases $\boldsymbol{\mu}_j$ transformados también a las nuevas coordenadas.

4.2.2.3 Inconvenientes del LDA estándar

Dependiendo de los datos empíricos manejados, es posible encontrar los siguientes problemas:

- Problema de singularidad de la matriz $\boldsymbol{W}$ (o submuestreo).
- La matriz de covarianzas intraclase, al ser un promedio sobre todas las clases, generalmente resulta bastante diferente a las matrices de covarianzas de los casos de cada clase.
- Al estar basado el LDA estándar en la norma L_2, el algoritmo es sensible a los datos "extraños".

Problema de singularidad de la matriz $\boldsymbol{W}$

En problemas con un pequeño tamaño de muestras, comparado con la dimensión A del espacio de atributos, es decir si $N \ll A$, la matriz $\boldsymbol{W}$ es singular y el algoritmo LDA estándar no es aplicable. Para solventar este problema se han propuesto ciertas variantes del LDA y el algoritmo LDA/FKT.

El algoritmo LDA/FKT [116] se fundamenta en la descomposición del espacio original en cuatro subespacios con diferentes niveles de "discriminatividad" que se miden mediante los ratios, $\lambda = \lambda_B/\lambda_W$, de los valores propios de las matrices $\boldsymbol{B}$ y $\boldsymbol{W}$ obtenidos aplicando la llamada transformación FKT ("Fukunaga-Koontz transform").

Para llevar a cabo esta transformación, se considera en primer lugar la matriz de covarianza de todos los datos $\boldsymbol{S} = \boldsymbol{W} + \boldsymbol{B}$ (a la cual se la supone de rango r) y se la diagonaliza obteniéndose:

$$\boldsymbol{S} = [\boldsymbol{U} \quad \boldsymbol{U}_\perp] \begin{bmatrix} \boldsymbol{D} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{U}^T \\ \boldsymbol{U}_\perp^T \end{bmatrix}$$
$$\boldsymbol{D} = diag\,\{\lambda_1, \lambda_2, \dots, \lambda_r\} \quad \lambda_i \geq \lambda_{i+1}\,, \lambda_r > 0 \tag{4.11}$$

donde $\boldsymbol{U}_\perp$ es el conjunto de vectores propios del subespacio nulo (que se demuestra que es la intersección de los espacios nulos de $\boldsymbol{B}$ y $\boldsymbol{W}$).

Se define una transformación que "esferoide" los datos: $\boldsymbol{P} = \boldsymbol{U}\boldsymbol{D}^{-1/2}$. Para luego aplicarla a las matrices $\boldsymbol{B}$ y $\boldsymbol{W}$ y, por último, proceder a una descomposición espectral de las matrices transformadas:

$$\widetilde{\boldsymbol{B}} = \boldsymbol{P}^T\,\boldsymbol{B}\,\boldsymbol{P} = \boldsymbol{V}\,\boldsymbol{\Lambda}_B\,\boldsymbol{V}^T$$
$$\boldsymbol{\Lambda}_B = diag\,\{\lambda_1, \lambda_2, \dots, \lambda_B\} \quad \lambda_i \geq \lambda_{i+1}\,, \lambda_B > 0$$
$$\widetilde{\boldsymbol{W}} = \boldsymbol{P}^T\,\boldsymbol{W}\,\boldsymbol{P} = \boldsymbol{V}\,\boldsymbol{\Lambda}_W\,\boldsymbol{V}^T \tag{4.12}$$
$$\boldsymbol{\Lambda}_W = diag\,\{\lambda_1, \lambda_2, \dots, \lambda_W\} \quad \lambda_i \geq \lambda_{i+1}\,, \lambda_W > 0$$

Zhang et al. analizan las cuatro combinaciones posibles que resultan de los subespacios de estas últimas descomposiciones:

Descomposición del espacio original en subespacios		Caracterización	
Combinación de subespacios	Definición	Ratio $\lambda = \lambda_B/\lambda_W$	
S1	span($\widetilde{\boldsymbol{B}}$) ∩ null($\widetilde{\boldsymbol{W}}$)	∞	$\lambda_W = 0, \lambda_B = 1$
S2	span($\widetilde{\boldsymbol{B}}$) ∩ span($\widetilde{\boldsymbol{W}}$)	$0 < \lambda < \infty$	$0 < \lambda_W < 1$ $0 < \lambda_B < 1$
S3	null($\widetilde{\boldsymbol{B}}$) ∩ span($\widetilde{\boldsymbol{W}}$)	0	$\lambda_W = 1, \lambda_B = 0$
S4	null($\widetilde{\boldsymbol{B}}$) ∩ null($\widetilde{\boldsymbol{W}}$)		$\lambda_W = 0, \lambda_B = 0$

Tabla 4.4.- Combinaciones de los subespacios de acuerdo al algoritmo LDA/FKT .

Los subespacios S1, S2 y S3 son discriminantes, solo el S4 no tiene información útil. Fukunaga-Koontz determina la matriz de transformación lineal usando la descomposición QR; el espacio transformado resultante será la unión de los subespacios S1, S2 y S3, que representa la mayor capacidad discriminante posible.

Las distintas clases no tienen una misma matriz de covarianzas

Cuando esto ocurre la matriz de covarianzas intraclase conjunta resulta bastante diferente de la matriz de covarianzas de cada clase. Para evitar tanta

variabilidad se ha propuesto usar, para cada pareja de clases, la matriz de covarianzas promediada [117] y definir la función objetivo en términos de la medida de divergencia-KL (Kullback-Leibler).

4.2.2.4 *Métrica usada y su aprendizaje*

El método de clasificación basado en el discriminante lineal se puede considerar como una proyección del espacio de los atributos sobre una línea.

En la Figura 4.6 se puede apreciar cómo la interpretación geométrica del método LDA "proyecta" los valores de los atributos de los casos sobre la recta que separa los centros de las dos clases (todo ello en el espacio "esferoidado" respecto a la matriz ***W***). En esta recta (espacio de dimensión 1) se realiza ahora la clasificación de los casos.

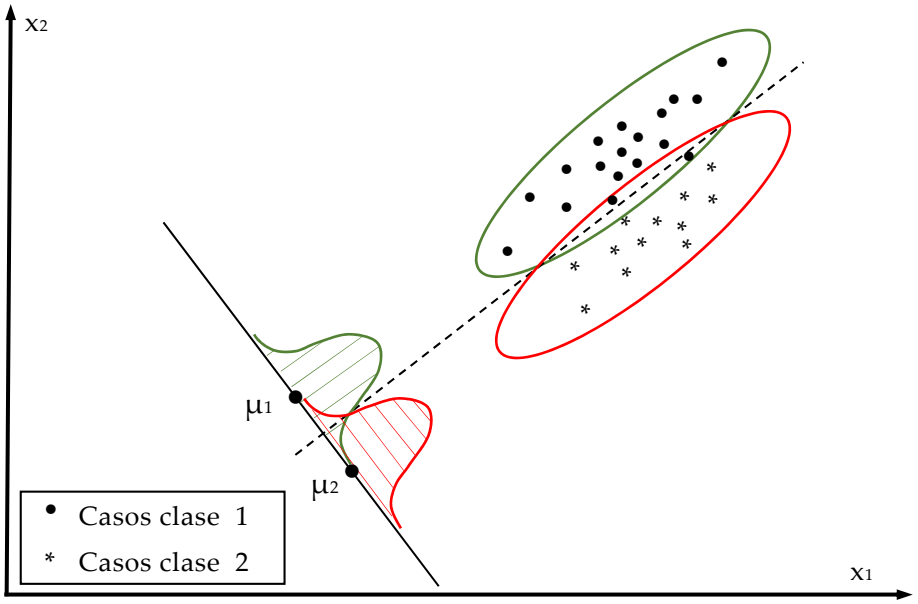


Figura 4.6.- Transformación implícita en el análisis del discriminante lineal.

4.2.2.5 *Ventajas e inconvenientes*

LDA no presenta parámetros a estimar/optimizar (si se exceptúan los vectores de medias de las clases y las matrices de covarianzas), lo cual refuerza su robustez; pero es poco adaptable a problemas donde la orientación de la frontera entre clases varíe mucho en función del punto del espacio considerado.

Este criterio de clasificación está fuertemente basado en que todas las clases presentan distribuciones normales e iguales, y en los problemas empíricos reales esto no suele ser cierto. A pesar de ello, los métodos de análisis discriminante han ofrecido muy buenos resultados en muy diversos estudios [108] y ha sido, y es, uno de los métodos que en primer lugar prueban los investigadores.

En vez de emplear distribuciones normales, otros autores han propuesto emplear funciones de distribución más sofisticadas y flexibles, pero es

imprescindible disponer de un conocimiento previo sobre su tipo (lo cual no suele ser cierto) si se desea mejorar la calidad de las clasificaciones.

4.2.2.6 Trabajo relacionado en la tesis

En esta tesis se utilizarán las direcciones marcadas por el algoritmo LDA para obtener las direcciones discriminantes de la métrica BTW (ver sección 5.2.4.1). También se utiliza LDA como algoritmo de comparación.

4.2.3 Máquinas de vectores de soporte (SVM)

Las SVM hunden sus raíces en las técnicas tradicionales de separación de clases mediante las técnicas de planos: LDA, regresión logística, perceptrones… serían sus referencias anteriores más próximas.

Ahora bien, a la luz de los estudios de Vapnik [118], en los últimos años del siglo XX apareció una nueva técnica de clasificación y regresión que intenta reducir lo más posible el error en las predicciones; no mediante la estrategia tradicional de reducir el error cuadrático de las predicciones, sino en alejar lo más posible los casos más "dudosos" de las fronteras de separación entre clases.

4.2.3.1 Objetivo

Establecer una superficie de decisión lineal en el espacio de los atributos, o en el de Hilbert asociado al "*kernel*" elegido, que permita separar los casos de las distintas clases con el máximo margen de seguridad posible. Posteriormente, la clase de un caso se determinará en función del semiplano en que este se encuentre.

4.2.3.2 SVM para clases separables, principio de funcionamiento

La derivación clásica de los algoritmos en que se basan las SVM [119] [120] parte de considerar dos clases perfectamente separables y de buscar el plano óptimo que hace de frontera entre ellas; posteriormente se extiende dicho algoritmo al escenario en el que las dos clases no son separables linealmente. Y, por último, se sustituye el plano por una superficie no lineal, proyectando el espacio de los atributos en el denominado espacio de las características y calculando los productos internos en dicho espacio por medio de "*kernels*".

Partiendo de un conjunto de N casos experimentales pertenecientes a dos clases linealmente separables $\{\boldsymbol{x}_i, y_i\},\ i = 1..N$, donde:

- $\boldsymbol{x}_i$ es el vector con los valores de los atributos de caso i-ésimo.
- $y_i \in \{-1, 1\}$ es la etiqueta que señala la clase a la que pertenece dicho caso.

se pretende establecer un plano: $\boldsymbol{w}^T.\ \boldsymbol{x} + b = 0$ que separe de forma óptima ambas clases, siendo $\boldsymbol{w}$ el vector director de dicho plano.

Las SVM toman por óptimo aquel plano que posea un mayor margen de distancia a los casos más cercanos de cada una de las clases. Estos puntos reciben el nombre de "vectores de soporte" (SV) y son los que realmente condicionan la posición y orientación del plano (el resto de casos no tendrían transcendencia alguna).

En la Figura 4.7 se pueden apreciar los casos de la clase 1 (círculos) y los de la clase -1 (estrellas). Se puede ver que la línea de separación gruesa se "aleja" lo más posible de los casos más cercanos de cada clase. Estos casos (marcados

en el gráfico por puntos más gruesos) son los vectores de soporte. Sobre las distintas rectas figuran sus correspondientes ecuaciones.

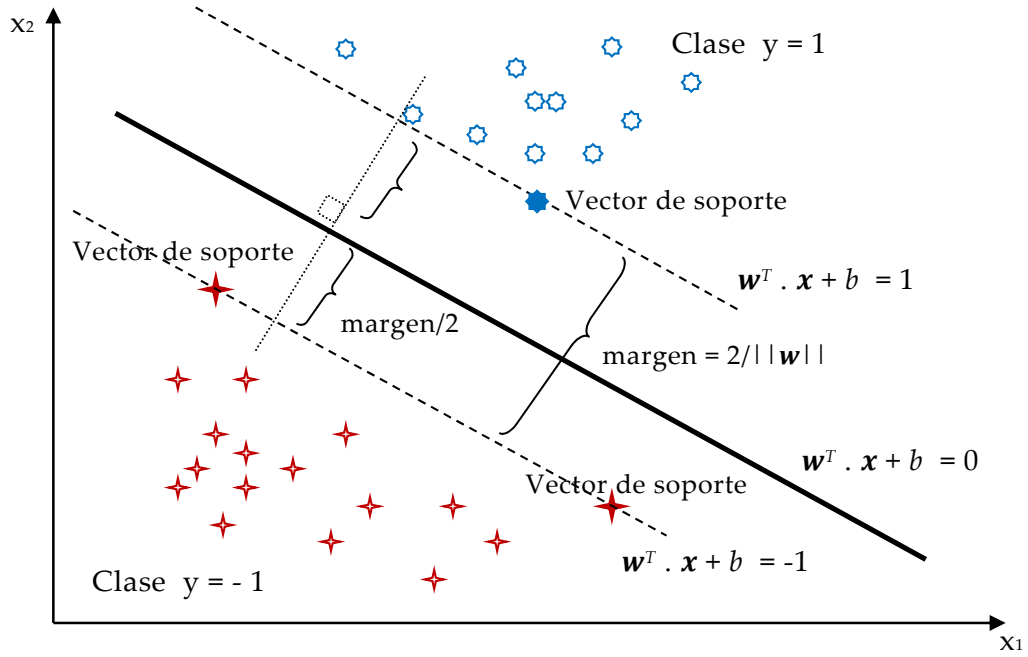


Figura 4.7.- Recta óptima, de acuerdo al algoritmo SVM, que separa dos clases en el espacio R^2.

Cuando los distintos casos son separables linealmente, siempre es posible plantear una expresión lineal de tipo $\boldsymbol{w}^T . \boldsymbol{x} + b$, y escalar el vector $\boldsymbol{w}$ y el parámetro b para conseguir que todos los casos de la clase 1 cumplan con la inecuación $\boldsymbol{w}^T . \boldsymbol{x} + b \geq +1$; y que todos los de la clase -1 con $\boldsymbol{w}^T . \boldsymbol{x} + b \leq -1$. [20]

Lo cual puede ser resumido mediante la inecuación:

$$y_i . (\boldsymbol{w}^T . \boldsymbol{x}_i + b) - 1 \geq 0, \quad \forall\ i \qquad (4.13)$$

Como bien se conoce, la distancia del origen al plano $\boldsymbol{w}^T . \boldsymbol{x} + b = +1$ es $| 1 - b | / \| \boldsymbol{w} \|$.

Y por tanto la distancia entre los planos: $\begin{cases} \boldsymbol{w}^T . \boldsymbol{x} + b = +1 \\ \boldsymbol{w}^T . \boldsymbol{x} + b = 0 \end{cases}$ será $1/\| \boldsymbol{w} \|$.

Y también la que hay entre los planos: $\begin{cases} \boldsymbol{w}^T . \boldsymbol{x} + b = -1 \\ \boldsymbol{w}^T . \boldsymbol{x} + b = 0 \end{cases}$ será $1/\| \boldsymbol{w} \|$.

Por tanto, el margen de separación entre los planos extremos que delimitan ambas clases es $2/\| \boldsymbol{w} \|$. Este margen representa también la distancia euclídea entre los puntos más próximos de ambas clases proyectada sobre la dirección perpendicular al plano separador.

[20] A este plano se le denomina "plano canónico".

El algoritmo que optimiza las SVM se propone como objetivo maximizar el margen de separación entre clases. De esta forma se intenta dificultar que futuros casos a clasificar, y que puedan estar contaminados con ruido o que están muy cerca del borde, sean adjudicados a la clase errónea (ver Figura 4.7 y Figura 4.8).

Maximizar dicho margen: $2/\|\boldsymbol{w}\|$, viene a representar lo mismo que minimizar la norma del vector $\|\boldsymbol{w}\|$ elevada al cuadrado. Así pues, la función objetivo a minimizar será:

$$Q(\boldsymbol{w}) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} = \frac{1}{2}\|\boldsymbol{w}\|^2 \tag{4.14}$$

Y estará sometida a las restricciones lineales:

$$y_i \,.\, (\boldsymbol{w}^T.\ \boldsymbol{x}_i + b) - 1 \geq 0, \quad \forall\ i \tag{4.15}$$

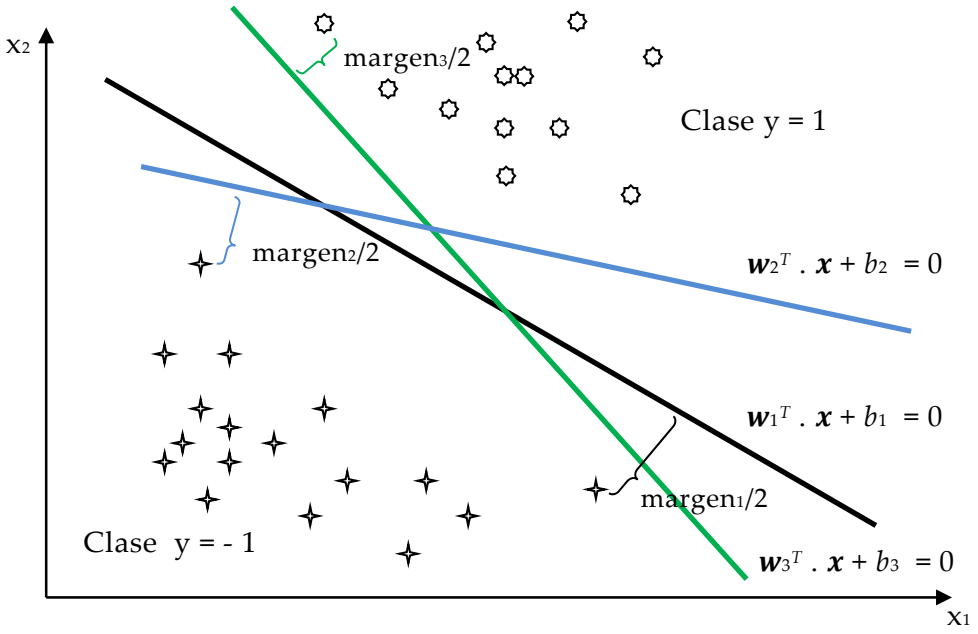


Figura 4.8.- Distintas rectas que, aunque separan correctamente las dos clases, no todas ellas presentan el mismo margen.

La forma habitual de resolver este problema de minimización es mediante la técnica de los multiplicadores de Lagrange.

Así pues, se plantea el Lagrangiano:

$$L_P(\boldsymbol{w}, b, \alpha_i) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N} \alpha_i (\, y_i \,.\, (\boldsymbol{w}^T.\ \boldsymbol{x}_i + b) - 1\,) \tag{4.16}$$

donde α_i son los multiplicadores de Lagrange.

Es un problema de minimización cuadrático convexo, ya que tanto la función objetivo como el conjunto de puntos que satisfacen las restricciones (4.15) son convexos.

En el mínimo se debe cumplir que se anulan las derivadas parciales de L_P respecto a $\boldsymbol{w}$ y b:

$$\frac{\partial}{\partial \boldsymbol{w}} L_P = \boldsymbol{w} - \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i = 0 \tag{4.17}$$

$$\frac{\partial}{\partial b} L_P = -\sum_{i=1}^{N} \alpha_i y_i = 0 \tag{4.18}$$

Y también se deben satisfacer las condiciones complementarias de Karush-Kuhn-Tucker (ver sección 3.1.2):

$$\begin{array}{l} \alpha_i \,.\, (\, y_i \,.\, (\, \boldsymbol{w}^T .\, \boldsymbol{x}_i + \, b) \;\; - 1\,) = 0 \\ \alpha_i \geq \, 0 \end{array} \qquad i = 1..N \tag{4.19}$$

Por razones que se verán claras más adelante, en este problema de minimización lo que realmente interesa es obtener los valores de α_i en función de los valores de $\boldsymbol{x}_i$ e y_i para posteriormente determinar $\boldsymbol{w}$ y b. Para focalizar el problema en los multiplicadores de Lagrange, se pasa del Lagrangiano expresado en forma primal a su forma dual[21] (o de Wolf).

Para ello se despeja $\boldsymbol{w}$ de la ecuación (4.17) y al introducirla en la expresión (4.16) y tener en cuenta la ecuación (4.18), el resultado es el siguiente problema dual, cuyo Lagrangiano es:

$$L_D(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k \boldsymbol{x}_i^T \boldsymbol{x}_k \tag{4.20}$$

sujeto a las condiciones:

$$\begin{array}{l} \sum_{i=1}^{N} \alpha_i y_i \;\; = 0 \\ \alpha_i \geq 0 \quad \forall\, i \end{array} \tag{4.21}$$

En los escenarios de clases linealmente separables, la gran mayoría de los multiplicadores óptimos α_{0_i} son cero y, de acuerdo a la expresión (4.19), permite que los casos a los que se aplican puedan estar alejados del plano separador más de una unidad, es decir, se cumpliría para aquellos casos en que:

$$\begin{array}{ll} \boldsymbol{w}^T .\, \boldsymbol{x}_i + \, b \;\; > 1 & \forall\; i \;\mid y_i = 1 \\ \boldsymbol{w}^T .\, \boldsymbol{x}_i + \, b \;\; < -1 & \forall\; i \;\mid y_i = -1 \end{array} \tag{4.22}$$

Aquellos multiplicadores α_{0i} distintos de cero se aplican a los casos que están sobre el plano canónico y que, por tanto, cumplen con la igualdad de la inecuación (4.13). A estos casos, que como se verá a continuación son los

[21] Como es conocido (ver sección 3.1.2), maximizando el Lagrangiano en forma dual se llega al mismo resultado que minimizando el Lagrangiano en forma primaria.

únicos relevantes en la clasificación, se les denomina "vectores de soporte" (SV). En la Figura 4.7, los casos marcados con más énfasis son los SVs de ese ejemplo.

Una vez obtenidos los valores de α_{0_i}, los parámetros $\boldsymbol{w}_0$ y b_0 óptimos se pueden obtener de las expresiones (4.17), (4.18) y (4.19), llegándose a:

$$\begin{aligned} \boldsymbol{w}_0 &= \sum_{i=1}^{nSV} \alpha_{0_i} y_i \boldsymbol{x}_i \\ b_0 &= \frac{1}{n_{SV}} \sum_{i=1}^{n_{SV}} (y_i - \boldsymbol{w}_0^T . \boldsymbol{x}_i) \end{aligned} \tag{4.23}$$

Examinando las expresiones en (4.23), se ve ahora con claridad que todos aquellos casos con el multiplicador α_{0_i} igual a cero no influyen en absoluto en el cálculo de la dirección del plano que separa óptimamente las clases.

A partir de ahora, el problema de clasificar un nuevo caso $\boldsymbol{x}_k$ se reducirá a sustituir los valores de los atributos en la ecuación del plano óptimo:

$$f_{dec}\,(\,\boldsymbol{x}_k) = \boldsymbol{w}_0^T . \boldsymbol{x}_k + b_0 \tag{4.24}$$

siendo $f_{dec}\,(\,\boldsymbol{x}_k)$ la función de decisión. Si su resultado es mayor que cero será asignado a la clase denominada +1, y si es menor que cero a la clase -1.

Se puede comprobar que el tiempo necesario para realizar el cálculo que lleva a la clasificación de un nuevo caso es reducido (depende linealmente del número de atributos del problema y del número de vectores de soporte) y por ello puede ser usado en algoritmos que deban tomar decisiones en tiempo real.

4.2.3.3 SVM para clases no separables, principio de funcionamiento

La inmensa mayoría de los problemas prácticos no admiten la separación perfecta de las clases mediante un plano:

- Muchos de ellos, porque la frontera entre clases no se adapta a un plano sino que presenta un comportamiento no lineal.
- En otros problemas, y aunque la separación óptima pudiera ser lineal, la dispersión[22] de cada conjunto de datos es lo suficientemente grande para que existan múltiples solapamientos entre los casos de ambas clases. Así pues, existirán casos que caerán "al otro lado" del semiespacio que les correspondería.

  No se podrá esperar una clasificación con una tasa de error del 0%, pero, aun así, sería deseable poder separar linealmente ambas clases.

[22] La dispersión se puede deber a la propia pdf a la que responden los atributos de los casos de cada clase, o a estar los valores de los atributos contaminados con ruido.

Este último escenario puede ser resuelto por las SVM, de forma muy elegante, con una mínima modificación de las ecuaciones expuestas en la sección anterior.

Se aprecia claramente que en este nuevo escenario el problema de minimización expuesto en la sección anterior no proporcionará ninguna solución porque el conjunto de inecuaciones (4.15), que se repite a continuación:

$$y_i \,.\, (\boldsymbol{w}^T .\, \boldsymbol{x}_i + b) - 1 \geq 0\,, \quad \forall\ i \tag{4.25}$$

no se puede satisfacer en aquellos casos cuya clasificación correcta es imposible.

Para evitarlo, se introduce en (4.25) una pequeña relajación en dichas inecuaciones:

$$y_i \,.\, (\boldsymbol{w}^T .\, \boldsymbol{x}_i + b) \ -1 \ + \xi_i \ \geq 0 \quad \forall\ i\ ,\ \xi_i \geq 0 \tag{4.26}$$

Siendo la variable de holgura ("slack variable") ξ_i un término no negativo que ayudaría a cumplir con cada inecuación.

Para todos aquellos casos que estuvieran en el semiespacio correcto de acuerdo al plano canónico de su clase, el valor de ξ_i sería 0. Aquellos que sobrepasasen el margen de seguridad y se acercasen al plano de separación entre clases tendrían valores de ξ_i comprendidos en el intervalo: $0 < \xi_i < 1$. Y los que cruzasen "al otro lado", y por tanto no estuviesen bien clasificados, presentarían valores de $\xi_i > 1$.

En la Figura 4.9 se puede apreciar una interpretación geométrica de las variables de holgura ξ_i.

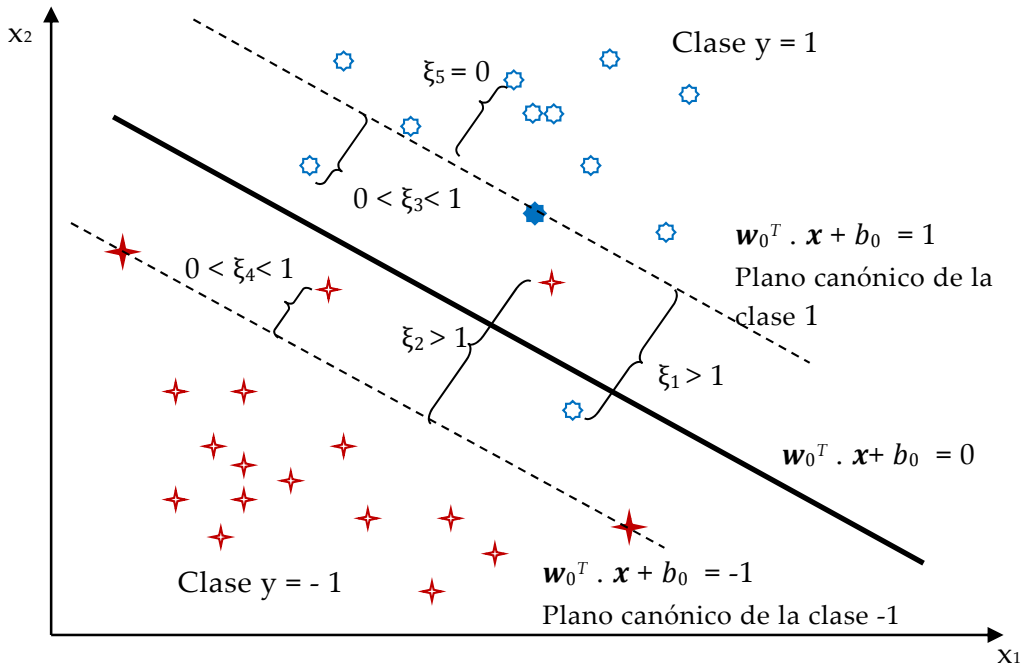


Figura 4.9.- Interpretación geométrica de las variables de holgura.

En el nuevo problema de minimización que se presenta, no solo interesa hacer que el margen de separación entre los vectores de soporte sea lo más grande posible, sino también intentar que los valores de las variables ξ_i sean lo más pequeños posible.

Por tanto la nueva formulación será:

$$Q(\boldsymbol{w}) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_{i=1}^{N}\xi_i \tag{4.27}$$

$$y_i \,.\, (\boldsymbol{w}^T.\ \boldsymbol{x}_i + b) - 1 + \xi_i \geq 0, \quad \forall\ i \tag{4.28}$$

$$\xi_i \geq 0 \quad \forall\ i \tag{4.29}$$

donde *C* es un parámetro a elegir que "castiga" el hecho de que los valores de las variables de holgura sean grandes, logrando un equilibrio entre estos y conseguir un margen de separación elevado.

Su resolución sigue los mismos pasos expuestos en las fórmulas (4.16) y ss. Siendo el Lagrangiano expresado en su forma primal:

$$\begin{aligned} L_P(\boldsymbol{w}, b, \xi_i, \alpha_i, \beta_i) &= \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}\xi_i - \\ &- \sum_{i=1}^{N}\alpha_i(\, y_i \,.\, (\boldsymbol{w}^T.\ \boldsymbol{x}_i + b) - 1 + \xi_i) - \sum_{i=1}^{N}\beta_i\,\xi_i \end{aligned} \tag{4.30}$$

con:

$$\begin{aligned} \alpha_i &\geq 0 \\ \beta_i &\geq 0 \end{aligned} \qquad \forall\ i \tag{4.31}$$

Anulando sus derivadas parciales respecto a $\boldsymbol{w}$, b y ξ_i:

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{w}} L_P &= \boldsymbol{w} - \sum_{i=1}^{N}\alpha_i y_i \boldsymbol{x}_i = 0 \\ \frac{\partial}{\partial b} L_P &= -\sum_{i=1}^{N}\alpha_i y_i = 0 \\ \frac{\partial}{\partial \xi_i} L_P &= C - \alpha_i - \beta_i = 0 \end{aligned} \tag{4.32}$$

e introduciendo en L_P las expresiones resultantes de $\boldsymbol{w}$ y C para escribir el Lagrangiano en su forma dual, se obtiene:

$$L_D(\alpha) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{N}\alpha_i\alpha_k y_i y_k \boldsymbol{x}_i^T\boldsymbol{x}_k \tag{4.33}$$

Esta expresión coincide con la establecida anteriormente en la ecuación (4.20).

En este caso, y debido a la condición impuesta por las ecuaciones (4.32) y a que los valores de las variables de holgura deben ser no negativos, los valores de α_i no pueden crecer de forma indefinida, estando limitado su rango a:

$$0 \leq \alpha_i \leq C, \quad \forall\ i \tag{4.34}$$

Las fórmulas (4.23) que permiten calcular $\boldsymbol{w}_0$ y b_0 en función de los α_{0i} , $\boldsymbol{x_i}$ e y_i siguen siendo válidas para este escenario.

Se mantiene también la importante propiedad de que (4.33) solo depende de las etiquetas de las clases y de los productos escalares de los atributos de las distintas combinaciones de casos.

4.2.3.4 SVM para la clasificación no lineal, principio de funcionamiento

Es muy difícil encontrar un problema práctico en el que una frontera lineal sea la función subyacente que clasifica de forma ideal los casos de dos clases. Las interacciones entre atributos, junto con la ponderación no siempre proporcional de estos a la hora de fijar la clase a la que pertenece un caso, hacen que los planos representen una simplificación de la realidad, muchas veces inapropiada.

Boser et al. [121] propusieron que el espacio original de atributos podría ser proyectado en otro espacio de mayor dimensionalidad ("espacio de características") de forma que en este último, y mediante el algoritmo SVM, se pudiera calcular el plano lineal que mejor separase las clases; el cual se correspondería con una superficie no lineal en el espacio original de los atributos.

Así pues, esta transformación proyecta vectores de atributos $\boldsymbol{x}_i$ (en $\mathbb{R}^A$) en vectores de características $\boldsymbol{z}_i$ (en $\mathbb{R}^F$):

$$\boldsymbol{x} \xrightarrow{\Phi} \boldsymbol{z} = \Phi(\boldsymbol{x}) \tag{4.35}$$

Como se ha comentado, la expresión (4.33) indica que el problema de maximización se puede resolver solamente conociendo los productos escalares de los valores de los atributos de los distintos casos.

En el espacio de características dicha expresión sería:

$$L_D(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k \Phi^T(x_i)\,.\,\Phi(x_k) \tag{4.36}$$

Y la función de decisión (4.24) quedaría:

$$f_{dec}\,(\,\boldsymbol{x}_k) = \sum_{i=1}^{nSV} \alpha_i y_i \Phi^T(\boldsymbol{x}_i)\ .\Phi(\boldsymbol{x}_k) + b_0 \tag{4.37}$$

Como se puede comprobar, en ambas expresiones simplemente se han sustituido los productos escalares $\boldsymbol{x}_i^T\ .\ \boldsymbol{x}_k$ por sus equivalentes en el espacio de

características: $\Phi^T(\boldsymbol{x}_i)\,.\,\Phi(\boldsymbol{x}_k)$. Y se espera que en este espacio ampliado se puedan separar linealmente las dos clases.
Realmente no es necesario transformar de vuelta los parámetros $\boldsymbol{w}_0$ y b_0 del espacio de las características hacia el de los atributos, ya que la función de decisión (4.37) se calcula directamente en el espacio de las características y ahí se toma la decisión sobre a qué clase pertenece un caso $\boldsymbol{x}_k$.

Lo más interesante de estas expresiones es que realmente no es necesario que la función de transformación Φ sea explicitada. Es suficiente con conocer el producto escalar:

$$\mathbf{z}^T.\ \mathbf{z}\ =\ \Phi^T(\boldsymbol{x}_i)\ .\Phi(\boldsymbol{x}_k)\ =\ K(\boldsymbol{x}_i,\boldsymbol{x}_k) \tag{4.38}$$

donde la función $K(\ )$ es denominada "función *kernel*".

Así pues, la expresión final para la función de decisión de una SVM empleada en un caso no lineal será:

$$f_{dec}\ (\boldsymbol{x}_k) = \sum_{i=1}^{nSV} \alpha_i y_i K(\boldsymbol{x}_i,\boldsymbol{x}_k) + b_0 \tag{4.39}$$

4.2.3.5 Funciones que pueden ser usadas como "kernels"

La función "*kernel*" puede ser diseñada con bastante libertad, teniendo en cuenta dos restricciones:

- Su cálculo deber ser posible empleando solamente $\boldsymbol{x}_i$ y $\boldsymbol{x}_k$.
- Debe cumplir que:

$$\iint K(\boldsymbol{x},\boldsymbol{y})\ .g(\boldsymbol{x})\ .g(\boldsymbol{y})\ \ d\boldsymbol{x}\,d\boldsymbol{y}\ >0 \qquad \forall\, g \in L_2(\mathbb{R}^N) \tag{4.40}$$

  donde $g(\ )$ es cualquier función con una norma euclídea finita en el espacio de los atributos[23].
- $K(\ )$ debe ser simétrica y definida positiva.

Todas estas restricciones se conocen como la condición de Mercer.
Algunas funciones "*kernel*" que cumplen dichas condiciones y son empleadas habitualmente en las SVM son:

$$K(\boldsymbol{x},\boldsymbol{y}) = (\boldsymbol{x}^T.\ \boldsymbol{y}\ +1\ )^{\lambda} \tag{4.41}$$

$$K(\boldsymbol{x},\boldsymbol{y}) = e^{-\gamma\ \|\boldsymbol{x}-\boldsymbol{y}\|^2} \tag{4.42}$$

$$K(\boldsymbol{x},\boldsymbol{y}) = \tanh\ (\kappa\,\boldsymbol{x}^T.\boldsymbol{y} - \delta) \tag{4.43}$$

[23] Lo cual es equivalente a decir que $\int g^2(\mathbf{x})\ d\mathbf{x}\ <\ \infty$.

donde los parámetros representados mediante letras griegas representan valores a elegir por el usuario.

Al "*kernel*" de la ecuación (4.41) se lo conoce como polinómico, al de la ecuación (4.42) como "Radial Basis Functions" (RBF), y al de la ecuación (4.43) como sigmoidal.

Aunque el "*kernel*" lineal da excelentes resultados en múltiples problemas prácticos, muchas investigaciones apuntan a que el que más habitualmente proporciona mejores resultados es el "*kernel*" RBF; siendo poco usado en la práctica el "*kernel*" sigmoidal.

4.2.3.6 Ventajas e inconvenientes

A pesar del consenso existente en la comunidad científica de que no existe un método de clasificación que supere a todos los demás en todos los dominios[24] [122], las máquinas de vectores de soporte son hoy en día uno de los métodos que presentan un mejor rendimiento para estas tareas:

- Las SVM fueron uno de los primeros métodos que se fundamentaron en la teoría de la minimización del riesgo estructural de Vapnik. Muchas de sus características positivas, como su reducido error de generalización, derivan de este hecho.
- Dentro de los problemas linealmente separables, las SVM presentan una robustez muy alta[25], así como una de las mejores tasas de generalización. En los problemas no lineales, las SVM con "*kernels*" de tipo RBF baten con regularidad al resto de métodos en cuanto a la precisión de clasificación.
- La maximización del margen de separación deviene en un problema de optimización convexa, la cual garantiza la existencia de un único máximo dentro de la región objetivo. El descubrimiento en 1998 del método "Sequential Minimal Optimization"(SMO) [123] ha permitido resolver eficientemente el problema de minimización en las SVM.

Como principales inconvenientes se pueden citar:

- El método habitual para abordar los problemas de clasificación con múltiples clases consiste en enfrentarlas por parejas y luego combinar los resultados de la clasificación.
- La elección del "*kernel*" es a decisión del usuario. No existe una teoría comúnmente aceptada para la construcción de "*kernels*" especialmente adaptados para un problema.
- Para evitar el sobreaprendizaje es necesario aplicar un método de validación cruzada, tanto para la selección de los parámetros del "*kernel*" como para elegir el factor que penaliza un valor grande en las variables de holgura. Esto puede generar un número enorme de

[24] Teorema del "No free lunch".

[25] Tolerancia respecto a casos contaminados con error.

SVM a entrenar (se suele aplicar el método de optimización conocido como "búsqueda en malla", ver sección 3.1.1.1).

- Las SVM que mejores resultados obtienen en los problemas de clasificación presentan también un elevado número de vectores de soporte (sobre todo en las SVM-RBF) y tienden al sobreaprendizaje.
- La fase de entrenamiento requiere de importantes recursos computacionales y suele consumir mucho tiempo.
- No es fácil justificar los resultados de una clasificación (en función de los pesos de los SV…).
- En la fase de aprendizaje no es fácil incorporar información "a priori" del dominio a clasificar.

4.2.3.7 Trabajo relacionado en la tesis

El cálculo de la función de separación entre dos clases mediante SVM-RBF se utiliza en los desarrollos del algoritmo LOM de esta tesis. También se utiliza como algoritmo de comparación.

4.3 Clasificación de los algoritmos de aprendizaje de métricas

Tal como se ha indicado en la introducción de este capítulo, se va a proporcionar también una visión global sobre los métodos de aprendizaje para las métricas, indicando las funciones de distancia o similitud a las que se destinan y describiendo brevemente algunos algoritmos relacionados con ellos.

4.3.1 Criterios de clasificación

Los algoritmos de aprendizaje de métricas que mejoren alguna de sus características pueden agruparse atendiendo a:

- La métrica o función de distancia/similitud seleccionada.
- El enfoque para la parametrización o aprendizaje de la métrica elegida.
- Caracterización del algoritmo desde el punto de vista de la realización conjunta o no del proceso de entrenamiento y la predicción de la clase.
- La presencia de una única solución o de soluciones múltiples.

Para una mejor comprensión de las grandes familias de algoritmos de aprendizaje aplicables a mejorar la clasificación supervisada nos apoyaremos en la figura adjunta.

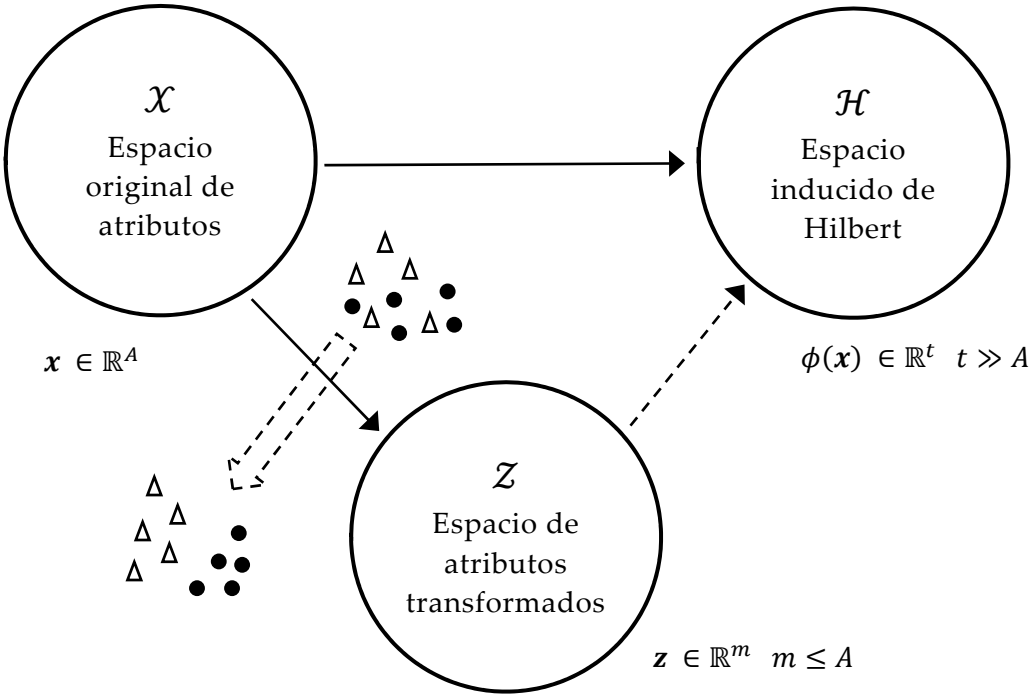


Figura 4.10.- Representación de los distintos espacios involucrados en el aprendizaje de una métrica.

Atendiendo a la forma de la función distancia o similitud seleccionada para el aprendizaje se distinguen entre métricas lineales y no lineales.

El aprendizaje de una métrica consiste en parametrizar dicha función. Este proceso se puede realizar usando los datos para entrenar un único conjunto de parámetros o varios conjuntos. En el primer caso los valores de los parámetros calculados serán únicos en todo el espacio original (y se dice que el algoritmo es global). En el caso de particularizar la métrica en función de la posición en el espacio, el algoritmo será local y los valores de los parámetros de la función distancia o similitud obtenidos pueden variar de una zona a otra dentro del espacio original de los atributos.

Por enfoque para la parametrización o aprendizaje de la métrica seleccionada queremos indicar el procedimiento matemático adoptado. Dicho procedimiento generalmente es un proceso de optimización, que en ocasiones puede reformularse en la forma de un problema de valores y vectores propios generalizados.

En definitiva, se trata de un planteamiento y de la resolución de un problema de optimización restringido. Las restricciones se formulan usando la información adquirible "a priori" acerca de la equivalencia o no de las clases de los casos de aprendizaje (o de la similitud de las mismas).

En cuanto a la caracterización del algoritmo considerando si el entrenamiento se realiza conjuntamente con la predicción de la clase de un caso, se hablará de algoritmos demorados ("lazy") o no demorados.

Desde el punto de vista del alcance de la solución, se distinguen entre algoritmos de solución simple (cuando la solución óptima obtenida es única) o múltiple (cuando pueden producirse varias soluciones que no tienen por qué ser los óptimos absolutos).

A continuación se exponen los fundamentos básicos de las métricas lineales, explicando al mismo tiempo los aspectos derivados de los demás criterios de clasificación señalados.

Las métricas lineales pueden estar basadas en una función de distancia o similitud y parametrizarse desde un enfoque global o local. Las métricas locales tienen "a priori" tanta importancia como las globales, por lo que se ha optado a estudiarlas separadamente. Las métricas locales se han desarrollado para mejorar la precisión de clasificación en el contexto del k-NN, pero existe el riesgo de problemas de sobreaprendizaje que se puede evitar introduciendo en la función objetivo un término de regularización.

Las métricas globales dan en principio mejores resultados en el contexto de las SVM, más que cuando se hibridan con el algoritmo k-NN.

Las métricas no lineales basadas en distancias se construyen "kernelizando" las lineales o directamente en el espacio inducido $\mathcal{Z}$.

También existen métricas no lineales basadas en medidas de similitud (siendo una muy utilizada la métrica coseno [124]). Estas métricas no se abordan en esta tesis al estar fuera de los objetivos de la misma.

4.3.2 Métricas lineales basadas en distancias

Con el objetivo de resolver un problema de clasificación se utiliza por defecto la distancia euclídea y en algunas ocasiones una de tipo Mahalanobis.

La métrica debería intentar mover las instancias de una misma clase hasta conjuntarlas en una misma región de tamaño más reducido y desplazar al mismo tiempo hacia otras regiones las instancias de otras clases (Figura 4.10).

Salvo en escenarios con distribuciones especiales de los casos, no es posible obtener esta solución ideal cuando se trabaja con datos empíricos y una métrica euclídea. De ahí que se busque una solución óptima en relación a las posibles restricciones de pertenencia de los casos a una u otra clase, una determinada región de vecindad o no, etc.

La métrica convencional euclídea no es parametrizable, por lo que conduce a una solución no óptima. Por esta causa, muchas de las métricas que se han planteado están basadas en una distancia de tipo Mahalanobis.

4.3.2.1 Fundamentos de métricas basadas en una distancia de tipo Mahalanobis

Sea $\boldsymbol{X} = \{\boldsymbol{x}_i, y_i\}$, $1 \leq i \leq N$, el conjunto de datos empíricos representados mediante vectores de dimensión A en el espacio $\mathcal{X}$ original de los atributos.

Es decir, $\boldsymbol{x}_i \in \mathbb{R}^A$, siendo A el número de atributos. El conjunto $\boldsymbol{X}$ de datos empíricos está compuesto de casos de J clases (correspondientemente etiquetados con los valores y_i), o sea:

$$\boldsymbol{X} = \left[\begin{array}{ccccc|c} x_{1,1} & x_{1,2} & \dots & x_{1,A-1} & x_{1,A} & y_1 \\ x_{2,1} & x_{2,2} & \dots & x_{2,A-1} & x_{2,A} & y_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ x_{N,1} & x_{N,2} & \dots & x_{N,A-1} & x_{N,A} & y_N \end{array}\right] \tag{4.44}$$

Seleccionar una métrica de tipo Mahalanobis implica determinar una matriz de transformación lineal de manera que en el espacio $\mathcal{Z}$ de los atributos transformados linealmente, es decir $\boldsymbol{Z} = \boldsymbol{L}\,\boldsymbol{X}$, las diferentes clases resultan lo suficientemente separadas, y dentro de una misma clase las instancias quedan conjuntadas en un área de vecindad reducida. O sea,

$$\boldsymbol{Z} = \left[\begin{array}{ccccc|c} z_{1,1} & z_{1,2} & \dots & z_{1,A-1} & z_{1,A} & y_1 \\ z_{2,1} & z_{2,2} & \dots & z_{2,A-1} & z_{2,A} & y_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ z_{N,1} & z_{N,2} & \dots & z_{N,A-1} & z_{N,A} & y_N \end{array}\right] \tag{4.45}$$

con:

$$d_E^2(\boldsymbol{z}_i, \boldsymbol{z}_j) = d^2(\boldsymbol{L}\boldsymbol{x}_i, \boldsymbol{L}\boldsymbol{x}_j) = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{L}^T \boldsymbol{L} (\boldsymbol{x}_i - \boldsymbol{x}_j) = \\ = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{M} (\boldsymbol{x}_i - \boldsymbol{x}_j) = d_M^2(\boldsymbol{x}_i, \boldsymbol{x}_j) \text{ con } \boldsymbol{M} \succcurlyeq \boldsymbol{0} \quad (4.46)$$

donde $\boldsymbol{M}$ pertenece al conjunto de matrices semidefinidas positivas.
Esta expresión indica que una distancia de tipo Mahalanobis calculada en el espacio original de los atributos puede interpretarse como una distancia euclídea en el espacio de los atributos transformados linealmente.

Por otra parte, se observa que la matriz de la métrica $\boldsymbol{M}$ puede tener o no una forma factorizada, es decir $\boldsymbol{M} = \boldsymbol{L}^T \boldsymbol{L}$.
En este último caso la matriz de la métrica es $\boldsymbol{L}$ y no se pone ninguna restricción sobre $\boldsymbol{M}$, porque resultará siempre simétrica y semidefinida positiva, cualquiera que sea $\boldsymbol{L}$. Al usar la matriz $\boldsymbol{M}$ en forma no factorizada se puede restringir su estructura a una de las siguientes formas:

$$\boldsymbol{M} = \sigma \boldsymbol{I} \quad \text{(matriz escalar)}$$
$$\boldsymbol{M} = diag\{ w_a \}, \quad 1 \leq a \leq A \quad \text{(matriz diagonal)} \quad (4.47)$$

Si $\boldsymbol{M}$ es una matriz escalar, los elementos transformados se presentan en un nuevo sistema de coordenadas que resulta simplemente escalado idénticamente en todas las direcciones. Al igual que en una distancia euclídea convencional, ninguna dirección es más discriminante que otras, porque todas ellas tienen la misma relevancia con respecto a la discriminación. Los ejes de coordenadas del espacio $\mathcal{Z}$ son paralelos a los del espacio $\mathcal{X}$ y la separabilidad entre clases no ha variado.

Por el contrario, si $\boldsymbol{M}$ es una matriz diagonal, y no son iguales todos sus elementos, serán más discriminantes las direcciones con mayores pesos w_a, aumentando la separabilidad entre clases en esas direcciones.
En este caso, existen A parámetros cuyos valores habrá que fijar a través del proceso de aprendizaje; dichos valores representan los factores de escala de los ejes de coordenadas (por lo que se puede contemplar como un problema de escalado). Fue Lowe [55] quien planteó primero una optimización sin restricciones para estos parámetros, si bien su planteamiento desde la óptica de una métrica se hizo más tarde [18].

Si se adopta una matriz $\boldsymbol{M}$ completa sin factorizar, esta debe ser simétrica y semidefinida positiva, y posee $N\,(N+1)/2$ parámetros a ajustar por medio de un procedimiento de aprendizaje. Comparativamente, el aprendizaje es más intensivo computacionalmente que el realizado cuando es una matriz diagonal, el algoritmo es menos robusto frente a fluctuaciones en los datos (causados por ruido) y subyace un posible problema de sobreaprendizaje.
Como contrapartida los ejes de coordenadas son orientables hacia las direcciones espaciales que puedan ser más interesantes. Esto significa que

una métrica con una matriz completa es más eficaz que la de una matriz diagonal. Esta última permite separar bien los casos cuyos atributos están organizados paralelamente a los ejes coordenados de $\mathcal{X}$ o en cualquier otra dirección.

4.3.2.2 Aprendizaje de métricas lineales globales

El problema de aprendizaje de una métrica vía una transformación lineal se estudia desde dos puntos de vista:

- Concepción de un problema de optimización, el cual comprende la formulación de una función objetivo y la selección de las restricciones que debe verificar la solución óptima buscada.
- Resolución del mismo.

En el tercer capítulo se han presentado someramente los diferentes procedimientos de resolución de un problema de optimización. Una vez planteado, se analiza su estructura para determinar si es convexo. En tal caso se averigua si puede encuadrarse en la clase programación cuadrática o en la programación semidefinida. Si se acomoda a alguna de estas técnicas, el problema de optimización puede resolverse eficientemente utilizando los algoritmos específicos para esta clase de problemas. Si no se encuadra en alguno de los casos particulares, se resolverá aplicando técnicas generales como el gradiente descendente, gradiente conjugado...

Formulación de la función objetivo

Generalmente, una función objetivo consta de dos términos, una función de coste (o suma de una o varias funciones de pérdida) y un término regularizador. Esto es [125]:

$$L(\boldsymbol{M}) = \underbrace{C \sum_{i} l_i(\boldsymbol{X}^T \boldsymbol{M} \boldsymbol{X})}_{\substack{\text{Suma de varias} \\ \text{funciones de pérdida}}} + \underbrace{r(\boldsymbol{M})}_{\substack{\text{Término de} \\ \text{regularización}}} \tag{4.48}$$

donde C representa un compromiso entre la influencia del regularizador y la penalización asociada a las respectivas funciones de pérdida.

El problema consiste en optimizar $L(\boldsymbol{M})$ dentro del dominio $\boldsymbol{M}$ (que es habitualmente el espacio $\mathcal{S}_+^A$). En ocasiones se restringe este dominio al espacio de matrices diagonales no negativas. Obsérvese que las funciones de pérdida están formuladas en términos de un producto interno, por lo que son lineales en $\boldsymbol{M}$.

Para completar esta formulación se deben definir las restricciones entre variables.

Formas más habituales de supervisión del aprendizaje de una métrica

Se trata de las restricciones que fijan las exigencias sobre las variables. Las formas más comunes son:

- Restricciones sobre distancias entre instancias similares y disimilares[26]:

$$\begin{aligned} &d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \leq \mu\,, \quad \boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{S} \qquad (\mathcal{S} \text{ es el conjunto de casos similares}) \\ &d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq \theta, \quad \boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{D} \quad (\mathcal{D} \text{ es el conjunto de casos disimilares}) \end{aligned} \tag{4.49}$$

  Se puede trasladar este concepto a la función de coste, pero ahora en forma de función de pérdida (usando por ejemplo la función de pérdida de Hinge):

$$\begin{aligned} l(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{M}) = &\max_{\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{S}} \left[0, d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) - \mu\right] = \left[d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) - \mu\right]_+ \\ l(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{M}) = &\max_{\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{D}} \left[0, \theta - d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)\right] = \left[\theta - d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)\right]_+ \end{aligned} \tag{4.50}$$

- Restricciones de distancias relativas:

  Generalmente se definen usando ternas $(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_l)$, en las cuales las distancias están calculadas desde $\boldsymbol{x}_i$ y expresan que la distancia desde $\boldsymbol{x}_i$ hasta $\boldsymbol{x}_j$ debe ser más pequeña que la que existe entre $\boldsymbol{x}_i$ y $\boldsymbol{x}_l$. Esto es:

$$d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \leq d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) \tag{4.51}$$

  Normalmente se le suele añadir un margen de seguridad γ, si bien en muchos casos se fija arbitrariamente a 1:

$$1 + d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \leq d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) \tag{4.52}$$

  De forma similar a lo explicado en el punto anterior, si se desease trasladar esta restricción a la función de coste por medio de la función de pérdida de Hinge:

$$l(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_l, \boldsymbol{M}) = \left[1 + d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) - d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l)\right]_+ \tag{4.53}$$

Aparte de estas formas comunes, se encuentran en la literatura otras formas de definir las restricciones como, por ejemplo, la acotación de la suma de distancias entre instancias similares.

La elección de una forma u otra depende de la aplicación específica.

[26] Este conjunto representa un conjunto de restricciones muy exigente.

Regularizadores más comunes

Los más habituales en la literatura [125] son:

$$\begin{aligned} r(\boldsymbol{M}) &= \frac{1}{2}\|\boldsymbol{M}\|_F^2 && \text{(cuadrado de la norma de Frobenius)} \\ r(\boldsymbol{M}) &= tr(\boldsymbol{M}\,\boldsymbol{W}) && \left(\text{si } \boldsymbol{W} = \boldsymbol{I}, \text{el regularizador resulta } tr(\boldsymbol{M})\right) \\ r(\boldsymbol{M}) &= tr(\boldsymbol{M}) - \log|\boldsymbol{M}| \end{aligned} \tag{4.54}$$

Casos representativos

Existen varios métodos que se usan habitualmente en el aprendizaje de métricas lineales, muchos de ellos se caracterizan por un cierto tipo de regularizador y una forma de representar las restricciones.

1) Regularización vía normas de Frobenius. Método de Schultz y Joachims

El objetivo es optimizar los elementos de una matriz tipo Mahalanobis para un problema con restricciones de distancias relativas.

La función de costo está basada en que las distancias relativas en las ternas se ajusten en lo posible a lo expresado en la ecuación (4.53), relajando por medio de la variable de holgura ξ_{ijl} la obligación de que se tenga que cumplir para toda $\boldsymbol{x}_l$ cuya clase sea distinta de la de $\boldsymbol{x}_i$. Además se utiliza el regulador de la norma de Frobenius y la obligación que $\boldsymbol{M} \in \mathcal{S}_+^A$.

Así pues, se puede plantear:

$$\begin{aligned} &\min_{\boldsymbol{M}} \quad \|\boldsymbol{M}\|_F^2 + C \sum_{i,j,l} \xi_{ijl} \\ &\text{sujeto a:} \\ &d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) \geq d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) + 1 - \xi_{ijl} \\ &\boldsymbol{M} \succcurlyeq 0, \qquad \xi_{ijl} \geq 0 \end{aligned} \tag{4.55}$$

Se puede eliminar la restricción de que $\boldsymbol{M} \in \mathcal{S}_+^A$, restringiendo su estructura a que se pueda expresar como: $\boldsymbol{M} = \boldsymbol{W}\,\boldsymbol{D}\,\boldsymbol{W}^T$, donde $\boldsymbol{D}$ es una matriz diagonal y $\boldsymbol{W}$ una matriz prefijada. Así se puede plantear:

$$\begin{aligned} &\min_{\boldsymbol{D}} \quad \|\boldsymbol{M}\|_F^2 + C \sum_{i,j,l} \xi_{ijl} \\ &\text{sujeto a:} \\ &d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) \geq d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) + 1 - \xi_{ijl} \\ &\xi_{ijl} \geq 0 \end{aligned} \tag{4.56}$$

El problema de optimización a resolver es de naturaleza similar al de las SVM y ejemplos de su aplicación se puede encontrar en el algoritmo POLA [126] y en el MLSVM [127].

2) Regularización lineal con $r(\boldsymbol{M}) = tr(\boldsymbol{M}\,\boldsymbol{W})$

Los algoritmos más representativos son el "Mahalanobis Metric Learning for Clustering" (MMC) desarrollado por Xing et al. [18] y el "Large Margin Nearest Neighbor Classification" (LMNN) de Weinberger et al. [128].

No se pasará a detallar el algoritmo MMC por tratarse de un método orientado al aprendizaje no supervisado y tampoco el LMNN porque se estudiará con detalle en la sección de los trabajos de referencia de la tesis (sección 4.4.5).

3) Regularización con $r(\boldsymbol{M}) = tr(\boldsymbol{M}) - \log|\boldsymbol{M}|$

El método más representativo es el "Information-Theoretic Metric Learning" (ITML) [129].

Aquí el problema de optimización formulado por los autores es:

$$\min_{\boldsymbol{M}} \quad tr(\boldsymbol{M}) - \log|\boldsymbol{M}|$$

sujeto a:

$$\begin{aligned} d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) &\leq \mu, \qquad \boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{S} \\ d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) &\geq \theta, \qquad \boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{D} \end{aligned} \tag{4.57}$$

Y se pueden relajar estas restricciones fuertes, introduciendo variables de holgura. En este caso el problema se convierte en:

$$\min_{\boldsymbol{M}} \quad tr(\boldsymbol{M}) - \log|\boldsymbol{M}| + C \sum_{i,j,l} \xi_{ijl}$$

sujeto a:

$$\begin{aligned} d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) &\leq \mu + \xi_{ijl}, \qquad \boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{S} \\ d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) &\geq \theta - \xi_{ijl}, \qquad \boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{D} \end{aligned} \tag{4.58}$$

4.3.2.3 Aprendizaje de métricas locales

Frente a los métodos globales en los que los valores de los parámetros ajustados mediante aprendizaje son únicos para todo el espacio $\mathcal{X}$ (o $\mathcal{Z}$), en los métodos locales estos valores pueden variar de una zona a otra (o en la vecindad de un caso $\boldsymbol{x}_o$ que se desea clasificar).

Dicha variabilidad tiene su origen en un eventual cambio fuerte de la densidad de probabilidad en las proximidades de la frontera de separación entre clases o en una alta dimensionalidad del problema considerado.

A tenor de esta consideración, se constata que existen dos causas principales para esa falta de homogeneidad en las probabilidades "a posteriori":

- Dispersión de los datos debido al problema COD.
- Distribución desigual de clases en la vecindad de $\boldsymbol{x}_o$ cuando este caso está próximo a la frontera de separación.

Una posible solución para la primera causa es determinar la relevancia de los atributos, ensanchar la región del vecindario de $\boldsymbol{x}_o$ en las direcciones en las que los atributos son menos relevantes y contraerlo en las más discriminantes

(más influyentes o relevantes). Esto da lugar a las métricas locales basadas en el escalado de los ejes de coordenadas, lo cual implica una ponderación de los atributos.

Para solucionar los efectos derivados de la segunda causa, el aprendizaje de una métrica debe enfocarse a incrementar la resolución espacial en las proximidades de la frontera de separación y disminuirla simultáneamente en cualquier otro punto. La frontera de separación será determinada en una etapa previa. Se fundamentan en este principio el aprendizaje vía las transformaciones cuasiconformes y el de una métrica riemanniana.

En ambos enfoques se busca modificar la vecindad de $\boldsymbol{x}_o$ para lograr que la función de densidad de probabilidad se mantenga en ella prácticamente uniforme; esto es $p(y_j|\boldsymbol{x}_0 + \delta\boldsymbol{x}) \cong p(y_j|\boldsymbol{x}_0)$.

Selección de una vecindad

La forma de elegirla depende de si el algoritmo a implementar es de tipo "lazy" o no, tal como se expone a continuación.

Algoritmos locales "lazy"

La vecindad se adapta a cada punto del espacio $\mathcal{X}$, y se calcula en el momento en que se presenta un nuevo caso $\boldsymbol{x}_0$ a clasificar. Es decir, dado el punto $\boldsymbol{x}_0$, se selecciona su vecindad y el proceso de aprendizaje de la métrica se desarrolla solo sobre esa vecindad. Generalmente el número de vecinos a buscar están fijados de antemano[27].

Dentro de los algoritmos que se pueden incluir en este apartado están el DANN de Hastie et al. que se expondrá con mayor detalle en los trabajos de referencia para esta tesis (sección 4.4.2), y el algoritmo ADAMENN [130], predecesor del LaMaNNa que se comentará con detalle en la sección 4.4.4 de esta tesis. Y también la distorsión del espacio $\mathcal{X}$ en función de la abundancia de casos en torno al que se quiere clasificar, propuesta por Chang et al. en [131] y que se explica con detalle en la sección 4.4.6.

Algoritmos locales no "lazy"

En este caso se determina "a priori" en el espacio original $\mathcal{X}$ una vecindad para cada caso y se ajusta una métrica apropiada para esa vecindad. El conjunto de todas las métricas aprendidas[28] se puede considerar como una métrica local.

Quizás el algoritmo de aprendizaje supervisado más representativo sea la variante local del LMNN conocida como "Multi-metric LMNN" (MM-LMNN) [128], donde mediante una técnica de "clustering" se divide el espacio $\mathcal{X}$ en un conjunto de agrupaciones, ajustando una métrica para cada una de ellas.

---

[27] Pero también se puede adaptar ese número al contexto de los datos de entrenamiento, incorporando en el aprendizaje una etapa previa de cálculo de ese parámetro.

[28] Que son métricas basadas en matrices formadas por valores escalares.

4.4 Trabajos de referencia en el estado del arte

En esta amplia sección se van a exponer aquellos trabajos punteros de investigación que han tenido más relevancia en el desarrollo de esta tesis.

De la sección 4.4.1 a la 4.4.5 se revisa, de más antiguo a más moderno, el uso de medidas de distancias globales o locales que intentan mejorar la precisión de los algoritmos de clasificación. Y en las secciones 4.4.6 y 4.4.7 se exponen enfoques avanzados sobre el uso de "*kernels*" que proporcionen una métrica local de tipo riemanniana.

4.4.1 Algoritmo VSM ("Similarity metric learning for a variable-kernel classifier")

Es el más antiguo de todos los trabajos de referencia para esta tesis. En 1995, David Lowe presentó un artículo [55] que abría una nueva puerta[29] en la clasificación en función de los vecinos más próximos (algoritmo *k*-NN). Aun reconociendo su buen comportamiento en las tareas de clasificación, Lowe criticaba su escasa capacidad de generalización cuando aumentaba el número de dimensiones del espacio de atributos o cuando estos no eran relevantes.

4.4.1.1 Objetivo

Reducir el error de generalización de un clasificador *k*-NN en problemas con atributos irrelevantes.

4.4.1.2 Enfoque

En el método tradicional de clasificación *k*-NN, los vecinos más próximos a una nueva instancia $\boldsymbol{x}_m$ a clasificar se determinan basándose en una métrica euclídea, en cuyo caso todas las dimensiones contribuyen uniformemente.

Mediante el algoritmo VSM el problema de clasificar se plantea en dos etapas: etapa de entrenamiento y etapa de predicción. El entrenamiento consiste en el aprendizaje de una métrica de tipo Mahalanobis basada en una matriz diagonal $\boldsymbol{W} = diag\{ w_a \}$, $a = 1,2, \dots, A$. Sus elementos se determinan minimizando una función de error cuadrático entre la probabilidad estimada de que el caso $\boldsymbol{x}_m$ pertenezca a la clase *j*, $p_j(\boldsymbol{x}_m)$ y la probabilidad contrastada de pertenecer a dicha clase $c_j(\boldsymbol{x}_m) = I(y_m = y_j)$.

El problema de minimización se aborda mediante una optimización sin restricciones y se resuelve aplicando el algoritmo PR-CG (Gradiente Conjugado, algoritmo de Polak-Ribiere), ver sección 3.1.1.2.

[29] Es posible rastrear técnicas precursoras tanto en el aprendizaje de las redes neuronales mediante el algoritmo de "backpropagation" [38], como en el libro de Silverman [151].

Para estimar la probabilidad "a posteriori" $p(y_j|\boldsymbol{x}_m)$ se selecciona un subconjunto relativamente pequeño (k_m) de vecinos próximos[30] a $\boldsymbol{x}_m$ y se calcula el cociente entre la suma de similitudes entre ese caso y los casos vecinos que pertenecen a la clase *j*, respecto a la suma de similitudes entre ese caso y la totalidad de los k_m de vecinos próximos.
Así pues, la probabilidad "a posteriori" de clasificar el caso $\boldsymbol{x}_m$ en la clase *j* será:

$$p(y_j|\boldsymbol{x}_m) = \frac{\sum_{i=1}^{k_m} sim(\boldsymbol{x}_m, \boldsymbol{x}_i)\, c_j(\boldsymbol{x}_i)}{\sum_{i=1}^{k_m} sim(\boldsymbol{x}_m, \boldsymbol{x}_i)} \tag{4.59}$$

donde:
$c_j(\boldsymbol{x}_i)$: es un valor que será 1 o 0 en función de que el caso *i*-ésimo pertenezca realmente o no a la clase *j*.
$sim(\boldsymbol{x}_m, \boldsymbol{x}_i)$: es la medida de similitud entre el caso *m* y el *i* (el caso *i* será uno de los k_m vecinos más próximos de *m*; obviamente $m \neq i$).

Dicha medida de similitud se calcula, partiendo de la norma ponderada de la diferencia entre los atributos de los casos, mediante una función de tipo "*kernel*" (en este caso gaussiano):

$$sim(\boldsymbol{x}_m, \boldsymbol{x}_i) = e^{-\frac{\|\boldsymbol{x}_m - \boldsymbol{x}_i\|^2_{\boldsymbol{W}}}{2\sigma^2}} = e^{-\left(d^2_{\boldsymbol{W}}(\boldsymbol{x}_m, \boldsymbol{x}_i)/2\sigma^2\right)} \tag{4.60}$$

donde $d^2_{\boldsymbol{W}}(\boldsymbol{x}_m, \boldsymbol{x}_i)$ es la distancia ponderada de acuerdo a la matriz $\boldsymbol{W}$ entre los casos *m*-ésimo e *i*-ésimo, que es calculada mediante:

$$\|\boldsymbol{x}_m - \boldsymbol{x}_i\|^2_{\boldsymbol{W}} = d^2_{\boldsymbol{W}}(\boldsymbol{x}_m, \boldsymbol{x}_i) = \sum_{a=1}^{A} w_a \left(x_{m,a} - x_{i,a}\right)^2 \tag{4.61}$$

donde:
w_a : es el peso asignado al atributo *a*.
$x_{m,a}$ y $x_{i,a}$: son los valores de los *a*-ésimos atributos para los casos *m* e *i*.

El valor σ en la fórmula (4.60) es un parámetro relacionado con anchura (radio) del "*kernel*" gaussiano empleado para establecer la similitud entre casos. Lowe propone estimarlo en función de los 5 vecinos más próximos mediante una expresión (que posee otro parámetro a estimar u optimizar, *r*) como:

$$\sigma = \frac{r}{5} \sum_{k=1}^{5} d_{\boldsymbol{W}}(\boldsymbol{x}_m, \boldsymbol{x}_k) \tag{4.62}$$

[30] Aunque el método, en general, se puede aplicar a cualquier número de vecinos, Lowe recomienda emplear el valor de 10 (y es el valor que seguiremos en esta explicación).

El objetivo del algoritmo VSM es minimizar una función de error cuadrático que acumula, a lo largo de todos los casos de aprendizaje, las discrepancias que se producen entre la probabilidad calculada de que un caso pertenezca a una determinada clase y la probabilidad constatada de pertenecer a esa clase (que obviamente será 1 o 0):

$$f_{err}(\boldsymbol{W}, r) = \sum_{m=1}^{N} \sum_{j=1}^{J} \left(c_j(\boldsymbol{x}_m) - p\left(y_j|\boldsymbol{x}_m\right) \right)^2 \tag{4.63}$$

Para minimizar esta expresión de error se emplea una estrategia basada en el gradiente conjugado (técnica de Polak-Ribiere) y en una validación cruzada de tipo LOO (valorando el error cometido al estimar la clase de un elemento empleando los *N*-1 casos restantes). El resultado final serán los valores de los pesos $\boldsymbol{W} = \{ w_a \}, \ \ a = 1,2, \dots, A$ y el valor del parámetro *r*.

Para poder aplicar el método de optimización mediante el gradiente conjugado son necesarias las derivadas parciales de la función de error respecto a los pesos w_a y a *r*, estas serán:

$$\begin{aligned} \frac{\partial f_{err}(\boldsymbol{W}, r)}{\partial w_a} &= 2 \sum_{n=1}^{N} \sum_{j=1}^{J} \left(p\left(y_j|\boldsymbol{x}_m\right) \ - c_j(\boldsymbol{x}_m) \right) \frac{\partial p\left(y_j|\boldsymbol{x}_m\right)}{\partial w_a} \\ \frac{\partial f_{err}(\boldsymbol{W}, r)}{\partial r} &= 2 \sum_{n=1}^{N} \sum_{j=1}^{J} \left(p\left(y_j|\boldsymbol{x}_m\right) \ - c_j(\boldsymbol{x}_m) \right) \frac{\partial p\left(y_j|\boldsymbol{x}_m\right)}{\partial r} \end{aligned} \tag{4.64}$$

donde:

$$\begin{aligned} \frac{\partial p\left(y_j|\boldsymbol{x}_m\right)}{\partial w_a} &= \frac{\sum_{i=1}^{10} \left(c_j(\boldsymbol{x}_i) - p\left(y_j|\boldsymbol{x}_m\right) \right) \frac{\partial sim(\boldsymbol{x}_m, \boldsymbol{x}_i)}{\partial w_a}}{\sum_{i=1}^{10} sim(\boldsymbol{x}_m, \boldsymbol{x}_i)} \\ \frac{\partial p\left(y_j|\boldsymbol{x}_m\right)}{\partial r} &= \frac{\sum_{i=1}^{10} \left(c_j(\boldsymbol{x}_i) - p\left(y_j|\boldsymbol{x}_m\right) \right) \frac{\partial sim(\boldsymbol{x}_m, \boldsymbol{x}_i)}{\partial r}}{\sum_{i=1}^{10} sim(\boldsymbol{x}_m, \boldsymbol{x}_i)} \end{aligned} \tag{4.65}$$

y:

$$\begin{aligned} \frac{\partial sim(\boldsymbol{x}_m, \boldsymbol{x}_i)}{\partial w_a} = \\ = -\frac{sim(\boldsymbol{x}_m, \boldsymbol{x}_i)\left(x_{m,a} - x_{i,a}\right)^2}{2\sigma^2} + \\ + \frac{sim(\boldsymbol{x}_m, \boldsymbol{x}_i) d_{\boldsymbol{W}}^2(\boldsymbol{x}_m, \boldsymbol{x}_i) \frac{r}{5} \sum_{k=1}^{5} \frac{\left(x_{m,a} - x_{k,a}\right)^2}{d_{\boldsymbol{W}}(\boldsymbol{x}_m, \boldsymbol{x}_k)}}{2\sigma^3} \end{aligned} \tag{4.66}$$

$$\frac{\partial sim(\boldsymbol{x}_m, \boldsymbol{x}_i)}{\partial r} = \ sim(\boldsymbol{x}_m, \boldsymbol{x}_i) \frac{d_{\boldsymbol{W}}^2(\boldsymbol{x}_m, \boldsymbol{x}_i)}{r\sigma^2}$$

4.4.1.3 Complejidad computacional

Se parte de que no se pretende optimizar el valor del parámetro k_m, tomando como bueno el valor de 10 propuesto por Lowe. En caso contrario habría que plantear otro problema de validación cruzada para determinar este valor. En este nuevo escenario, la complejidad se multiplicaría por el número de valores enteros que se intentasen probar.

Se tendrán que efectuar un número desconocido[31] de iteraciones del algoritmo PR-CG antes de encontrar el mínimo del error. En cada iteración se tendrán que calcular (en los dos primeros pasos se utiliza la técnica LOO):

- Los k_m vecinos próximos de cada uno de los N casos y con ellos calcular las $p(y_j|\boldsymbol{x}_m)$ para las J clases.
- Calcular la función de error y almacenar su valor.
- Calcular las derivadas de la función de error respecto a los A pesos y respecto al parámetro r, se almacenarán estos A+1 valores.
- Aplicar un paso del algoritmo PR-CG.

Todo ello no representa grandes requisitos ni de memoria ni de velocidad de cálculo, por lo que el algoritmo se puede escalar fácilmente a problemas con muchos atributos y con una muestra de tamaño medio.

4.4.1.4 Ventajas e inconvenientes

El empleo del algoritmo VSM para resolver un problema de clasificación permite escalar las dimensiones individuales del espacio de atributos. Grandes factores de escala se corresponden con atributos que portan mucha información en relación con la separabilidad de las clases.

Frente a los algoritmos que optimizan todos los elementos de una matriz completa de la métrica, este método tiene como principal ventaja que solo debe ajustar los parámetros de la diagonal principal (los pesos de cada dimensión) y la anchura del "*kernel*", lo cual le hace más robusto frente a los problemas de sobreaprendizaje y, de acuerdo a la teoría de Vapnik, su riesgo estructural está más limitado.

Como se deduce de la fórmula (4.60)(4.63), el algoritmo VSM establece una medida de distancia global, ya que los pesos asignados a los distintos atributos toman el mismo valor en todo el espacio de atributos.

Muchos otros autores han estudiado generalizaciones del método de Lowe, digna de mencionar es aquella que en vez de limitarse a optimizar los valores de la diagonal principal de la matriz empleada para calcular las distancias, optimizan la matriz completa de una métrica cuadrática [132].

---

[31] Probablemente no muy alto.

Los resultados de Lowe superan a los de las técnicas habituales *k*-NN y en muchos casos también a los obtenidos con redes neuronales de propagación hacia atrás o con RBFs.

Al igual que se comentará para el algoritmo LaMaNNa, el algoritmo VSM solo es efectivo en aquellos problemas en los que las fronteras de separación entre clases siguen los ejes coordenados. VSM solo puede aumentar o disminuir la relevancia de los atributos (por medio de los pesos) en dimensiones paralelas a los ejes coordenados (en la Figura 4.11 se muestra un problema similar) y no tiene en cuenta las correlaciones que pudieran existir entre atributos.

Además se deben sintonizar A+1 parámetros, lo que puede ser complejo en la práctica en aquellos problemas en que el ratio entre el número de casos de aprendizaje y el número de dimensiones no sea relativamente elevado.

4.4.2 Algoritmo DANN ("Discriminant Adaptive Nearest Neighbor Classification")

Hastie y Tibshirani en [61] proponen emplear una métrica local que intente paliar los efectos de la maldición de la dimensionalidad en los algoritmos de clasificación mediante la búsqueda de vecinos próximos.
Para ello abandonan la métrica euclídea y la sustituyen por otra "orientada". Para obtener la dirección de la nueva métrica se inspira en el tradicional algoritmo del análisis discriminante lineal (LDA), pero aplicado en este caso de una forma local en el entorno del punto que se desea clasificar.

4.4.2.1 Objetivo

El aprendizaje de una métrica local de tipo Mahalanobis mediante un procedimiento recursivo usando los k_m vecinos más próximos de un caso $\boldsymbol{x}_m$ a clasificar. Se aplican las siguientes hipótesis:

- Las probabilidades "a posteriori" de las clases son casi constantes en las inmediaciones del caso $\boldsymbol{x}_m$.
- Las distribuciones espaciales de los casos dentro de la vecindad de un determinado caso son de tipo normal multivariable y poseen la misma matriz de covarianzas.
- Se considera que los atributos medidos de los casos están exentos de ruido.

4.4.2.2 Enfoque

Se busca una métrica que se adapte a la región local de un nuevo caso $\boldsymbol{x}_m$ a clasificar de manera que no varíen apreciablemente las probabilidades "a posteriori" de pertenecer a las distintas clases. El mecanismo de adaptación consiste en deformar la geometría de esa región, contrayéndola en dirección de máxima variación entre clases y alargándola en su plano perpendicular (esto es fundamental ya que las probabilidades de pertenecer a una u otra clase varían fuertemente es las proximidades de la frontera entre estas).
Formalmente, una vez seleccionados los k_m vecinos próximos de un caso $\boldsymbol{x}_m$, se diseña mediante un proceso iterativo la siguiente métrica:

$$d^2_{\boldsymbol{\Sigma}_D}(\boldsymbol{x}, \boldsymbol{x}_m) = (\boldsymbol{x} - \boldsymbol{x}_m)^T \, \boldsymbol{M}_D \, (\boldsymbol{x} - \boldsymbol{x}_m)$$

con: (4.67)

$$\boldsymbol{M}_D = \boldsymbol{W}^{-1} \boldsymbol{B} \, \boldsymbol{W}^{-1}$$

donde $\boldsymbol{W}$ es la matriz de covarianzas intraclase (que se calcula mediante la media ponderada de las matrices de covarianza de las clases usando los k_m vecinos próximos de $\boldsymbol{x}_m$). Y $\boldsymbol{B}$ es la matriz de covarianza interclases, calculada con esos mismos casos.

$$\boldsymbol{W} = \frac{1}{k_m}\sum_{j=1}^{J} k_{m_j}\ \boldsymbol{W}_j\ , \qquad \boldsymbol{W}_j = \frac{\sum_{n=1}^{k_{m_j}} (\boldsymbol{x}_n - \boldsymbol{\mu}_j)\,(\boldsymbol{x}_n - \boldsymbol{\mu}_j)^T}{k_{m_j}} \tag{4.68}$$

donde $\boldsymbol{x}_n$ es un vector con los valores de los atributos del caso n-ésimo (dentro de los k_m vecinos), k_{m_j} la cardinalidad del conjunto de esos casos que pertenecen a la clase j, y $\boldsymbol{\mu}_j$ el vector con los valores medios de los atributos de los casos para ese mismo conjunto.

De forma similar se calcula la matriz de covarianzas entre clases $\boldsymbol{B}$ (también promediada):

$$\boldsymbol{B} = \frac{\sum_{j=1}^{J}\ k_{m_j}\,(\boldsymbol{\mu}_j - \bar{\boldsymbol{\mu}})\,(\boldsymbol{\mu}_j - \bar{\boldsymbol{\mu}})^T}{k_m} \tag{4.69}$$

donde $\bar{\boldsymbol{\mu}}$ es el vector con la media de los atributos de estos casos (independientemente de a qué clase pertenezcan).

A continuación, se determinan las direcciones discriminantes, y se muestra que la región de los k_m vecinos próximos a $\boldsymbol{x}_m$ se alarga en la dirección de las coordenadas no discriminantes, las cuales corresponden a los ejes de coordenadas del espacio nulo de la matriz de covarianza interclases definida al proyectar $(\boldsymbol{x} - \boldsymbol{x}_m)$ sobre un espacio cuyos ejes principales están definidos por $\boldsymbol{W}^{-1/2}$.

Al tener en cuenta la invarianza de la matriz de covarianzas en una proyección ortogonal, $\boldsymbol{M}_D$ puede escribirse como sigue:

$$\begin{aligned}\boldsymbol{M}_D = \boldsymbol{W}^{-1}\,\boldsymbol{B}\ \boldsymbol{W}^{-1} = \boldsymbol{W}^{-1/2}\big(\boldsymbol{W}^{-1/2}\,\boldsymbol{B}\ \boldsymbol{W}^{-1/2}\big)\boldsymbol{W}^{-1/2} = \\ = \boldsymbol{W}^{-1/2}\,\boldsymbol{B}^*\,\boldsymbol{W}^{-1/2}\end{aligned} \tag{4.70}$$

donde $\boldsymbol{B}^*$ es la matriz de covarianza entre clases en el espacio transformado esféricamente.

Al descomponer $\boldsymbol{B}^*$ espectralmente:

$$\boldsymbol{B}^* = \boldsymbol{V}^*\boldsymbol{\Lambda}_B\,\boldsymbol{V}^{*T} = \sum_{i=1}^{A} \lambda_i \boldsymbol{v}_i \boldsymbol{v}_i{}^T \tag{4.71}$$

resulta:

$$\begin{aligned}d^2_{\boldsymbol{M}_D}(\boldsymbol{x}, \boldsymbol{x}_m) = \left[\boldsymbol{W}^{-1/2}\,(\boldsymbol{x} - \boldsymbol{x}_m)\right]^T\ \boldsymbol{V}^*\,\boldsymbol{\Lambda}_B\,\boldsymbol{V}^{*T}\left[\boldsymbol{W}^{-1/2}\,(\boldsymbol{x} - \boldsymbol{x}_m)\right] = \\ = \left[\boldsymbol{V}^{*T}\boldsymbol{W}^{-1/2}\,(\boldsymbol{x} - \boldsymbol{x}_m)\right]^T\,\boldsymbol{\Lambda}_B\ \left[\boldsymbol{V}^{*T}\boldsymbol{W}^{-1/2}\,(\boldsymbol{x} - \boldsymbol{x}_m)\right]\end{aligned} \tag{4.72}$$

Por tanto, las direcciones correspondientes a los valores propios nulos de $\boldsymbol{B}$ no contribuyen al valor de $d^2_{\boldsymbol{M}_D}(\boldsymbol{x}, \boldsymbol{x}_m)$, mientras que son discriminantes las del subespacio engendrado por los vectores columnas de $\boldsymbol{W}^{-1/2}\,\boldsymbol{V}^*$.

Desde el punto de vista de adaptación local de la región de vecinos a un caso dado, esto significa que se puede alargar en la dirección de los ejes del subespacio complemento de dicha región y contraer en la dirección de los ejes del subespacio engendrado por las columnas de $\boldsymbol{W}^{-1/2}\,\boldsymbol{V}^*$.

Para evitar el alargamiento indefinido de esa región, se recomienda regularizar la matriz $\boldsymbol{B}^*$ sumándole una matriz escalar, esto es:

$$\boldsymbol{M}_{DR} = \boldsymbol{W}^{-1/2}\,(\boldsymbol{B}^* + \epsilon\,\boldsymbol{I})\,\boldsymbol{W}^{-1/2} \tag{4.73}$$

La regularización evita el uso de casos muy lejanos (en el sentido de la distancia euclídea) a $\boldsymbol{x}_m$.

En la primera de las variantes del algoritmo DANN, una vez obtenida esta matriz $\boldsymbol{M}_{DR}$ el procedimiento se da por terminado; en una segunda versión, empleando esta matriz como métrica para las distancias (en vez de la métrica euclídea), se procede a una nueva iteración para calcular los casos más cercanos, procediendo de nuevo a evaluar las ecuaciones (4.68) y ss.; y así hasta que estabilice el conjunto de casos vecinos obtenido.

Cuando se obtiene la convergencia se dan por terminadas las iteraciones y se emplea la fórmula $d_{\boldsymbol{M}_{DR}}(\boldsymbol{x},\boldsymbol{x}_m) = \sqrt{(\boldsymbol{x}-\boldsymbol{x}_m)^T\,\boldsymbol{M}_{DR}\,(\boldsymbol{x}-\boldsymbol{x}_m)}$ para calcular la distancia entre el caso dado y sus vecinos más próximos; empleándose finalmente la técnica *k*-NN para obtener la clase del caso dado.

4.4.2.3 Interpretación de la distancia DANN como una distancia χ^2

La forma adoptada para $\boldsymbol{M}_D$ tiene una interpretación estadística basada en las dos primeras hipótesis. En efecto, al manipular algebraicamente la expresión de la distancia Chi-cuadrado:

$$d^2_{\chi^2}(\boldsymbol{x},\boldsymbol{x}_m) = \sum_{\boldsymbol{j=1}}^{\boldsymbol{J}} \frac{\big(p(j|\boldsymbol{x}) - p(j|\boldsymbol{x}_m)\big)^{\boldsymbol{2}}}{p(j|\boldsymbol{x}_m)} \tag{4.74}$$

aproximando $p(j|\boldsymbol{x})$ por el término lineal de su desarrollo en serie de Taylor en un entorno de $p(j|\boldsymbol{x}_m)$:

$$p(j|\boldsymbol{x}) \cong p(j|\boldsymbol{x}_m) - p(j|\boldsymbol{x}_m)\big(\boldsymbol{\mu}_j - \overline{\boldsymbol{\mu}}\big)^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{x}_m) \tag{4.75}$$

donde $\boldsymbol{\mu}_j$ y $\boldsymbol{\Sigma}$ representan el valor medio y la matriz de covarianzas de cada clase, $\overline{\boldsymbol{\mu}} = \sum_{\boldsymbol{j=1}}^{\boldsymbol{J}} p(j|\boldsymbol{x}_m)\,\boldsymbol{\mu}_j$ es el valor medio de los atributos del conjunto de los k_m vecinos, y $p(j|\boldsymbol{x})$ y $p(j|\boldsymbol{x}_m)$ son las probabilidades "a posteriori" de pertenecer $\boldsymbol{x}$ y $\boldsymbol{x}_m$ a la clase j.

Así pues, resulta:

$$\begin{aligned} d^2_{\chi^2}(\boldsymbol{x},\boldsymbol{x}_m) &= (\boldsymbol{x}-\boldsymbol{x}_m)^T\boldsymbol{\Sigma}^{-1}\left[\sum_{j=1}^{J} p(j|\boldsymbol{x}_m)\big(\boldsymbol{\mu}_j-\overline{\boldsymbol{\mu}}\big)\big(\boldsymbol{\mu}_j-\overline{\boldsymbol{\mu}}\big)^T\right]\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{x}_m) = \\ &= \left[\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{x}_m)\right]^T\left[\sum_{j=1}^{J} p(j|\boldsymbol{x}_m)\big(\boldsymbol{\mu}_j-\overline{\boldsymbol{\mu}}\big)\big(\boldsymbol{\mu}_j-\overline{\boldsymbol{\mu}}\big)^T\right]\left[\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{x}_m)\right] \end{aligned} \tag{4.76}$$

Esta expresión es formalmente igual a la de la ecuación (4.67) si se realizan las siguientes sustituciones:

$$\boldsymbol{\Sigma}^{-1} \leftarrow \boldsymbol{W}^{-1}$$
$$\sum_{j=1}^{J} p(j|\boldsymbol{x}_m)(\boldsymbol{\mu}_j - \bar{\boldsymbol{\mu}})\,(\boldsymbol{\mu}_j - \bar{\boldsymbol{\mu}})^T \leftarrow \boldsymbol{B} \tag{4.77}$$

Por otra parte, obsérvese que $\boldsymbol{B}$ determina las direcciones de mayor variación de la probabilidad de pertenecer a una u otra clase habiendo realizado la transformación lineal defina por $\boldsymbol{W}^{-1}$. De ahí que el mecanismo de adaptación pueda interpretarse geométricamente como un modelado local vía el algoritmo LDA.

4.4.2.4 Complejidad computacional

El algoritmo DANN se puede clasificar entre los métodos de clasificación de aprendizaje demorado ("lazy learning").

Los autores sugieren emplear un valor de k_m bastante mayor que el empleado para el parámetro *k* de *k*-NN; apuntan heurísticamente a emplear un valor tal como $k_m = max(N/5\,, 50)$. Tampoco dan una orientación para el valor de ϵ (aparte de que es un valor pequeño).

Para un caso nuevo a clasificar $\boldsymbol{x}_m$:

- Se calculan sus k_m vecinos próximos.
- Con ellos se estiman las matrices $\boldsymbol{W}$ y $\boldsymbol{B}$.
- Se calcula la matriz de la métrica: $\boldsymbol{M}_D = \boldsymbol{W}^{-1/2}\,(\boldsymbol{B}^* + \epsilon\,\boldsymbol{I})\,\boldsymbol{W}^{-1/2}$.
- Y con ella la distancia del punto a los k_m vecinos próximos mediante $d_{\boldsymbol{M}_D}(\boldsymbol{x}, \boldsymbol{x}_m) = \sqrt{(\boldsymbol{x} - \boldsymbol{x}_m)^T\,\boldsymbol{M}_D\,(\boldsymbol{x} - \boldsymbol{x}_m)}$
- Con estas distancias se clasifica el caso de acuerdo al algoritmo *k*-NN.
- Si se desea emplear la versión iterativa del algoritmo, se calcularían de nuevo los k_m vecinos más próximos (de acuerdo a la nueva métrica) y se ejecutarían de nuevo todos los pasos antes enumerados hasta lograr que la clasificación del caso no varíe entre iteraciones.

Todo ello no representa grandes requisitos ni de memoria $A * (A + 1)/2$ para la matriz $\boldsymbol{W}$) ni de velocidad de cálculo (el mayor problema es la diagonalización de la matriz $\boldsymbol{W}$ para obtener sus valores y vectores propios), salvando esta limitación, el algoritmo se puede escalar fácilmente a problemas con muchos atributos y con una muestra de gran tamaño.

Si se deseasen estimar los valores óptimos de k_m y ϵ, se debería emplear una estrategia de validación cruzada y el tiempo de cálculo anterior se vería multiplicado por el número de parejas de valores seleccionados.

4.4.2.5 Ventajas e inconvenientes

DANN se apoya en varios supuestos (los cuales no suelen ser ciertos en los casos prácticos).

El primero es que considera que los casos próximos a uno dado presentan distribuciones espaciales de tipo normal multivariable con la misma matriz de covarianzas (para poder obtener distribuciones de tipo esférico al aplicarles la transformación $\boldsymbol{W}^{-1/2}$). Habitualmente esto no será cierto, aunque sí es verdad que el algoritmo de clasificación LDA (que también necesita de este requisito) se comporta bastante correctamente cuando no se cumple.

El segundo es que tampoco tienen en cuenta el ruido inherente a las medidas obtenidas en casos reales. Esto hace que para casos relativamente próximos la dirección de separación entre clases pueda variar ostensiblemente. Los propios autores lo reconocen y proponen emplear un número de vecinos k_m mucho más grande a la hora de calcular las matrices $\boldsymbol{W}$ y $\boldsymbol{B}$, que el que se utilizará posteriormente para dilucidar la clase del caso dado mediante la técnica *k*-NN. Aun así, la estimación de la métrica será poco robusta.

Por último, este parámetro k_m, y el valor de ϵ de la fórmula (4.73) deben ser propuestos heurísticamente sin más, o bien ser ajustados mediante validación cruzada.

4.4.3 Algoritmo LFM-SVM ("Local Flexible Metric classification based on SVMs")

El algoritmo LFM-SVM fue presentado por Carlotta Domeniconi y sus colaboradores en el año 2002 [103]. Entre sus aportaciones más importantes hay que destacar:

- Identificaba nítidamente la necesidad de abandonar la métrica euclídea y sustituirla por una métrica local que alargase el área de búsqueda de vecinos próximos de forma paralela a la frontera de separación de clases, estrechándola en la dirección perpendicular a esta.
- Otra de sus propuestas era reemplazar los enfoques de tipo aprendizaje demorado, predominantes en aquella época, por un sistema de pesos calculados de antemano. Para ello, este algoritmo introdujo en el diseño de la métrica el uso de un conocimiento "a priori": una curva de separación entre clases precalculada mediante una SVM (un enfoque similar se puede encontrar ya expuesto en los trabajos de Amari y Wu [133]).
- Otro enfoque muy interesante era el uso del gradiente como herramienta para conocer la dirección perpendicular a la superficie de separación entre clases.

Tres años después de la publicación de este artículo, los mismos autores dieron a conocer una versión extendida de este algoritmo, al que le llamaron LaMaNNa. Las características de ambos algoritmos son muy similares, así que para evitar la duplicidad en las explicaciones, su exposición conjunta se acometerá en la siguiente sección.

4.4.3.1 Ventajas e inconvenientes

Al poder considerarse una versión previa del algoritmo LaMaNNa, sus bondades e inconvenientes se discutirán con las de este.

4.4.4 Algoritmo LaMaNNa ("Large Margin Nearest Neighbor classifiers")

Este nuevo trabajo de Domeniconi y sus colaboradores [63][32] continúa y completa los desarrollos del algoritmo citado en la sección anterior [103] y los conecta con los de otra investigación similar firmada por uno de sus colaboradores [105].

4.4.4.1 Objetivo

Diseñar una medida de distancia local que pondere los pesos de los distintos atributos en relación con una función de decisión conocida previamente:

$$d^2_{\boldsymbol{M}_L}(\boldsymbol{x}, \boldsymbol{x}_m) = (\boldsymbol{x} - \boldsymbol{x}_m)^T \, \boldsymbol{M}_L \, (\boldsymbol{x} - \boldsymbol{x}_m)$$

con:

$$\boldsymbol{M}_L = diag\{\, w_a(\boldsymbol{x}_m) \,\}, \quad a = 1,2, \dots, A \tag{4.78}$$

donde los pesos $w_a(\boldsymbol{x}_m)$ dependen de la posición en el espacio de atributos donde se encuentra el nuevo caso a clasificar.

4.4.4.2 Enfoque

Domeniconi basa su medida de distancia en el conocimiento previo de la función de separación entre clases (y de su gradiente). En el caso de LaMaNNa, esta función de decisión procede de una SVM entrenada con anterioridad[33].

Así pues, el algoritmo LaMaNNa parte de estimar, con los propios datos de aprendizaje, la superficie de separación entre clases $f(\boldsymbol{x}) = 0$, por medio de una SVM (ver una explicación detallada sobre el aprendizaje mediante SVM en la sección 4.2.3).

$$f_{SVM}(\boldsymbol{x}) = \sum_{i=1}^{nSV} \alpha_i y_i K\left(\boldsymbol{x}, \boldsymbol{x}_{sv_i}\right) + b \tag{4.79}$$

Una vez que se le presenta un nuevo caso a clasificar $\boldsymbol{x}_m$, en primer lugar, busca el punto más próximo $\boldsymbol{x}_f$ sobre la superficie de separación de clases $f(\boldsymbol{x}) = 0$.

$$\boldsymbol{x}_f = \arg\min_{\boldsymbol{x}} \|\boldsymbol{x} - \boldsymbol{x}_m\|, \quad \text{sujeto a } f_{SVM}(\boldsymbol{x}) = 0 \tag{4.80}$$

---

[32] Este algoritmo representó una de las influencias iniciales más importantes para la métrica LOM.

[33] Se podría considerar como un conocimiento "a priori" si no fuese porque la función de decisión ha sido calculada también mediante los casos de aprendizaje.

En este punto $(\boldsymbol{x}_f)$, se calcula el gradiente de la superficie $f_{SVM}(\boldsymbol{x}) = 0$, esto es:

$$\boldsymbol{n}_{x_f} = \nabla f_{SVM}(\boldsymbol{x}_f) / \left\| \nabla f_{SVM}(\boldsymbol{x}_f) \right\| \tag{4.81}$$

A continuación, expresa la relevancia relativa de cada dimensión en el espacio de atributos en función de la dirección de ese gradiente. Para ello, simplemente, proponen tomar el valor absoluto de cada componente del gradiente.

$$R_a(\boldsymbol{x}_m) = \left| \boldsymbol{n}_{x_f,a} \right| \tag{4.82}$$

donde $R_a(\boldsymbol{x}_m)$ es la relevancia relativa en el atributo *a*-ésimo.

Y, por último, calcula una ponderación para cada una de las dimensiones del espacio de atributos mediante la siguiente fórmula:

$$w_a(\boldsymbol{x}_m) = \frac{e^{C.R_a(\boldsymbol{x}_m)}}{\sum_{i=1}^{A} e^{C.R_i(\boldsymbol{x}_m)}} \tag{4.83}$$

donde C es un parámetro que se calcula en función del vector de soporte más cercano y la distancia media de los casos de aprendizaje a la frontera de separación entre clases.

Para terminar, la distancia entre dos puntos se calcula mediante una métrica ponderada de acuerdo a la siguiente expresión:

$$d(\boldsymbol{x}, \boldsymbol{x}_m) = \sqrt{\sum_{a=1}^{A} w_a(\boldsymbol{x}_m)\left(x_a - x_{m,a}\right)^2} \tag{4.84}$$

4.4.4.3 Complejidad computacional

El algoritmo LaMaNNa se puede clasificar entre los métodos de clasificación de aprendizaje demorado ("lazy learning"); aunque parte del aprendizaje (función de decisión basada en una SVM) se haya calculado previamente.

Una vez conocida la función de decisión $f_{SVM}(\boldsymbol{x}) = 0$, para clasificar un caso nuevo $\boldsymbol{x}_m$ se debe:

- Mediante un procedimiento iterativo se encuentra el punto más cercano a $\boldsymbol{x}_m$ sobre la función de decisión: $\boldsymbol{x}_f$.
- En ese punto se calcula el gradiente $\boldsymbol{n}_{x_f}$, bien analíticamente (obteniendo las derivadas parciales de la ecuación (4.79) respecto a los distintos atributos), o bien de forma empírica (midiendo las variaciones de la función de decisión al incrementar infinitesimalmente cada uno de los atributos).
- Se obtienen las distintas relevancias $R_a(\boldsymbol{x}_m)$ y de ahí los pesos $w_a(\boldsymbol{x}_m)$.

- Con ello ya se puede calcular la distancia de ese punto a cualquier otro del espacio de atributos mediante $d_{\boldsymbol{M}_L}(\boldsymbol{x}, \boldsymbol{x}_m) = \sqrt{(\boldsymbol{x} - \boldsymbol{x}_m)^T \, \boldsymbol{M}_L \, (\boldsymbol{x} - \boldsymbol{x}_m)}$
- Con estas distancias se clasifica el caso de acuerdo al algoritmo *k*-NN.

Los requisitos de memoria son mínimos (se deberán almacenar los SV de la SVM y vectores de dimensión *A* para los gradientes, relevancias y pesos de la matriz de la distancia); tampoco se requiere gran velocidad de cálculo (el mayor problema es la búsqueda del punto más próximo sobre la superficie de decisión), salvando esta limitación, el algoritmo se puede escalar fácilmente a problemas con muchos atributos y con una muestra de gran tamaño.

4.4.4.4 Ventajas e inconvenientes

Una ventaja importante del algoritmo LaMaNNa es su simplicidad y los escasos recursos computacionales que requiere.

Pero las críticas son varias, la principal es que este algoritmo no conduce a una métrica, ya que no se satisfacen los axiomas de simetría ni de desigualdad triangular.

Buscar el punto más próximo a uno dado, que esté sobre la superficie de separación de clases, es una tarea que requiere de un procedimiento iterativo (los autores proponen avanzar desde el punto $\boldsymbol{x}_m$ con un paso arbitrariamente pequeño e ir duplicándolo hasta que se cruce la frontera entre clases[34]) y que complica mucho (por no decir, imposibilita) el análisis matemático de la solución propuesta.

Los autores tampoco aportan en ninguno de los dos artículos ninguna prueba que justifique el tener que utilizar un punto de dicha superficie para calcular el gradiente. En un entorno cercano a la superficie de separación de las clases, las direcciones de los gradientes de las distintas curvas de nivel serán similares a los encontrados en la propia $f_{SVM}(\boldsymbol{x}) = 0$. Según nos alejemos de la frontera entre clases, adoptar una dirección correcta para el gradiente tiene mucha menos importancia, ya que en esta zona del espacio de atributos la métrica euclídea se comportaría también de forma correcta.

Este algoritmo es efectivo en aquellos problemas en los que las fronteras de separación entre clases siguen los ejes coordenados. Al igual que en el algoritmo VSM (ver sección 4.4.1), no pueden inclinar los ejes principales del "vecindario" que equidista de un punto dado. Por ejemplo, para un problema bidimensional como el de la Figura 4.11, la distancia que se mediría con el algoritmo LaMaNNa desde el punto $\boldsymbol{x}_m$ sería la misma que la que obtendría mediante una métrica euclídea, excepto en una constante multiplicativa.

[34] Además este recorrido en pos de "buscar la frontera" no emplearía la dirección del gradiente para ir avanzando (ya que aún no se ha calculado).

En el algoritmo expuesto los autores no identifican el tipo de optimización empleado para fijar los parámetros de la SVM (C y γ); de hecho en su artículo más relevante [63] no indican ni el tipo de "*kernel*" que se emplea para esta SVM (personalmente supongo que será una SVM-RBF por el tipo de algoritmo empleado para encontrar el punto más próximo en la frontera de separación entre clases).

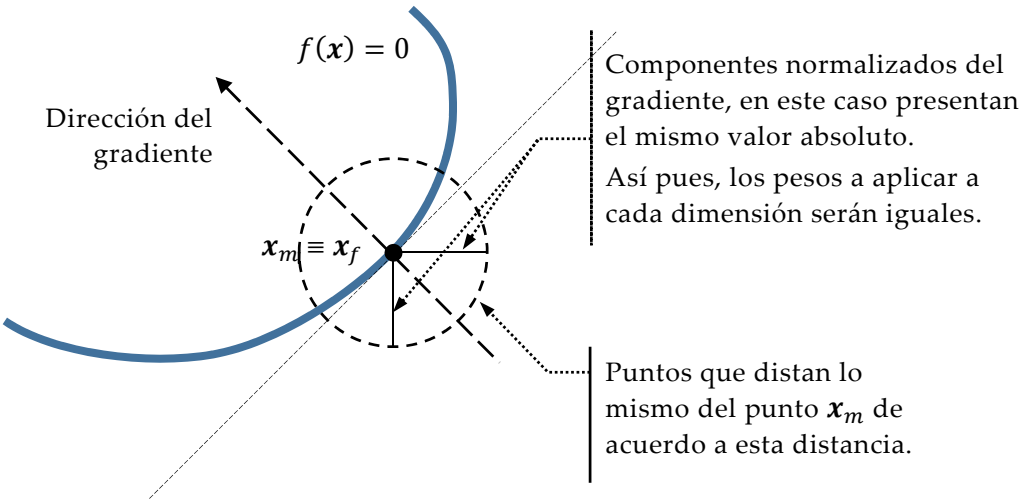


Figura 4.11.- Crítica a la orientación del vecindario equidistante de un cierto punto que proporciona el algoritmo LaMaNNa.

4.4.5 Algoritmo LMNN ("Large Margin Nearest Neighbor")

Kilian Weinberger propone, en el artículo más significativo sobre el algoritmo LMNN [128], optimizar la matriz completa $\boldsymbol{M}$ de una métrica[35] (que además debe ser semidefinida positiva) basándose en la información presente en los casos empíricos de aprendizaje.

4.4.5.1 Objetivo

Determinar una transformación lineal $\boldsymbol{L}$ que optimice la clasificación k-NN, operando con ternas de instancias $\{\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_l\}$, en la que los pares $\{\boldsymbol{x}_i, \boldsymbol{x}_j\}$ pertenecen al subconjunto de instancias vecinas y los $\{\boldsymbol{x}_i, \boldsymbol{x}_l\}$ a la de casos disimilares.

4.4.5.2 Enfoque

De acuerdo con la regla de clasificación k-NN, se mejorará la tasa de aciertos cuando todos los k casos próximos a uno dado tiendan a concentrarse en una misma clase. Además, si se dota de una cierta "distancia de seguridad" entre los casos de aprendizaje de la clase correcta y los denominados "impostores" (que pertenecen a otras clases) el algoritmo de clasificación incrementará su robustez.

Matemáticamente, esto se expresa diciendo que la matriz de la métrica se parametrizará de manera que resulten penalizadas las grandes distancias entre casos, y las pequeñas a los casos disimilares (forzando a estos casos "extraños" a salir de un área de seguridad entorno a cada clase).

De ahí la imagen de que está métrica actúa "empujando" hacia adentro de la vecindad las instancias con igual clase y "expulsando" fuera de esa zona a los "impostores".

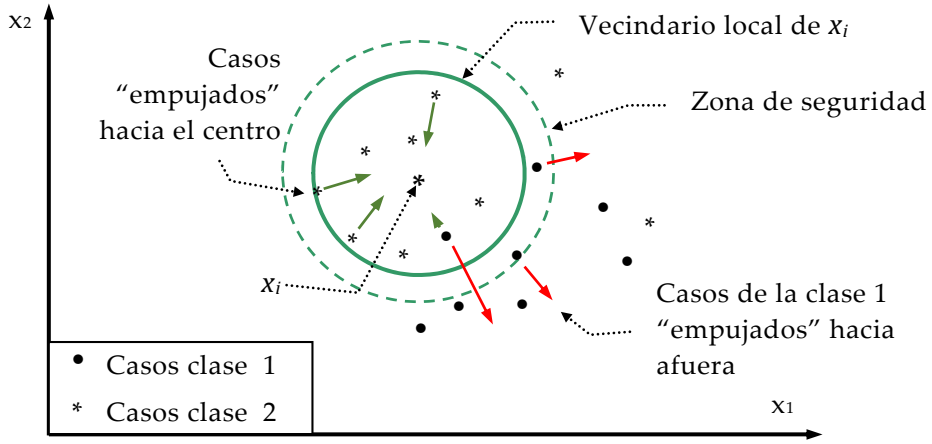


Figura 4.12.- Fundamento de la actuación de la métrica LMNN.

[35] En el artículo citado el autor se refiere a "una matriz de Mahalanobis", aunque explica claramente que nada tiene que ver con la inversa de la matriz de covarianzas de los atributos.

En resumen, esta métrica (expresada mediante la matriz $\boldsymbol{M}_M$) intentará aumentar la cohesión (medida en términos de distancia) de los casos que pertenezcan a una misma clase alejando al resto.

$$d^2_{LMNN}(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{M}_M (\boldsymbol{x}_i - \boldsymbol{x}_j) \tag{4.85}$$

Como $\boldsymbol{M}_M$ es simétrica y semidefinida positiva (por exigencia del algoritmo), la matriz de la métrica siempre se puede expresar como:

$$\boldsymbol{M}_M = \boldsymbol{L}^T \boldsymbol{L} \tag{4.86}$$

Como se ha comentado, el objetivo de la optimización de la métrica es aumentar la distancia entre los casos pertenecientes a distintas clases ("push" impostores), al mismo tiempo favorecer que los N vecinos próximos a uno dado se acerquen ("pull" vecinos), por tanto la función de coste a minimizar será una combinación convexa[36] de dos términos (que dependerán de los elementos de la matriz de transformación $\boldsymbol{L}$):

$$f_{err}(\boldsymbol{L}) = \frac{1}{2}\mathcal{J}_{pull}(\boldsymbol{L}) + \frac{1}{2}\mathcal{J}_{push}(\boldsymbol{L})$$

$$\begin{aligned} \mathcal{J}_{pull}(\boldsymbol{L}) &= \sum_i \sum_j A_{i,j} \left\| \boldsymbol{L}(\boldsymbol{x}_i - \boldsymbol{x}_j) \right\|^2 \\ \mathcal{J}_{push}(\boldsymbol{L}) &= \sum_i \sum_j \sum_l A_{i,j} \left(1 - c_{y_i}(\boldsymbol{x}_l)\right) * \\ &\quad * \left[1 + \left\| \boldsymbol{L}(\boldsymbol{x}_i - \boldsymbol{x}_j) \right\|^2 - \left\| \boldsymbol{L}(\boldsymbol{x}_i - \boldsymbol{x}_l) \right\|^2 \right]_+ \end{aligned} \tag{4.87}$$

donde:

$A_{i,j}$ es un elemento de una matriz[37] $\boldsymbol{A}$ que vale 1 si el caso j-ésimo es un vecino próximo del caso i-esimo, en caso contrario este elemento valdrá 0.

$c_{y_i}(\boldsymbol{x}_l)$ es una función que se evalúa como 1 si la clase del caso i-ésimo coincide con la del l-ésimo.

El subíndice l enumera todos los casos del problema; al incluir el término $\left(1 - c_{y_i}(\boldsymbol{x}_l)\right)$, la última de las expresiones de las ecuaciones (4.87) contempla todos los casos que no pertenecen a la misma clase que el caso i-ésimo.

Y, por último, el $[\ ]_+$ es una variante de la función de pérdida Hinge, donde el resultado del corchete es el valor máximo de cero o el valor presente dentro de dicho corchete.

---

[36] Aunque la combinación convexa más general es: $f_{err}(\boldsymbol{L}) = (1 - \mu)\, \mathcal{J}_{pull}(\boldsymbol{L}) + \mu\ \mathcal{J}_{push}(\boldsymbol{L})$, los autores indican que la calidad de la solución no depende en gran medida de este parámetro, pudiéndose asumir el valor $\mu = 1/2$.

[37] Esta matriz no es simétrica.

Recopilando, la función a minimizar presenta:

- Un primer sumatorio que pretende "acercar" a un determinado caso *i* sus vecinos más próximos.
- Un segundo sumatorio que penaliza que casos de distinta clase (que el *i*) se encuentren más cerca que cualquier otro caso de su misma clase en menos de una unidad (esto es, el margen, y que solo afecta al algoritmo como un factor de escala).

Se diferencia del algoritmo LDA en que no pretende maximizar la proximidad entre sí de todos los casos que pertenecen a una misma clase, sino que tiene un enfoque local, limitando la maximización a los N vecinos más próximos.

Al igual que en las SVM (que también emplean la función de pérdida Hinge), la función a minimizar es convexa, pero no es lineal en la matriz de los parámetros a optimizar (a diferencia de la matriz $\boldsymbol{M}_M$, la matriz $\boldsymbol{L}$ es real pero no tiene por qué ser semidefinida positiva).

Para conseguir una optimización mediante un método que no permita que quede atrapada en mínimos locales, Weinberger replantea el problema como una programación semidefinida[38] [134].

Sustituyendo la ecuación (4.85) en (4.87):

$$f_{err}(\boldsymbol{M}_M) = \frac{1}{2}\sum_i \sum_j A_{i,j}\, d^2_{LMNN}(\boldsymbol{x}_i,\boldsymbol{x}_j) + \\ + \frac{1}{2}\sum_i \sum_j \sum_l A_{i,j}\left(1 - c_{y_i}(\boldsymbol{x}_l)\right)\left[1 + d^2_{LMNN}(\boldsymbol{x}_i,\boldsymbol{x}_j) - d^2_{LMNN}(\boldsymbol{x}_i,\boldsymbol{x}_l)\right]_+ \tag{4.88}$$

Para abordar problemas con difícil separabilidad entre clases, es necesario relajar la exigencia que de que todos los casos con clase distinta a la deseada estén fuera de la zona de seguridad. Para ello se introducen variables de holgura ("slack variables") para sustituir la función de Hinge en el término $\left[1 + d^2_{LMNN}(\boldsymbol{x}_i,\boldsymbol{x}_k) - d^2_{LMNN}(\boldsymbol{x}_i,\boldsymbol{x}_l)\right]_+$.

Ahora sí será posible plantear un problema de tipo SDP:

$$\text{Minimizar} \quad \frac{1}{2}\sum_i \sum_j A_{i,j}\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)^T \boldsymbol{M}_M \left(\boldsymbol{x}_i - \boldsymbol{x}_j\right) + \\ + \frac{1}{2}\sum_i \sum_j \sum_l A_{i,j}\left(1 - c_{y_i}(\boldsymbol{x}_l)\right)\xi_{ijl} \tag{4.89}$$

Sujeto a:

$(\boldsymbol{x}_i - \boldsymbol{x}_l)^T \boldsymbol{M}_M (\boldsymbol{x}_i - \boldsymbol{x}_l) - \left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)^T \boldsymbol{M}_M \left(\boldsymbol{x}_i - \boldsymbol{x}_j\right) \geq 1 - \xi_{ikl} \quad \forall\, A_{i,j} = 1$

$\xi_{ikl} \geq 0$

$\boldsymbol{M}_M \succcurlyeq 0$

[38] Ver una sucinta descripción de la programación semidefinida en la sección 4.2.3.

4.4.5.3 Complejidad computacional

Para clasificar datos nuevos el trabajo del algoritmo es liviano, tanto en términos de memoria requerida como de cómputo necesario. En el fondo se emplearía una clasificación *k*-NN con una norma basada en la matriz $\boldsymbol{M}_M$ (que es una matriz de N x N); para ello se emplearía la distancia $d_{LMNN}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ reflejada en la ecuación (4.85).

Los requerimientos computacionales son más importantes en la etapa de aprendizaje. La función objetivo es una forma cuadrática, las restricciones son lineales y la matriz de la forma cuadrática debe ser semidefinida positiva. Bajo este enfoque será necesario[39]:

- Calcular los vecinos próximos de cada caso $\boldsymbol{x}_i$.
- Calcular cuál es el caso más lejano de su misma clase y de ahí calcular la "distancia de seguridad".
- Calcular los vecinos dentro de esta área de seguridad.
- Ir acumulado los productos de la forma cuadrática para todos los vecinos de $\boldsymbol{x}_i$.
- Identificar a los vecinos dentro del área de seguridad que no poseen la misma clase que el $\boldsymbol{x}_i$.
- Con cada uno de estos últimos preparar una inecuación de acuerdo a la primera restricción de las fórmulas (4.89).
- Someter el problema de minimización a un paquete de programación semidefinida.

De este proceso se obtendrán los elementos de la matriz $\boldsymbol{M}_M$.

4.4.5.4 Ventajas e inconvenientes

El enfoque del algoritmo LMNN es muy interesante, aplica a la optimización de la medida de distancias, ideas procedentes de las SVM (que ya se adelantaban en el algoritmo POLA de Shalev-Shwartz et al. [126]) y conceptos de programación semidefinida (SDP), para intentar crear una métrica que se "ajuste" al problema de clasificación a resolver.

El objetivo de optimizar una matriz completa de una métrica también se observa en trabajos anteriores de otros investigadores como Goldberger et al. en su algoritmo NCA ("Neighborhood Component Analysis") [100].

Uno de los problemas principales de este algoritmo es que hay que identificar (lo mejor sería de forma manual) a los vecinos más próximos de todos y cada uno de los casos. Estos vecinos no variarán a lo largo del todo el proceso. No es una cuestión totalmente resuelta[40], en el mismo artículo se comentan alternativas para afrontarla. La variación del algoritmo conocida como

---

[39] Para la versión básica.

[40] Los propios autores lo reconocen en el artículo de referencia.

"Multi-pass LMNN" plantea realizar varias iteraciones en el algoritmo, empleando la métrica optimizada en la anterior vuelta para determinar los vecinos más próximos en la iteración actual. Sus resultados son mixtos, en unos casos se logran mejoras y en otros se presenta bastante sobreaprendizaje.

Finalmente, el mismo artículo desvela un ejemplo en el que esta métrica se comporta especialmente mal; se trata de la clasificación de casos que forman dos círculos concéntricos. El error en la clasificación alcanza el 100% y los autores proponen optimizar métricas diferentes en distintas áreas del espacio de atributos (estas áreas las obtienen mediante un algoritmo de formación de "clusters"). A esta nueva versión del algoritmo la denominan "Multi-metric LMNN", la cual pierde la propiedad de ser una métrica (ya que no cumple con el axioma de que la medida de distancia sea simétrica); y, en mi opinión, dudo también que cumpla con el axioma de la desigualdad triangular.

4.4.6 Escalado de un "*kernel*" RBF dependiendo de la densidad de casos en el espacio de atributos.

Aunque posterior en el tiempo a los enfoques expuestos en la siguiente subsección, la aportación de Chang et al. es la más simple de todas las que aquí se contemplan [131], y por eso se revisa en primer lugar.

4.4.6.1 Objetivo

Su objetivo es ajustar localmente el radio de los "*kernel*" RBF en función de la densidad de casos de aprendizaje presentes en un cierto volumen del espacio de atributos.

4.4.6.2 Enfoque

Pretende aumentar dicho radio en aquellas zonas en las que la densidad de casos es menor y reducirlo donde se acumulan muchos casos. Así pues, se trata de expresar la σ del "kernel" en función de la densidad, de forma que varíe inversamente.

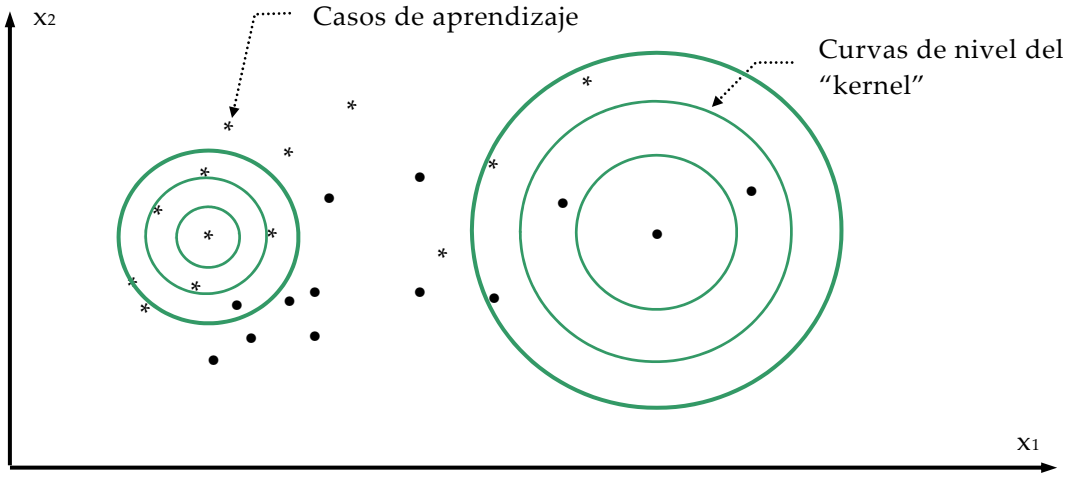


Figura 4.13.- "Kernel" escalado de acuerdo a la densidad de casos en la zona. A la izquierda se ve que las curvas de nivel del "kernel" se juntan entre sí y, por tanto el radio de influencia de este "kernel" es más reducido. A la derecha se ve el fenómeno contrario.

Para evaluar la densidad de información empírica (ρ) que circunda al caso *i*-ésimo, los autores promedian el valor de similitud entre él y sus *k* vecinos más próximos.

El valor de similitud se obtiene al evaluar la función del "*kernel*" entre dicho caso *i*-ésimo y cada uno de sus *k* vecinos más próximos; dicho promedio será:

$$\rho(\boldsymbol{x}_i) = \frac{1}{k}\sum_{m=1}^{k} K(\boldsymbol{x}_i, \boldsymbol{x}_m), \quad \boldsymbol{x}_m \in k\text{-}NN \tag{4.90}$$

Un valor grande para $\rho(\boldsymbol{x}_i)$ indica que el caso *i*-ésimo está rodeado por otros muchos casos.

Luego estima la densidad media global para todo el problema $\bar{\rho}$, promediando el valor de la densidad calculada para todos los casos individuales:

$$\bar{\rho} = \frac{1}{N}\sum_{n=1}^{N} \rho(\boldsymbol{x}_n) \tag{4.91}$$

Y luego, con ambos términos, calcula un factor de ponderación para el "*kernel*", que se aplica de forma individual para cada caso $\boldsymbol{x}_i$:

$$\omega(\boldsymbol{x}_i) = 1 + \eta(\rho(\boldsymbol{x}_i) - \bar{\rho}) \tag{4.92}$$

donde η es un parámetro a elegir (u optimizar) comprendido entre 0 y 1.

En aquellas zonas donde la densidad de casos se aproxime a la media este peso $\omega(\boldsymbol{x}_i)$ valdrá aproximadamente 1, en las zonas de alta densidad su valor estaría comprendido entre 1 y 2, y en aquellas con muy pocos casos sería menor que la unidad.

El objetivo de estos pesos es modificar el radio de influencia del "*kernel*" RBF, para ello los autores plantean modificarlo de la siguiente forma:

$$\widetilde{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) = e^{-\omega(\boldsymbol{x}_i).\,\omega(\boldsymbol{x}_j)\,\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\sigma^2}} \tag{4.93}$$

Como se aprecia, el producto de ambos pesos afecta de forma inversa al radio del "*kernel*", sería equivalente a reemplazar $2\sigma^2$ por $2\sigma^2/[\omega(\boldsymbol{x}_i).\,\omega(\boldsymbol{x}_j)]$.

A partir de ahora se empleará este "*kernel*" modificado localmente para clasificar los nuevos casos.

4.4.6.3 Complejidad computacional

Este algoritmo de modificación del "*kernel*" se podría encuadrar dentro de los métodos de aprendizaje demorado si el conjunto de casos de aprendizaje se modificase a lo largo del tiempo y los pesos se tuviesen que calcular en el momento en que se presenta un nuevo caso a clasificar. En dicho escenario, su costo computacional haría que fuese muy poco escalable frente al número de casos.

Pero para problemas con un número de casos fijos, la densidad propia de cada caso de aprendizaje y la densidad global se han podido establecer de antemano. Con cada caso de aprendizaje se guardaría una nueva información que sería su peso $\omega(\boldsymbol{x}_i)$ a aplicar[41] y cuando se presente un nuevo caso a clasificar $\boldsymbol{x}_m$:

- Se calculan sus k vecinos próximos.
- Con ellos y empleando el "*kernel*" original, y de acuerdo a la ecuación (4.90) se calcularía la densidad de vecinos $\rho(\boldsymbol{x}_m)$.

[41] Todos estos pesos se calculan de forma previa y se almacenan.

- Mediante ella y la densidad global se calcula el peso para el nuevo caso $\omega(\boldsymbol{x}_m)$.
- A partir de este punto es posible calcular rápidamente el valor de un "*kernel*" que implique a este caso y otro cualquiera (ya que los pesos de cada uno de los casos de aprendizaje están precalculados).

Todo ello no representa grandes requisitos de memoria (almacenar un valor real más por caso), y tampoco los de cálculo son muy elevados, ya que por cada caso nuevo a clasificar se debe ejecutar el cálculo previo equivalente al necesario en un algoritmo de tipo *k*-NN. Así pues, el método se puede escalar fácilmente a problemas con muchos atributos y con una muestra de casos de tamaño moderadamente grande.

Si se desease estimar el valor óptimo de η en aquellos casos en que este tipo de "*kernels*" se utilizasen para resolver problemas de clasificación, en la fase de precálculo se debería emplear una estrategia de validación cruzada, y el tiempo de cálculo anterior se vería multiplicado por el número de valores seleccionados para dicho parámetro. En la fase de uso del "*kernel*" modificado el tiempo no se vería afectado (ya que el valor de η estaría ya fijado).

4.4.6.4 Ventajas e inconvenientes

Este "*kernel*" modificado sería de aplicabilidad tanto en los métodos de clasificación SVM-RBF, como en aquellos de tipo *k*-NN que aplicasen una medida de similitud en el vecindario para estimar la clase de un nuevo caso (como sucede en nuestro algoritmo BTW).

Las aportaciones más interesantes son su inherente simplicidad y el haber diseñado una métrica local donde la proximidad entre casos depende de la zona del espacio de atributos donde se encuentre un caso.

Desde mi punto de vista presenta un importante aspecto de fondo considerar: si el aumentar o disminuir el radio localmente, pero de forma igual en todas las direcciones, realmente influye de forma importante en la precisión de la clasificación o no (ver la aportación de esta tesis en la sección 4.5).

4.4.7 Modificación de la métrica de un "*kernel*" mediante transformaciones cuasiconformes

Para tratar de mejorar la tipificación de casos en problemas no separables linealmente, los métodos de clasificación basados en "*kernels*" (ver sección 4.2.3.5) aplican una transformación no lineal $\Phi(\boldsymbol{x})$, proyectándolos sobre un espacio de Hilbert ($\mathcal{H}$) de grandes dimensiones.

Una vez elegida correctamente la función del "*kernel*" $K(\boldsymbol{x}, \boldsymbol{x}')$, es de esperar que aumente la separabilidad general de los casos en $\mathcal{H}$, lo cual permitirá a su vez mejorar la de aquellos casos que estuviesen próximos a la frontera de separación de las clases y, por tanto, mejorar la precisión de clasificación.

Transformaciones cuasiconformes

Antes de describir el funcionamiento de las transformaciones cuasiconformes, será necesario definir qué son.

Si la función $f(\ \ )$, continua y diferenciable, representa un homeomorfismo entre dos espacios métricos $\mathcal{X}$ y $\mathcal{Z}$; y si para $\boldsymbol{x}_i,\ \boldsymbol{x}_j \in \mathcal{X}$ y $r \in \mathbb{R}^+$:

$$\begin{aligned} L_f(\boldsymbol{x}_i, r) &= \sup\left\{ dist\left(f(\boldsymbol{x}_i), f(\boldsymbol{x}_j)\right) \mid dist(\boldsymbol{x}_i, \boldsymbol{x}_j) = r \right\} \\ l_f(\boldsymbol{x}_i, r) &= \inf\ \left\{ dist\left(f(\boldsymbol{x}_i), f(\boldsymbol{x}_j)\right) \mid dist(\boldsymbol{x}_i, \boldsymbol{x}_j) = r \right\} \end{aligned} \tag{4.94}$$

el ratio:

$$H_f(\boldsymbol{x}_i, r) = L_f(\boldsymbol{x}_i, r)/l_f(\boldsymbol{x}_i, r) \tag{4.95}$$

mide la excentricidad de una esfera infinitesimal bajo la transformación f.

Se dice que f es H-cuasiconforme si:

$$\forall\, \boldsymbol{x} \in \mathcal{X},\quad \lim_{r \to 0} \sup H_f(\boldsymbol{x}, r) \leq H\,,\ \ H \geq 1 \tag{4.96}$$

Así pues, una transformación cuasiconforme [135], de acuerdo a la definición métrica, es aquella que transforma esferas infinitesimales en elipsoides infinitesimales de excentricidad acotada.

Factor de magnificación

Se recuerda que en la métrica del espacio $\mathcal{H}$, un elemento diferencial de volumen se puede calcular mediante:

$$dV = \sqrt{|\boldsymbol{G}|}\ \ dx_1 \ldots dx_A$$

con:

$$\begin{aligned} |\boldsymbol{G}| &= \left|\{\, g_{ij}(\boldsymbol{x})\}\right| \\ g_{ij}(\boldsymbol{x}) &= \frac{\partial}{\partial x_i}\frac{\partial}{\partial x_j'} K(\boldsymbol{x}, \boldsymbol{x}')|_{x=x'} \end{aligned} \tag{4.97}$$

Al factor $\sqrt{|\boldsymbol{G}|}$ se lo conoce como "factor de magnificación" (y en la métrica euclídea vale 1).

En esta subsección de la revisión del estado del arte se van a exponer enfoques de diversos autores sobre las posibilidades de modificar este factor de

magnificación mediante transformaciones cuasiconformes (todo ello inmerso en el campo de las métricas de Riemann).

4.4.7.1 Objetivo

El objetivo es potenciar aún más el efecto separador del "*kernel*" en las proximidades de la frontera de separación entre clases, definiendo otro "*kernel*" $\widetilde{K}(\boldsymbol{x},\boldsymbol{x}')$ que permita una mayor separación entre los puntos en el espacio $\mathcal{H}$. Este nuevo "*kernel*" se derivará del anterior a través de una transformación cuasiconforme.

4.4.7.2 Enfoque

Para ello se modifica mediante una transformación cuasiconforme el "*kernel*" inicial, o sea:

$$\widetilde{K}(\boldsymbol{x},\boldsymbol{x}') = D(\boldsymbol{x})\, K(\boldsymbol{x},\boldsymbol{x}')\, D(\boldsymbol{x}') \qquad \text{con} \quad D(\boldsymbol{x}) > 0 \tag{4.98}$$

En este nuevo "*kernel*", un elemento del tensor de su métrica riemanniana inducida se expresará como:

$$\tilde{g}_{ij}(\boldsymbol{x}) = D^2(\boldsymbol{x}) g_{ij}(\boldsymbol{x}) + D_i(\boldsymbol{x}) D_j(\boldsymbol{x}) + D(\boldsymbol{x})\big[D_i(\boldsymbol{x}) K_j(\boldsymbol{x}) + D_j(\boldsymbol{x}) K_i(\boldsymbol{x})\big]$$

donde:

$$D_i(\boldsymbol{x}) = \partial D(\boldsymbol{x}) / \partial x_i$$

$$K_i(\boldsymbol{x}) = \partial K(\boldsymbol{x},\boldsymbol{x}') / \partial x_i \,|_{\boldsymbol{x}=\boldsymbol{x}'}. \tag{4.99}$$

Por medio de esta transformación, un elemento de volumen diferencial en el espacio original resultará magnificado en el espacio $\mathcal{H}$.

Se intentará que esta magnificación sea más acentuada en las proximidades de la superficie de separación entre clases; para ello $D(\boldsymbol{x})$ deberá tomar en esa zona el máximo valor posible y, fuera de ahí, valores pequeños. Bajo esta consideración, se puede considerar que la función del "*kernel*" $\widetilde{K}(\boldsymbol{x},\boldsymbol{x}')$ varía localmente.

Propuestas de elección para la transformación $\boldsymbol{D(x)}$

Propuesta 1:

Williams et al. [136] proponen que $D(\boldsymbol{x})$ decaiga progresivamente al alejarse de la frontera de separación entre clases mediante la expresión[42]:

$$D(\boldsymbol{x}) = e^{-\frac{f^2(\boldsymbol{x})}{2\tau^2}}$$

donde:

$$f(\boldsymbol{x}) = \sum_{i \in SV} \alpha_i\, y_i\, K(\boldsymbol{x},\boldsymbol{x}_i) + b \tag{4.100}$$

[42] Para una función de separación calculada mediante una SVM.

Así pues:

$$D(\boldsymbol{x}) = \begin{cases} 1 & f(\boldsymbol{x}) = 0 \\ e^{-1/\tau^2} & f(\boldsymbol{x}) = \pm 1 \end{cases} \tag{4.101}$$

Al desplazarse un punto desde el plano de separación[43] (donde $f(\boldsymbol{x}) = 0$) hacia su plano canónico, $D(\boldsymbol{x})$ decae progresivamente.

En la Figura 4.14 se puede apreciar, en forma tridimensional, los valores de una transformación $D(\boldsymbol{x})$ computada de acuerdo a esta propuesta, empleando el valor τ=0,6.
Para calcularla se ha empleado una función de decisión de dos dimensiones (la línea gruesa que se ve en el plano base), previamente estimada mediante datos experimentales (puntos que se ven también en el plano base).

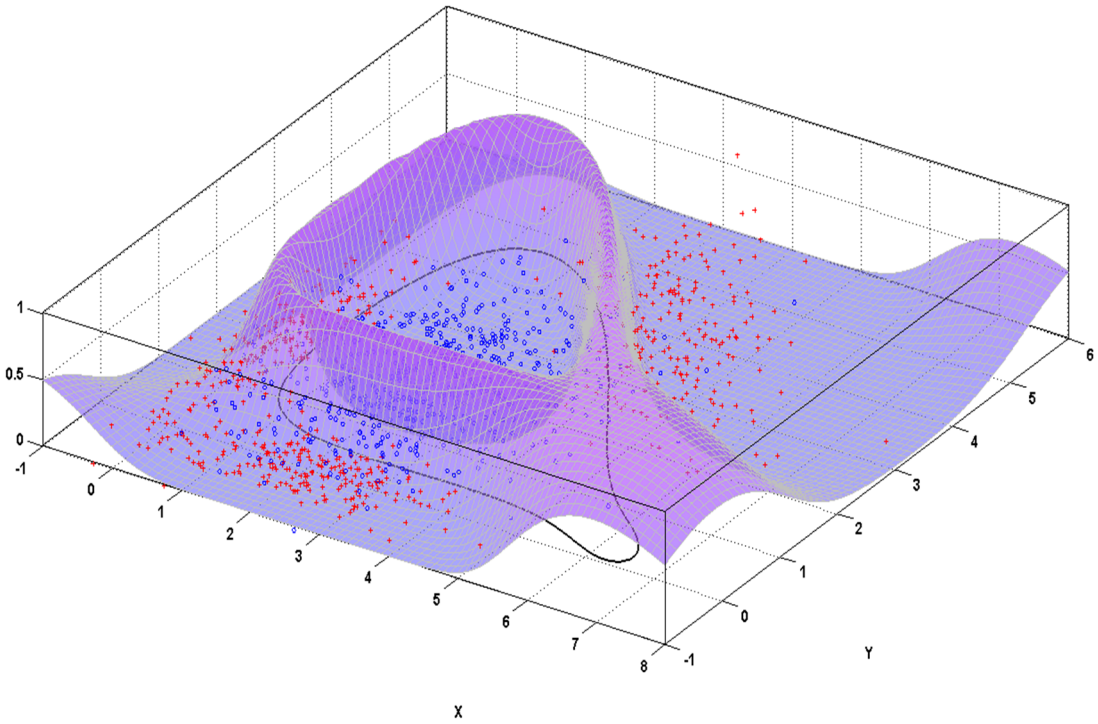


Figura 4.14.- Función $D(\boldsymbol{x})$ para la transformación cuasiconforme de acuerdo a Williams et al.

Propuesta 2:

Amari y S. Wu propusieron en [133] utilizar una transformación cuasiconforme que se basa en la suma de similitudes ponderadas del punto requerido ($\boldsymbol{x}$) con los distintos vectores de soporte ($\boldsymbol{x}_i$).

$$D(\boldsymbol{x}) = \sum_{i \in SV} \alpha_i \, e^{-\frac{\|\boldsymbol{x} - \boldsymbol{x}_i\|^2}{2\tau^2}} \tag{4.102}$$

donde τ es un valor fijo a elegir.

[43] En el espacio $\mathcal{H}$.

Ponderan más aquellos vectores de soporte que se localizan dentro del margen de seguridad.
En este caso, en el entorno del vector de soporte *i*-ésimo (y sin tener en cuenta la influencia de otros vectores de soporte), el factor de magnificación de la nueva métrica será:

$$\sqrt{|\boldsymbol{G}|} = \frac{\alpha_i^A}{\sigma^A}\, e^{-A\,\frac{\|\boldsymbol{x}-\boldsymbol{x}_i\|^2}{2\tau^2}} \sqrt{1+\frac{\sigma^2}{\tau^4}\|\boldsymbol{x}-\boldsymbol{x}_i\|^2} \tag{4.103}$$

Se puede comprobar, estudiando su derivada primera, que el punto de máxima magnificación, $r = \|\boldsymbol{x}-\boldsymbol{x}_i\|$, es:

Para $\tau < \sigma/\sqrt{A}$ $\quad r = \tau\sqrt{1/A - \tau^2/\sigma^2}$
Para $\tau \geq \sigma/\sqrt{A}$ $\quad r = 0$

Es decir la magnificación se da para valores pequeños de *r*, y por lo tanto en las proximidades de los vectores de soporte.

Propuesta 3:
Los mismos autores de la anterior propuesta, tres años más tarde y manteniendo la idea base, plantearon estimar el parámetro τ en función de los vectores soporte que rodean a uno dado [137].

$$D(\boldsymbol{x}) = \sum_{i\,\in\,SV} e^{-\frac{\|\boldsymbol{x}-\boldsymbol{x}_i\|^2}{\tau_i^2}} \tag{4.104}$$

donde:

$$\tau_i^2 = \frac{1}{M}\sum_{m=1}^{M}\|\boldsymbol{x}_m - \boldsymbol{x}_i\|^2$$

y $\{\boldsymbol{x}_m \mid 1 \leq m \leq M\}$ es el subconjunto de los M vectores de soporte más próximos al vector de soporte $\boldsymbol{x}_i$.

A diferencia de la segunda propuesta, en la cual el radio τ es fijo en todo el espacio, en esta el radio se adapta a la densidad de los vectores de soporte en el entorno de $\boldsymbol{x}_i$. De esta forma este parámetro presenta un carácter local y se acomoda a la distribución de información presente en el problema[44].

También sugieren en este mismo artículo que el algoritmo se puede aplicar iterativamente, empezando por ajustar al problema en cuestión una SVM con una función "*kernel*" convencional, para luego crear un nuevo "*kernel*" de acuerdo a las fórmulas (4.98) y (4.104), con él volver a ajustar otra SVM, crear un nuevo "*kernel*"...

[44] Este enfoque presenta ciertas similitudes conceptuales con el presentado por Chang et al. [131] y que ha sido expuesto en la sección 4.4.6.

Propuesta 4:

Posteriormente, G. Wu et al., partiendo de un planteamiento similar al reflejado en la ecuación (4.104) propusieron calcular el radio τ_i directamente en el espacio de las características [138] aplicando la siguiente fórmula:

$$\tau_i^2 = \frac{1}{V}\sum_v \|\Phi(\boldsymbol{x}_v) - \Phi(\boldsymbol{x}_i)\|^2 = \\ = \frac{1}{V}\sum_v K(\boldsymbol{x}_v, \boldsymbol{x}_v) + K(\boldsymbol{x}_i, \boldsymbol{x}_i) - 2K(\boldsymbol{x}_v, \boldsymbol{x}_i) \qquad (4.105)$$

donde:

$$\boldsymbol{x}_v \in SV \mid \|\Phi(\boldsymbol{x}_v) - \Phi(\boldsymbol{x}_i)\|^2 < R$$

donde V es el número de vectores de soporte localizados dentro de una esfera de radio R en el espacio de Hilbert.

Esta última propuesta intenta mejorar los resultados de la clasificación en los problemas donde el número de casos de aprendizaje para las distintas clases presenta grandes desequilibrios.

El elevado número de propuestas similares revela el interés que ha despertado este tipo de planteamientos entre la comunidad científica.

4.4.7.3 Ventajas e inconvenientes

Estos cuatro planteamientos se distinguen de los expuestos en los otros apartados del estado de arte en que se fundamentan en el uso de la geometría de Riemann para estudiar y modificar las propiedades de la superficie de separación entre clases en el espacio de Hilbert.

Williams et al. utilizaron directamente la función de decisión proveniente de una SVM, el grupo de Amari y S. Wu por el contrario se basaron en los vectores de soporte. C. Wu et al. introdujeron cálculos de distancia en el espacio de las características.

El principal inconveniente de todos estos enfoques es que operan sobre la base de "ensanchar" uniformemente la geometría de las zonas próximas a la frontera de separación de clases.

Como se expondrá en el capítulo 6, la métrica LOM de esta tesis resuelve este inconveniente proponiendo modificar el espacio "ensanchando" la geometría siguiendo unas direcciones discriminantes, en aras a mejorar la precisión de clasificación.

4.5 Acerca de la mejora de la clasificación inducida por la magnificación de un elemento diferencial de volumen

En los trabajos de Amari y Wu, Williams et al., Peng et al.,... se emplean transformaciones cuasiconformes que pretenden magnificar las zonas próximas a la frontera de separación entre clases.
Sin aportar pruebas analíticas, postulan que si se logra "ensanchar" el volumen de las zonas deseadas, la capacidad de separar casos de diferentes clases se verá incrementada en dichas áreas.

Para ilustrar gráficamente la propuesta, se considera un problema de clasificación en un espacio de dos dimensiones; sobre él se encuentran tres puntos muy próximos: A, B y C; A y C pertenecen a la clase +1, mientras que B pertenece a la clase -1. Los puntos B y C equidistan de A en una métrica euclídea. La línea horizontal de punto y raya sirve como frontera de separación teórica de las clases +1 y -1.

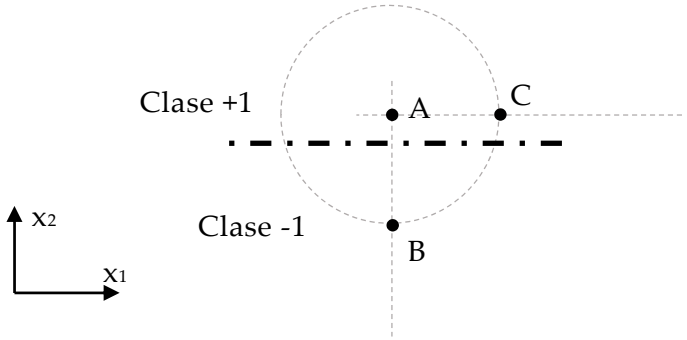


Figura 4.15.- Disposición de los puntos A, B y C en el plano y de la frontera de separación.

Obviamente, si se desease clasificar un punto cuya clase es "a priori" desconocida, p.ej. el A (aunque realmente se sabe que A pertenece a la clase +1) basándose en su vecino más próximo, se daría un empate a la hora de clasificarlo en una u otra clase[45].
Se recuerda que al estar utilizando una métrica euclídea, el factor de magnificación $\sqrt{|\boldsymbol{G}|}$ es aquí 1.

Supóngase que se desea magnificar de forma isótropa el espacio (como sugieren los autores citados al principio de esta sección) multiplicando el diferencial de longitud en cada dirección de los ejes coordenados por un valor constante como puede ser $1/\mathrm{tg}\,\alpha$ (α es una constante).

Así pues:

[45] De hecho, si el punto B se encontrase un poco más hacia arriba, el punto A sería clasificado de forma incorrecta.

$$dx_1' = 1/\mathrm{tg}\,\alpha \; dx_1$$
$$dx_2' = 1/\mathrm{tg}\,\alpha \; dx_2 \tag{4.106}$$

Un elemento diferencial de longitud tendrá una expresión tal como:

$$dz^2 = (1/\mathrm{tg}\,\alpha)^2 \, [dx_1, dx_2] \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} dx_1 \\ dx_2 \end{bmatrix} \tag{4.107}$$

Y:

$$\sqrt{|\boldsymbol{G}|} = \sqrt{1/tg^4\alpha} = 1/tg^2\alpha \tag{4.108}$$

Analizando esta expresión se puede comprobar cómo para $\alpha < 45º$ el factor de magnificación es mayor que la unidad[46]; pero, por muy pequeño que sea α, esto no modifica el ratio de la distancia que existe entre A y B y entre A y C (que serán mayores, pero siguen siendo de igual magnitud). La dificultad para clasificar el punto A se mantiene intacta.

Es posible ilustrar este comportamiento mediante una interpretación geométrica (ver Figura 4.16). Supóngase que en el plano bidimensional generamos una tercera dimensión perpendicular a este. Desde el punto A se genera una perpendicular al plano y se va bajando por ella un punto O hasta que el segmento $\overline{OC}$ forme un ángulo α con la vertical. Si suponemos que los puntos A, C y O están a distancias infinitesimales, la distancia original sería *ds* (y estaría en el plano original), mientras que la distancia "magnificada": *dz*, estaría en la perpendicular a dicho plano, siendo igual a la longitud del segmento $\overline{OA}$.

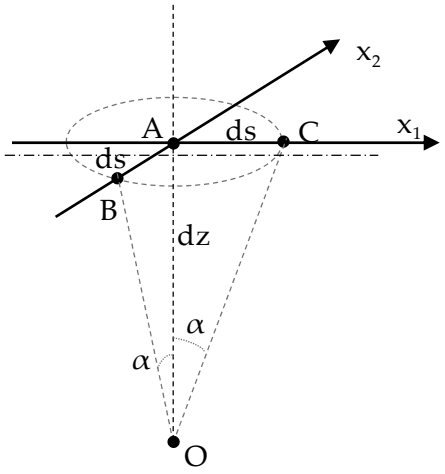


Figura 4.16.-Interpretación geométrica de la magnificación isótropa (con un factor igual a 1/tg α).

[46] El uso de $1/\mathrm{tg}\,\alpha$ también permite que la magnificación se pueda convertir en "reducción", solo hace falta que $\alpha > 45º$.

En este escenario, los puntos que se encuentran a la misma distancia *ds* de A en la métrica euclídea, compartirían también el valor *dz* en la métrica modificada, y por lo tanto, no hay mejora en su capacidad de discernir en el problema de clasificación.

Si, como se pretende en esta tesis, la magnificación no fuera isótropa, sino que fuese mayor a lo largo de la perpendicular a la frontera de separación entre clases (eje x_2): $1/\mathrm{tg}\,\alpha$, que en la dirección paralela (eje x_1): $1/tg\,\beta$, el escenario sería similar al que se puede apreciar en la Figura 4.17.

Se puede apreciar que en la nueva métrica "magnificada" los puntos que equidistan de A se encuentran en la elipse que pasa por B y D; y por tanto el punto C se encuentra "más cerca" de A que el B ya que:

$$dz_C = \overline{O'A} \quad \text{frente a:} \quad dz_B = \overline{OA} \tag{4.109}$$

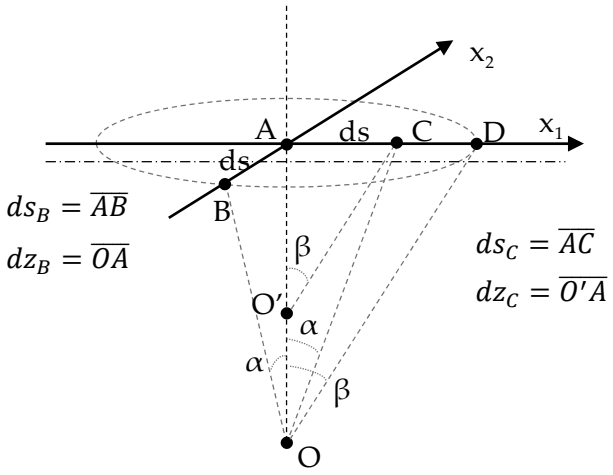


Figura 4.17.- Interpretación geométrica de una magnificación no isótropa que ayuda a clasificar el punto A.

En este nuevo escenario la medida de distancia magnificada ayuda a clasificar mejor el caso A; además, y en contra de lo que postulan los autores citados, su factor de magnificación es menor que en el ejemplo isótropo.

$$\sqrt{|\boldsymbol{G}|} = \sqrt{1/tg^2\alpha \;.\; 1/tg^2\beta} = \frac{1}{tg\;\alpha\;\; tg\;\beta} \tag{4.110}$$

Si no se considerasen elementos diferenciales de longitud, sino de dimensión finita, la crítica seguiría siendo válida. Si se estuviera en un escenario como el de la Figura 4.18, con la frontera de separación de clases "lejos" de C, podría justificarse que el punto A fuese bien clasificado de acuerdo a los postulados de Williams et al. La dilatación del espacio en la dirección $\overline{\mathrm{AC}}$ sería constante y en la dirección $\overline{\mathrm{AB}}$ iría aumentando según nos acercamos a la frontera entre clases (para después disminuir). Podría ser que dicho "aumento" hiciese que A estuviera "más lejos" de B que de C.

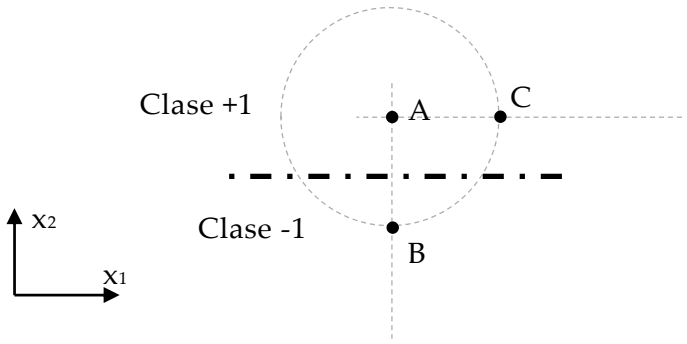


Figura 4.18.- En este escenario, la frontera de separación entre clases está "lejos" de C.

Pero si se analiza el contraejemplo de la Figura 4.19, se aprecia que el camino $\overline{AC}$ está jalonado de puntos "muy distantes" entre sí, mientras que el $\overline{AB}$ tiene una primera fase en la que los puntos están aún más lejos entre sí, pero según se sigue viajando hacia B, y se aleja de la frontera de separación de las clases, los puntos por los que pasa están "mucho más cerca" unos de otros. Por lo que en este escenario es posible que C esté más lejos de A que B.

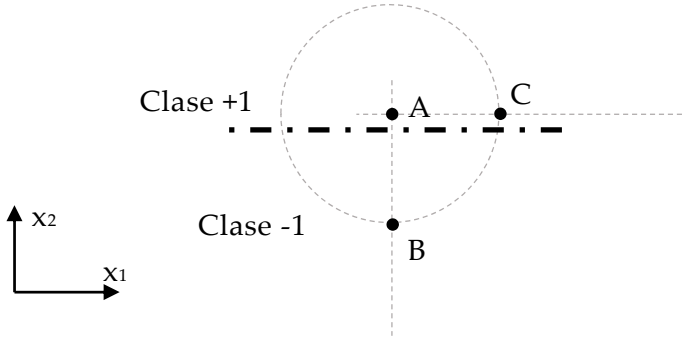


Figura 4.19.- En este escenario, la frontera de separación entre clases está "cerca" de C.

4.5.1 Discusión

Ofrecer un único valor escalar, como el factor de magnificación, para cuantificar la mejora que este puede aportar a un algoritmo de clasificación se demuestra que no es exacto. Tan importante, o más, que este número, serán las direcciones principales en las que se orienta esta magnificación.

Las afirmaciones de dichos autores sobre que la magnificación isótropa mejora la capacidad de clasificar un problema, las puedo comprender bajo el siguiente prisma: es positivo que en las proximidades de la frontera de separación entre clases las distancias "aumenten", disminuyendo cuando se está lejos. Eso hace que, para los elementos más complejos de clasificar (que son los que están cerca de la frontera), los casos más cercanos "decidan más" sobre la clasificación que los más lejanos.

Puede que al utilizar estas transformaciones en "*kernels*" de tipo RBF, el "aumentar la distancia" hace que el valor que proporciona la función exponencial negativa decaiga de forma más rápida y se consiga que en la zona fronteriza los casos "más cercanos" predominen de forma más acusada. Pero presenta más dudas su efectividad sobre el comportamiento de los casos cercanos.

De cualquier forma, las explicaciones aquí aportadas (que se extenderían sin dificultad a más de dos dimensiones) ofrecen un nuevo punto de vista sobre este uso de las transformaciones cuasiconformes y proporcionan una nueva base sobre cómo diseñar algoritmos que mejoren la capacidad de clasificar correctamente nuevos casos.

4.6 Conclusiones

En este capítulo se ha repasado la tecnología y el estado del arte relacionado con esta tesis. En primer lugar se han expuesto las características de las distintas medidas de distancia y similitud que se usan en las investigaciones en este campo y los tres algoritmos básicos de clasificación que luego tienen influencia directa en la tesis: *k*-NN, LDA y SVM.

En un segundo bloque, mucho más exigente desde un punto de vista técnico, se ha repasado el núcleo científico de lo que serán los avances de esta tesis: las técnicas de aprendizaje para las métricas, y las contribuciones más relevantes de diversos investigadores que han conducido a los planteamientos de las métricas BTW y LOM.

Por fin se ha incluido una pequeña aportación, desde un enfoque riemanniano, sobre cómo el elegir una dirección discriminante en la métrica puede mejorar el problema de la clasificación.

Antes de pasar a los dos últimos capítulos, donde se describen las métricas aportadas por esta tesis, sirva el siguiente esquema como recopilación gráfica de los aspectos tratados hasta el momento y sus relaciones sobre lo que falta por abordar.

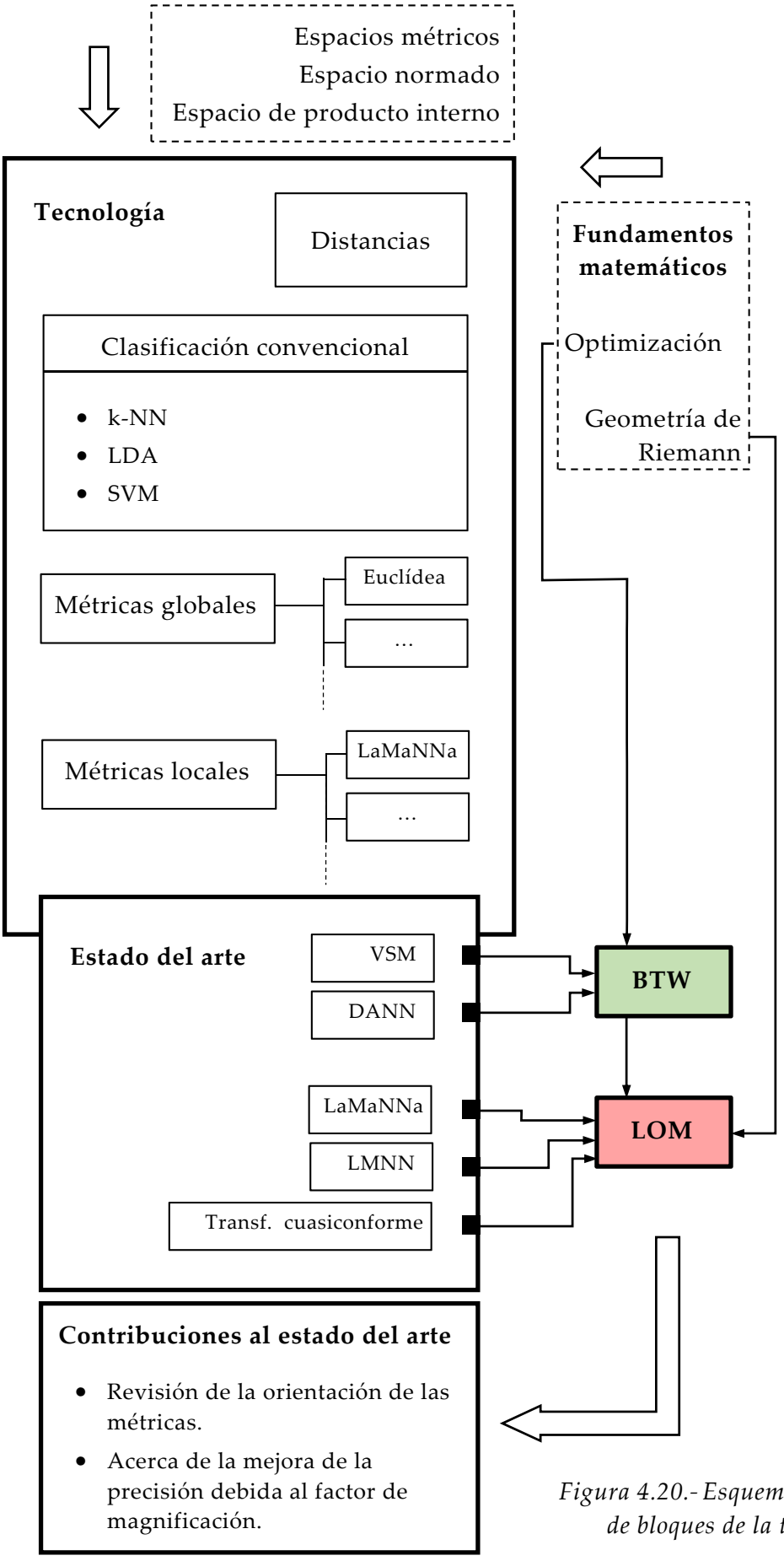


Figura 4.20.- Esquema general de bloques de la tesis.

5 Propuesta de una métrica global: BTW

Una de las aportaciones novedosas de esta tesis es proponer una métrica global y orientada para ser usada en el algoritmo de clasificación *k*-NN.

Se plantea un nuevo método de clasificación, el algoritmo BTW que mediante el uso de la métrica del mismo nombre, pretende mejorar el tiempo de respuesta que tarda el clasificador *k*-NN en predecir la clase a la que pertenece un determinado caso en función de otros casos, siendo además competitivo en cuanto a precisión de clasificación con los métodos tradicionales.

BTW intenta conjugar lo mejor de tres técnicas bien contrastadas: en primer lugar identifica aquellas direcciones del espacio de los atributos con mayor capacidad de discriminación entre dos clases (siguiendo una estrategia de tipo LDA); en segundo lugar, optimiza el radio de búsqueda de los vecinos próximos de forma que el número de aciertos sea lo más alto posible; y, por último, basa su estrategia final de búsqueda de casos similares en un algoritmo de tipo *k*-NN. Todo ello conduce a una nueva contribución al estado del arte: la posibilidad de emplear estrategias de clasificación basadas en la búsqueda de vecinos próximos bajo restricciones de funcionamiento en tiempo real, prácticamente independizando el tiempo de búsqueda de tamaño de la base de datos: $\mathcal{O}(\log N)$, siendo N el número de casos de aprendizaje.

Este capítulo comenzará con una sección de consideraciones previas, donde se reflexionará sobre la importancia de las direcciones de la métrica en la clasificación, para continuar con la descripción del algoritmo BTW, su implementación y el análisis de los resultados experimentales obtenidos.

5.1 Consideraciones previas

Antes de entrar a describir en profundidad el algoritmo BTW, en esta sección se van a exponer un conjunto de reflexiones previas sobre dos de las que serán las aportaciones más importantes de este algoritmo. Por una parte, cuáles son las limitaciones del algoritmo *k*-NN básico (y, por tanto, cuáles son las posibilidades de mejora) y por otra, la interpretación geométrica de la

influencia de la matriz de la métrica que se empleará en los cálculos de distancia.

5.1.1 Las posibles mejoras del algoritmo k-NN

Los clasificadores de tipo *k*-NN (ver sección 4.2.1), a pesar de ser de los mejores del estudio europeo "StatLog", presentan una serie de características y limitaciones:

- El valor de *k* se calcula de forma experimental, siendo muy diferente para distintos problemas.
- Es necesario calcular la distancia desde el caso a clasificar a todos los casos que forman la base de casos. Se debe establecer una lista con los *k* casos más próximos, lista que se irá creando según se va explorando secuencialmente la base de casos. Si el número de casos de aprendizaje es grande, esta exploración es relativamente lenta; existen métodos, como los "k-d trees" que aceleran la búsqueda, pero requieren de la construcción de una estructura adicional.
- La mayor parte de las implementaciones del algoritmo *k*-NN dan la misma importancia, a efectos de clasificación, a los *k* casos más próximos. Para ellas es tan relevante el caso más cercano como el más lejano.
- Para un mismo problema, el valor de *k* cambia radicalmente si se añaden/retiran elementos a/de la base de casos (aunque estos no añadan nuevos matices).
- No se distingue entre los atributos más relevantes y aquellos cuya información no tiene que ver con la clasificación[1]. Esto conduce al problema de la maldición de la dimensionalidad (tal como se ha comentado en la sección 2.6.2.2).
- Para los casos nuevos a clasificar, no se ofrece una información (ni siquiera heurística) sobre la certeza de la predicción que se está realizando.

En esta tesis se va a proponer un nuevo algoritmo: ∞-NN (sección 5.2.4.2), que formará parte del BTW, el cual va a emplear todos los elementos de la base de casos para predecir la clasificación de los casos nuevos, empleará un "*kernel*" de tipo gaussiano para convertir la medida de distancia en una de similitud y optimizará los ejes principales del "*kernel*" para mejorar la exactitud de las predicciones y evitar la maldición de la dimensionalidad. En cierta forma intenta buscar lo mejor de las técnicas de *k*-NN y las redes neuronales de tipo RBF.

Se diferencia de las redes neuronales basadas en RBF en que no existen los pesos que unen el nivel de las neuronas del nivel radial con el de salida (los

[1] Exceptuando la versión de Lowe y otras similares.

casos ponderan en función de su importancia o peso relativo), y sobre todo se hace especial hincapié en buscar un radio óptimo para el "*kernel*" gaussiano, tema casi siempre olvidado en los libros que tratan de este tipo de redes neuronales.

5.1.2 La orientación de la métrica

Se comenzará recordando la expresión que permite calcular la distancia entre dos casos en una métrica euclídea:

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{\Sigma} (\boldsymbol{x}_i - \boldsymbol{x}_j) \tag{5.1}$$

donde $\boldsymbol{\Sigma}$ es la matriz identidad (esto es $\boldsymbol{\Sigma} = \boldsymbol{I}$). En esta métrica la contribución de los distintos atributos es idéntica y el lugar geométrico de los puntos que equidistan de uno dado son superficies esféricas (ver Figura 5.1).

Esta métrica se puede "flexibilizar" de dos formas:

- Generalizando $\boldsymbol{\Sigma}$ para que sea una matriz diagonal, con los elementos de esta diagonal mayores o iguales a 0. Las contribuciones a la distancia de los distintos atributos es independiente y las curvas de los puntos equidistantes a uno dado son elipsoides de ejes paralelos a los coordenados (ver Figura 5.2).

  En la discusión de las ventajas e inconvenientes del algoritmo de Lowe (sección 4.4.1) se indican las consecuencias de optimizar una matriz diagonal.

- O proponiendo que $\boldsymbol{\Sigma}$ pueda ser una matriz simétrica completa cuyos elementos individuales puedan ser parámetros para una posterior optimización[2]. De esta forma se pueden obtener métricas muy flexibles que permitan ponderar "distancias" entre los diversos atributos, evitar sus correlaciones, etc. Las curvas de equidistancia a un punto serán elipsoides, y ahora sus ejes principales pueden estar orientados en cualquier dirección (ver Figura 5.5).

  Como se ha comentado en la sección 4.1.4.2 para las métricas de tipo Mahalanobis, el número de elementos independientes de la matriz $\boldsymbol{\Sigma}$ es $A \,.\, (A+1)/2$ (donde A es la dimensión del espacio de atributos). Si se intenta realizar una optimización por alguno de los métodos relatados en la sección 3.1.1, lo más probable es que se obtenga una solución que presente fuerte sobreaprendizaje y que, por lo tanto, generalice muy mal[3].

---

[2] Se recuerda que $\boldsymbol{\Sigma}$ debe ser una matriz simétrica semidefinida positiva.

[3] El algoritmo LMNN (ver sección 4.4.5) emplea una programación semidefinida para evitar este problema.

Ilustrando con detalle el primero de los escenarios (métrica euclídea), en la Figura 5.1 se muestra un problema sintético donde los casos pertenecen a dos clases (círculos y triángulos) y se distribuyen de forma uniforme en tres bandas verticales, sin mezclarse (la banda central para los triángulos y las dos bandas laterales para los círculos). Si se desease identificar la clase de un caso próximo a la frontera, como el resaltado en dicha Figura, mediante un algoritmo de tipo *k*-NN basado en una métrica euclídea, se obtendrían distintas clasificaciones dependiendo del valor de *k*.

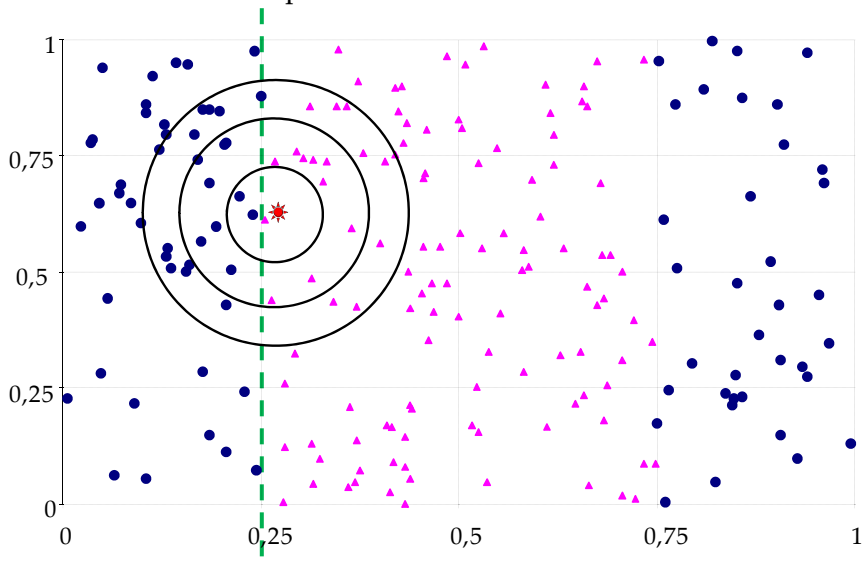


Figura 5.1.- Curvas de puntos que equidistan de uno dado en la métrica euclídea.

En el segundo de los escenarios, modificando la métrica empleada en la clasificación para que sea similar a:

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T \begin{bmatrix} 200 & 0 \\ 0 & 1 \end{bmatrix} (\boldsymbol{x}_i - \boldsymbol{x}_j) \qquad (5.2)$$

se obtendrán unas curvas similares a las que se muestran en la Figura 5.2.

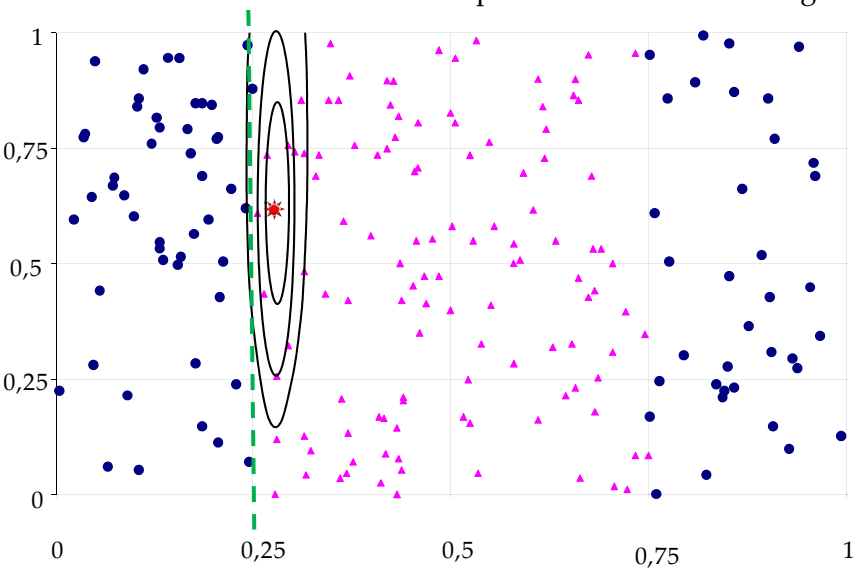


Figura 5.2.- Curvas de puntos equidistantes para una métrica basada en una matriz diagonal como la indicada en la ecuación (5.2).

Estás curvas de nivel proceden de unas funciones conocidas como "*kernels*" (ver sección 4.2.3.5 para una definición más rigurosa de lo que es una función "*kernel*"). En las siguientes Figuras se pueden apreciar los "*kernels*" correspondientes a una métrica con matriz unitaria y con una matriz diagonal.

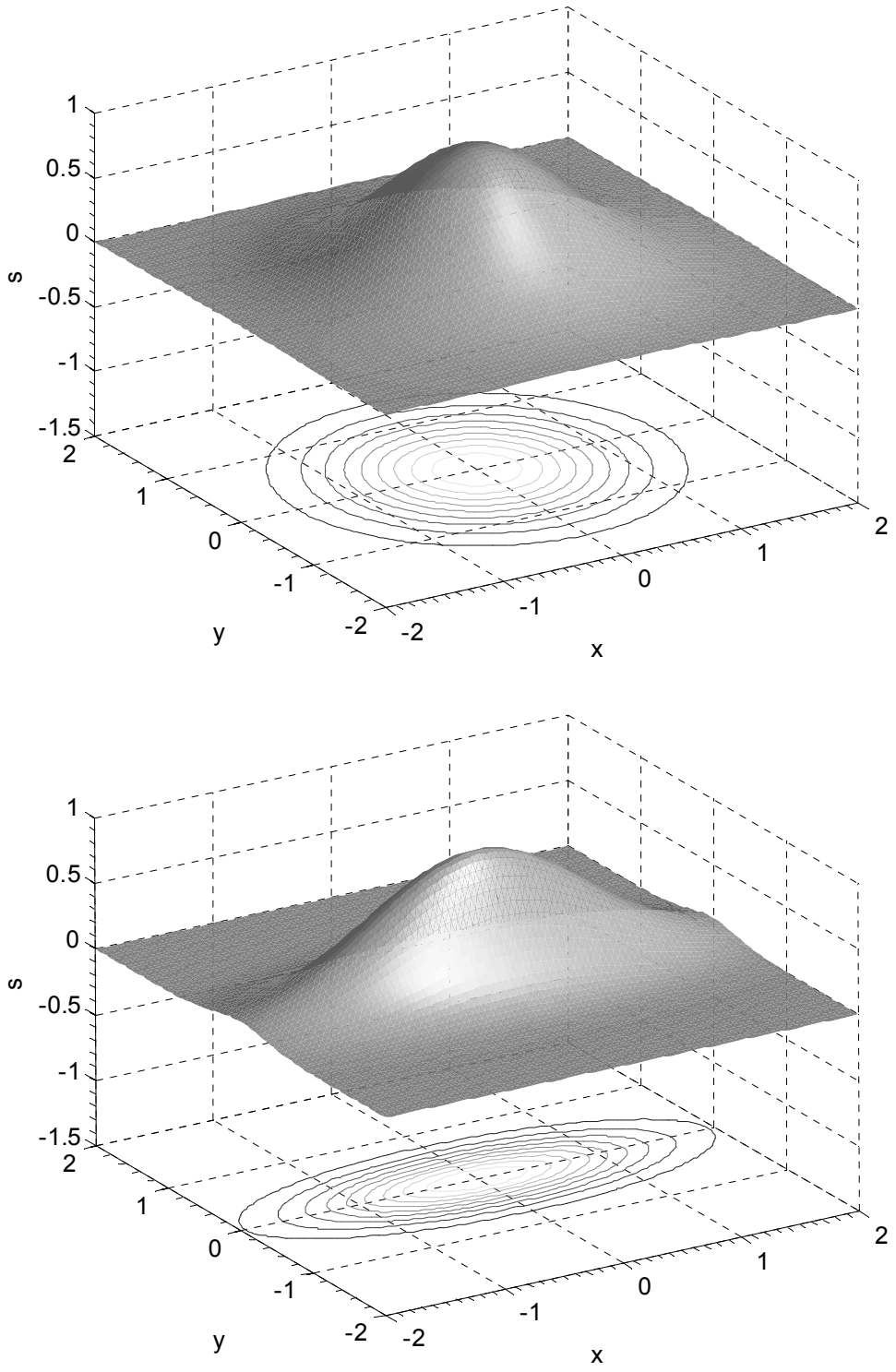


Figura 5.3.- "Kernels" relacionados con una métrica. El primero correspondería a una métrica euclídea, el segundo a uno en que las relevancias de los atributos en los ejes X e Y serían diferentes.

Pero basándose en una matriz diagonal solo es posible modificar las dimensiones relacionadas con los ejes paralelos al sistema de coordenadas. No es posible "inclinar" estas curvas de nivel (y sus correspondientes "*kernels*"). En el tercero de los escenarios sí será posible.

Consideremos otro problema sintético de clasificación en el que los casos se distribuyen en los dos semiplanos delimitados por una línea diagonal, tal como se muestra en la Figura 5.4. Si la diagonal estuviese inclinada exactamente 45 grados, la solución óptima basada en una matriz diagonal (segundo escenario) ofrecería como mejor solución la misma que la métrica euclídea.

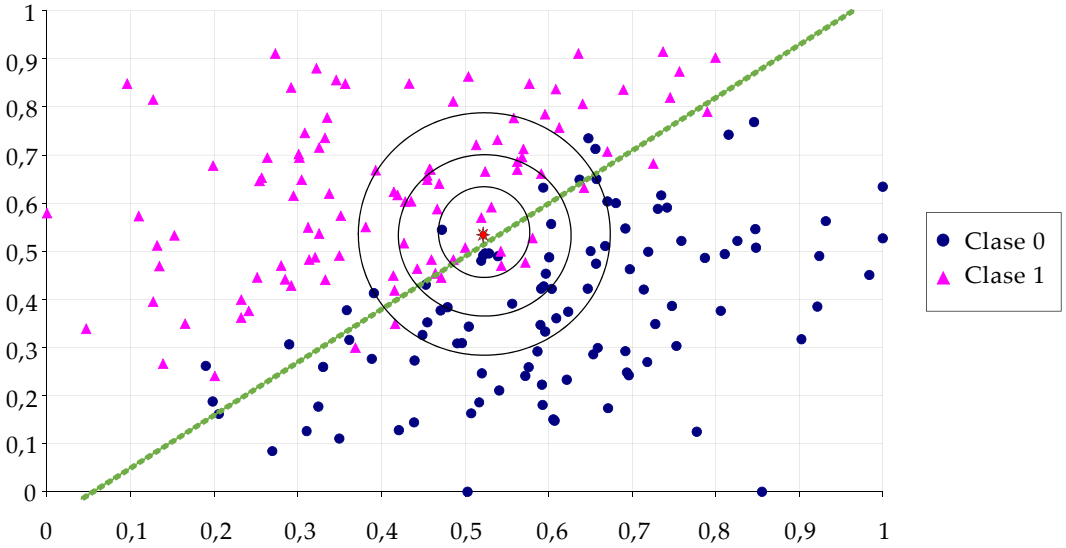


Figura 5.4.- Curvas de nivel en una métrica euclídea para el problema de la distribución en diagonal de los casos.

En cambio, en el tercer escenario, si se adoptase una métrica basada en una matriz simétrica en la que se pudiesen modificar sus tres elementos[4], tal como la que se muestra en la ecuación (5.3), sí sería posible "inclinar" las curvas de equidistancia a un punto dado para que sean parecidas a las que se pueden ver en la Figura 5.5.

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T \begin{bmatrix} 10{,}5 & -9{,}5 \\ -9{,}5 & 10{,}5 \end{bmatrix} (\boldsymbol{x}_i - \boldsymbol{x}_j) \qquad (5.3)$$

[4] Estamos en un problema de dos dimensiones, por lo tanto el número de elementos potencialmente distintos en una matriz simétrica es tres.

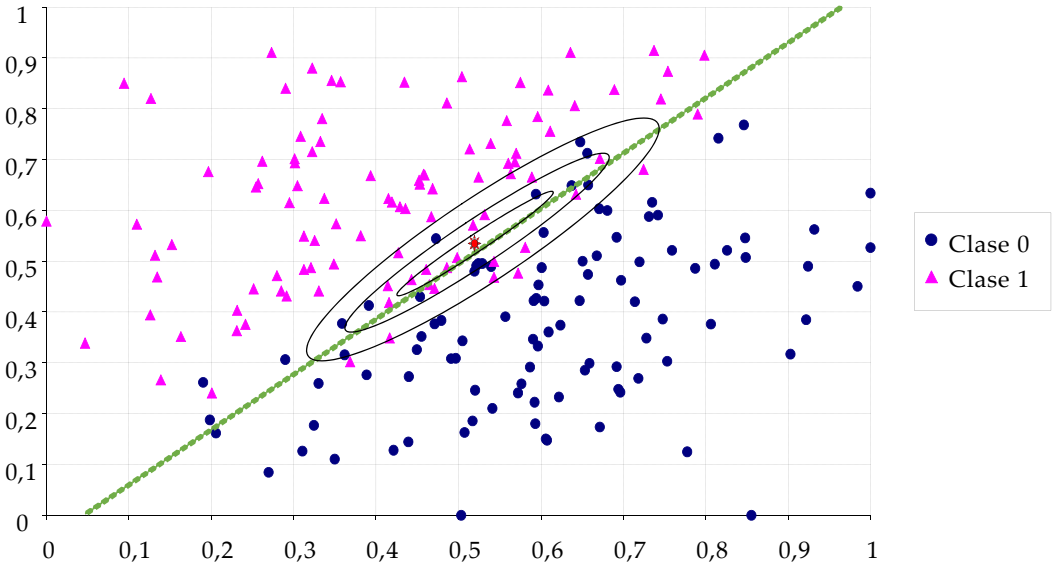


Figura 5.5.- Curvas de nivel para una métrica "orientada" para el problema de la distribución en diagonal de los casos.

El reto que se plantea la métrica BTW es conseguir poder "orientar" la medida de distancias en cualquier dirección sin tener que estimar/optimizar todos los elementos de la matriz **Σ**, y evitando así el sobreaprendizaje.

5.2 El algoritmo BTW

En esta tesis se ha desarrollado un nuevo algoritmo al que se le ha puesto por nombre BTW. En él se propone transformar los atributos originales a unos nuevos ejes orientados de tal forma que separen las clases de acuerdo con la lógica del discriminante lineal LDA. Mediante una variación de la técnica *k*-NN, que optimiza la influencia de cada vecino en función de la distancia a la que se encuentra, se determinará la clase a la que pertenece un nuevo caso a ser clasificado.

Esta sección comenzará indicando el objetivo y el enfoque global del algoritmo BTW, a continuación se realizará un breve repaso de las influencias en las que se basa, para continuar con una explicación detallada de su derivación e implementación. Por último se mostrarán los resultados obtenidos al aplicarlo a once problemas típicos de clasificación.

5.2.1 Objetivo

Desarrollo de una métrica global para clasificadores de tipo *k*-NN que pueda ser usada en aplicaciones que funcionen en tiempo real, mejorando la escalabilidad del algoritmo *k*-NN con respecto al número de casos de aprendizaje. Además debe ser competitivo con *k*-NN, LDA y QDA (sus inmediatos predecesores) en cuanto a la precisión de clasificación.

5.2.2 Enfoque

Busca la dirección más discriminante, desde una perspectiva de tipo LDA, y plantea un método de clasificación de tipo *k*-NN con un radio de vecindad optimizado, operando con los casos proyectados sobre esta dirección.

5.2.3 Trabajos relacionados

La idea del algoritmo BTW [139] [140] tiene sus raíces tanto en el algoritmo DANN de Hastie y Tibshirani [61] (sección 4.4.2), y su posterior revisión por Peng et al. [105], como en el algoritmo VSM de Lowe (sección 4.4.1).

El algoritmo DANN es un método de búsqueda de una dirección discriminante basada en la vecindad local del caso a clasificar (y siguiendo un proceso iterativo se estabiliza el vecindario que proporciona esta dirección). Para ello se utiliza la siguiente expresión (copiada de la ecuación (4.73)):

$$\boldsymbol{\Sigma} = \boldsymbol{W}^{-1/2} \, (\boldsymbol{B}^{*} + \epsilon \boldsymbol{I}) \, \boldsymbol{W}^{-1/2} \tag{5.4}$$

donde $\boldsymbol{W}$ (A x A) es la matriz promediada de las covarianzas de cada clase, $\boldsymbol{B}$ es la matriz de covarianzas entre clases en el espacio original y $\boldsymbol{B}^{*} = \boldsymbol{W}^{-1/2} \, \boldsymbol{B} \, \boldsymbol{W}^{-1/2}$ es la matriz de covarianza entre clases en el espacio transformado esféricamente.

Dichos autores calculan localmente las matrices $\boldsymbol{B}$ y $\boldsymbol{W}$ basándose exclusivamente en los vecinos próximos al que se desea clasificar, es pues un algoritmo de tipo "lazy learning" con métrica local.

El algoritmo VSM es un método de escalado de los ejes coordenados del espacio de atributos que optimiza una función del error cuadrático empírico. Mediante ello se consigue ponderar la relevancia relativa de los distintos atributos. Es un algoritmo en el que la matriz de la métrica es calculada durante la fase de aprendizaje y que se encuadra dentro de la tipología de métricas globales.

Para el desarrollo del algoritmo BTW se ha heredado la propuesta básica de la dirección discriminante del algoritmo DANN y la optimización de la función MSE del algoritmo VSM. Para evitar la gran variabilidad de las direcciones discriminantes del DANN, BTW propone emplear una métrica global (en vez de una local) y al mismo tiempo optimizar el radio del vecindario. En cuanto al algoritmo VSM, el BTW le aventaja en la capacidad de orientar la dirección discriminante en otros ejes que no sean los coordenados.

5.2.4 Enfoque de resolución

Para explicar el desarrollo del algoritmo BTW lo mejor es recurrir a su descomposición en las tres etapas que se enumeran a continuación:

- En primer lugar se establece una transformación de coordenadas que permita proyectar los valores de los atributos a unos nuevos ejes que faciliten la discriminación (separación) entre clases.
- A continuación se establece una función que permita pasar del concepto de "distancia" al de "similitud", extendiendo así la estrategia del algoritmo *k*-NN a ∞-NN y ponderando la influencia clasificatoria de un caso de aprendizaje en forma inversa a la distancia a la que se encuentra del caso a clasificar.
- Por último, se procede con una fase de optimización que calcula cuáles son los pesos óptimos para la métrica en los nuevos ejes.

5.2.4.1 Orientación de los ejes principales de la métrica en las direcciones de mayor discriminabilidad entre clases

El primer paso, se basará en orientar los ejes principales de la métrica en las direcciones que ofrezcan la mayor capacidad discriminante (de acuerdo con la lógica del algoritmo LDA).

Para ello se partirá de la métrica propuesta por el algoritmo DANN, pero ahora aplicada a todo el espacio de atributos:

$$\boldsymbol{\Sigma} = \boldsymbol{W}^{-1/2} \left(\boldsymbol{W}^{-1/2}\, \boldsymbol{B}\, \boldsymbol{W}^{-1/2} \right) \boldsymbol{W}^{-1/2} \qquad (5.5)$$

El rango de la matriz $\boldsymbol{\Sigma}$ es igual al de $\boldsymbol{B}$, porque $\boldsymbol{\Sigma}$ se obtiene premultiplicando y postmultiplicando $\boldsymbol{B}$ por una matriz invertible. El subespacio nulo de $\boldsymbol{\Sigma}$, $\mathcal{N}(\boldsymbol{\Sigma})$, es de dimensión $A - J$ ya que el rango de $\boldsymbol{B}$ es J. Así pues, en problemas de clasificación con dos clases, la dimensión de este subespacio va a ser elevada.

Basándose en el enfoque del método LDA, en los problemas de dos clases existe una única dirección principal que presenta la mayor separabilidad entre ambas. Esta dirección es la del vector singular correspondiente al valor singular no nulo de $\boldsymbol{\Sigma}$. En problemas con J clases, $\boldsymbol{\Sigma}$ tendrá J direcciones singulares. Como quiera que cada dirección principal está asociada a un valor singular no nulo, se pueden retener solamente las m direcciones correspondientes a los m primeros valores singulares (los más grandes, es decir, los que contienen la mayor información sobre la separabilidad entre clases).

Así pues, se procederá a realizar un análisis de componentes principales de la matriz $\boldsymbol{\Sigma}$, obteniendo sus direcciones singulares. La matriz $\boldsymbol{\Sigma}$ es real, cuadrada, simétrica y presenta un gran subespacio nulo (como se ha comentado, para dos clases, este subespacio será de dimensión A-1); para realizar la diagonalización se empleará el método SVD ("Singular Value Decomposition"):

$$\boldsymbol{\Sigma} = \boldsymbol{U}\,\boldsymbol{\sigma}\,\boldsymbol{V}^{\boldsymbol{T}} \tag{5.6}$$

donde:

$\boldsymbol{U}$: es una matriz ortonormal cuyas columnas son los vectores singulares correspondientes a cada uno de los valores singulares de σ.

$\boldsymbol{\sigma}$: es una matriz diagonal $\boldsymbol{\sigma} = diag\,\{\sigma_i\},\ i = 1 \ldots A$, cuyos valores son reales, mayores o iguales que cero, son los llamados valores singulares. El método de cálculo los proporciona habitualmente ordenados, es decir $\sigma_1 \geq \sigma_2 \leq \sigma_3 \leq \cdots$

$\boldsymbol{V}$: es otra matriz ortonormal que, en este caso (por ser $\boldsymbol{\Sigma}$ matriz real y simétrica), coincidirá con $\boldsymbol{U}$.

Geométricamente, los valores y vectores singulares representan la longitud y dirección de los semiejes principales del elipsoide $\boldsymbol{\Sigma}\,.\,\boldsymbol{x}$ que se obtiene al ir moviendo un vector genérico $\boldsymbol{x}$ a lo largo de una esfera de radio unitario e ir multiplicando la matriz $\boldsymbol{\Sigma}$ por dicho vector $\boldsymbol{x}$; es decir, es la imagen de una esfera cuando se le somete a la transformación $\boldsymbol{\Sigma}$. En esta tesis interesa marcar la dirección de máxima discriminación entre clases de $\boldsymbol{\Sigma}$, que se corresponderá con la dirección del vector singular que tiene el mayor valor singular.

Así pues, la obtención de los vectores singulares mediante un algoritmo que los ordene por sus valores singulares (de mayor a menor), permite identificar el primero (el vector más relevante) y determinar que esta será la dirección principal de la métrica BTW. El resto de vectores singulares estarán en un plano perpendicular a dicha dirección (ya que los vectores singulares son ortonormales).

Una vez obtenidas las nuevas direcciones, el vector de atributos originales de cualquiera de los casos puede ser proyectado sobre estas direcciones mediante la siguiente transformación matricial:

$$\boldsymbol{z}_1 = \boldsymbol{U}_1^{-1}\boldsymbol{x} = \boldsymbol{U}_1^T\,\boldsymbol{x} \tag{5.7}$$

Si se cumpliesen las condiciones de que las poblaciones de las dos clases siguiesen una distribución normal multidimensional y tuviesen la misma matriz de covarianzas, la única coordenada relevante del resultado de esta transformación sería la primera, ya que es la que se corresponde con el primer valor singular, y el resto de coordenadas pertenecerían al subespacio nulo (que no ofrece ninguna información de interés). Como para los problemas que se estudian en el mundo real no se van a cumplir estas condiciones, se ha pretendido buscar alguna otra relación entre los atributos proyectados en ese espacio nulo que ayude a mejorar el rendimiento de la clasificación. Para ello se ha procedido a repetir todo el método anterior partiendo ahora de los atributos proyectados en el subespacio de dimensión *A*-1,… y repetir este procedimiento sucesivamente hasta llegar a agotar las *A* dimensiones.

Todo este trabajo de transformación de coordenadas se puede resumir en una única transformación si se calcula:

$$\boldsymbol{U}_{final} = \boldsymbol{U}_1\,\boldsymbol{U}_2\,\boldsymbol{U}_3\,\ldots\boldsymbol{U}_A \tag{5.8}$$

donde, en este caso:

$\boldsymbol{U}_i$ es una matriz de dimensiones *A* x *A*, donde las primeras *i*-1 filas y columnas se corresponderían con los elementos de una matriz identidad y las columnas del bloque restante (de dimensión (*A*-*i*+1) x (*A*-*i*+1)) estarán formadas con las direcciones singulares obtenidas al realizar la transformación SVD de los atributos ya referidos en ese subespacio:

$$\boldsymbol{U}_i = \begin{bmatrix} \boldsymbol{I} & 0 \\ 0 & [\boldsymbol{U}_{subespacio}] \end{bmatrix} \tag{5.9}$$

Así pues, la matriz $\boldsymbol{U}_{final}$ se genera multiplicando las $\boldsymbol{U}_i$, *i*=1,…,*A* (sucesivas matrices de transformación individuales).

Al final, los atributos expresados en las coordenadas originales serán transformados en las nuevas "coordenadas BTW" mediante:

$$\boldsymbol{z} = \boldsymbol{U}_{final}^T\,\boldsymbol{x} \tag{5.10}$$

5.2.4.2 Función de similitud

Una vez representados los datos en las nuevas coordenadas, se propone clasificar el nuevo caso por medio de un método que he denominado ∞-NN en el que todos los casos, debidamente ponderados en forma inversa a su distancia al caso a clasificar, contribuyan a dilucidar la clase resultante.

Para la ponderación se elige una curva gaussiana centrada en el valor cero y cuya desviación estándar en cada dimensión sea ajustable mediante un conjunto de pesos $\{\omega_a\}$.

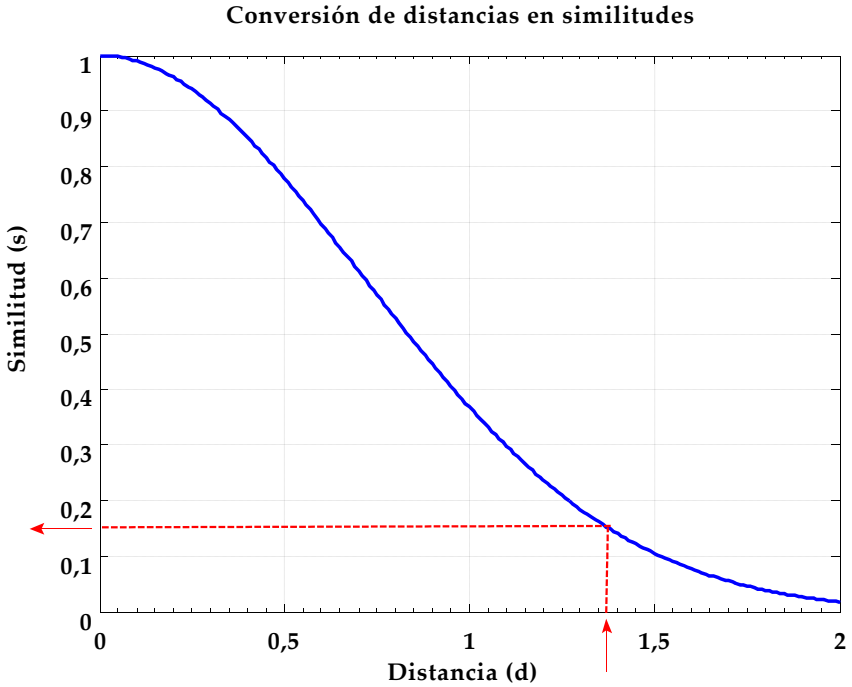


Figura 5.6.- Estrategia propuesta en esta investigación para pasar del concepto de distancia al de similitud. En esta figura se muestra una curva gaussiana unidimensional.

Si se emplease el método ∞-NN, la clase resultante para un determinado caso sería simplemente aquella que más "votos ponderados" tenga.

En tiempo de procesamiento y necesidad de almacenamiento no supone ningún inconveniente frente al tradicional método *k*-NN; incluso es más rápido, ya que para cada caso solo es necesario ir acumulando las similitudes recién calculadas y no se precisa mantener una cola de prioridad con la distancia (o similitud) de los *k* casos más cercanos. Tampoco será necesario realizar sucesivas iteraciones para conocer el valor óptimo para *k*.

Ni siquiera representa un contratiempo si se desea implementar una técnica que no emplee todos los casos prototipo (tipo "k-d tree"), ya que aquellos casos que estuvieran a mucha distancia del nodo a clasificar apenas ponderarían a favor de su clase.

5.2.4.3 *Empleo del método ∞-NN en el algoritmo BTW*

Así pues, la "similitud" entre el caso actual, *i*-ésimo, y el *m*-ésimo será:

$$sim_{\infty-NN}(\mathbf{z}_i, \mathbf{z}_m) = e^{-dist^2_{\infty-NN}(\mathbf{z}_i, \mathbf{z}_m)} = e^{-(\mathbf{z}_i - \mathbf{z}_m)^T \, \mathbf{\Omega}_{diag} \, (\mathbf{z}_i - \mathbf{z}_m)} \quad (5.11)$$

donde $\mathbf{\Omega}_{diag}$ es una matriz diagonal con los pesos atribuibles a cada atributo en las nuevas coordenadas:

$$\mathbf{\Omega}_{diag} = \begin{bmatrix} \omega_1 & 0 & \cdots & 0 \\ 0 & \omega_2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \omega_A \end{bmatrix} \quad (5.12)$$

Los casos más próximos presentarán una similitud cercana a la unidad (al ser su distancia casi cero), mientras que para los lejanos su similitud se acercará prácticamente a cero y no contribuirán a decidir la clase del caso actual.

La decisión final de clasificar el caso actual *i* en una clase u otra se tomará en función de los ratios entre la suma de similitudes a una determinada clase y la suma total de similitudes, por ejemplo, para la clase *j*=1:

$$p_{\infty-NN_1}(\mathbf{z}_i) = \frac{\sum_{n_1=1}^{N_1} sim_{\infty-NN}(\mathbf{z}_i, \mathbf{z}_{n_1})}{\sum_{n_1=1}^{N_1} sim_{\infty-NN}(\mathbf{z}_i, \mathbf{z}_{n_1}) + \sum_{n_2=1}^{N_2} sim_{\infty-NN}(\mathbf{z}_i, \mathbf{z}_{n_2})} \quad (5.13)$$

donde se suman todas las similitudes del caso *i*-ésimo con los N_1 casos de la clase 1 y se divide entre la suma de las similitudes de ese caso *i*-ésimo con todos los casos de las dos clases.

5.2.4.4 *Optimización de los pesos*

Partiendo del ratio de probabilidad de pertenecer a la clase *j*=1, definida en la anterior expresión, se plantea una función de error que luego será objeto de minimización:

$$f_{error}(\mathbf{\Omega}_{diag}) = \sum_{n=1}^{N} \sum_{j=1}^{J} \left(c_j(\mathbf{z}_n) - p_{\infty-NN\,j}(\mathbf{z}_n) \right)^2 \quad (5.14)$$

donde:

El primer sumatorio se extiende por todos los casos de test y el segundo por las *J* clases.

$c_j(\mathbf{z}_n)$ vale 1 si el caso *n*-ésimo pertenece a la clase *j*, siendo 0 en caso contrario.

$p_{\infty-NN\,j}(\mathbf{z}_n)$ es la probabilidad calculada de que el caso *n*-ésimo pertenezca a la clase *j*.

Obviamente, la función de error depende implícitamente de los parámetros de la matriz $\mathbf{\Omega}_{diag}$ (ecuación (5.11)).

Minimizar empíricamente la función de error supone encontrar los pesos óptimos que maximicen la probabilidad de clasificar correctamente todos los casos.

Esta optimización se puede realizar mediante múltiples algoritmos, en esta tesis en un primer momento se lo intentó mediante el algoritmo de Levenberg-Marquardt (sección 3.1.1.4). Los resultados no fueron todo lo satisfactorios que cabría esperar; después de un largo y profundo análisis se observó que en todos los escenarios reales, los pesos correspondientes a los atributos del primer subespacio nulo solo contribuían a empeorar la calidad de la predicción de la clase (el ruido aleatorio que poseían, junto a la correlación que los primeros atributos del subespacio nulo presentaban con el atributo de la dirección principal, conseguían que la minimización generase muchas veces valores erráticos). Eliminándolos se ganaba tiempo y mejoraba la clasificación.

Además, se sabe que el algoritmo de minimización de Levenberg-Marquardt puede quedarse atrapado en un mínimo local, no garantizando de esta forma encontrar el mínimo absoluto.

Como segunda tentativa, y gracias a la posibilidad que se abría de reducir la optimización a un único atributo, se consideró emplear una exploración exhaustiva de todo el rango de variación de solo el primer atributo. Para reducir el número de evaluaciones de la función error, se empleó el algoritmo de búsqueda exhaustiva mediante sección áurea (sección 3.1.1.1), precediéndolo de una rutina que permitía acotar los intervalos donde se producen los mínimos relativos.

Para evitar que se produjera un sobreaprendizaje en la estimación de los pesos, se procedió a emplear una técnica de "bootstrap" (sección 1.3.3.4), en la que se realizaban 200 minimizaciones con datos obtenidos al azar.

Se estudiaron cualitativamente estos resultados, encontrándose que, para un rango muy amplio de pesos, la función de error daba resultados muy similares. En el siguiente gráfico se puede apreciar que en el rango de valores $[0{,}1 \rightarrow 10.000]$ para ω_1, el error es bastante independiente del valor del peso.

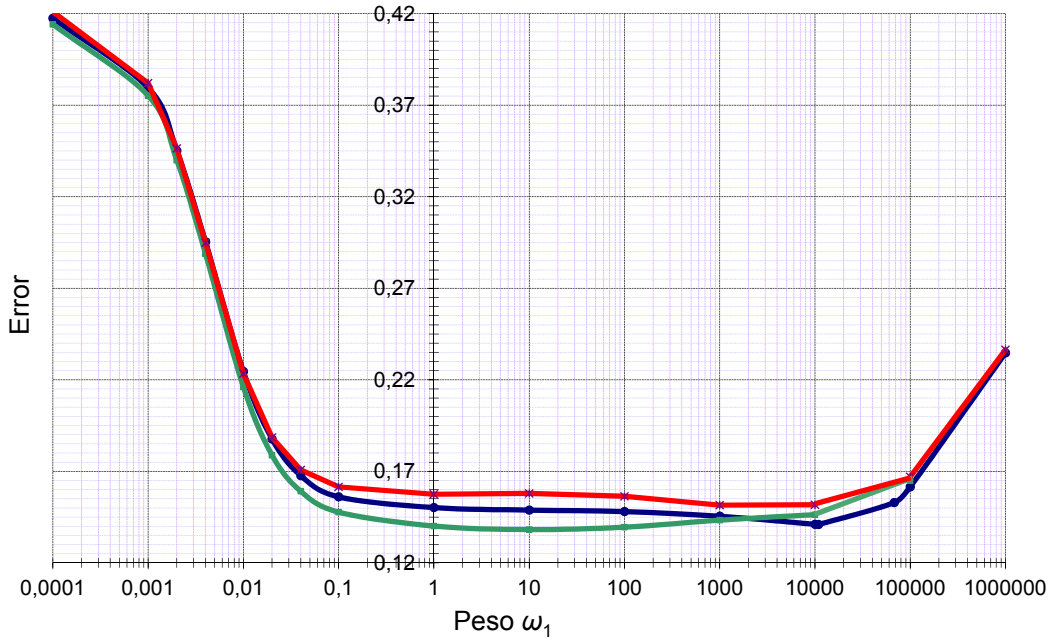


Figura 5.7.- Error cometido en la clasificación en función del peso aplicado (ω_1).

Como a menor valor del peso, mayor es el rango de casos que entran en la evaluación de la probabilidad de pertenecer a una clase, y por tanto más robusta será su predicción, se propone emplear como valor óptimo del peso aquel más próximo a 0 que no sobrepase el mínimo absoluto empírico en más de un 2,5 %. De esta forma se pretende mejorar la capacidad de generalización del algoritmo BTW.

Para terminar con la optimización, se calcula la mediana de los pesos óptimos obtenidos en las 200 réplicas calculadas mediante el método de "bootstrap", y el valor resultante es el que se empleará en el algoritmo de clasificación.

5.3 Implementación

Como se podrá ver en el apartado 5.4, mediante el algoritmo BTW, y basándose en un único atributo que condensa gran parte de la información, se puede establecer una clasificación relativamente fiable.

Pero la gran aportación del algoritmo es que, al depender la clasificación de un solo atributo, se puede acelerar de forma importante la búsqueda de los vecinos más próximos.

Para ello será necesario dividir la aplicación del algoritmo en dos fases: un acondicionamiento previo de los datos (fase "offline"), y otra fase "online" cuando se desee clasificar un caso nuevo.

En la fase "offline" se realizarán las siguientes acciones:

1. Calcular las matrices $\boldsymbol{B}$ y $\boldsymbol{W}$ de acuerdo a las ecuaciones (4.68) y (4.69) para el conjunto de casos de aprendizaje.

2. Construir la matriz $\boldsymbol{\Sigma}$ y obtener sus valores y vectores singulares (ecuación (5.6)). Quedarse únicamente con el primer vector singular, y con él formar la matriz $\boldsymbol{U}_f$ (es una matriz de dimensión A x 1).
3. Transformar los atributos originales de acuerdo a ese vector singular normalizado.

$$z_f = \boldsymbol{U}_f^T \boldsymbol{x} \tag{5.15}$$

   donde:

   $\boldsymbol{x}$ es el vector de atributos en las coordenadas originales.

   z_f es el valor del nuevo atributo (es un escalar), proyectado sobre el eje principal de la métrica.
4. Ordenar los casos de acuerdo a los valores de los atributos obtenidos en el paso anterior (o sea, en la nueva coordenada).
5. Calcular el peso óptimo para la medida de similitud, de acuerdo al método descrito como "segunda tentativa" en la página 164.

Esta será una fase que consume bastante tiempo, pero se lleva a cabo anticipadamente y no influirá en el tiempo necesario para clasificar un nuevo caso en tiempo real.

En el momento en el que se presente un nuevo caso, y haya que clasificarlo de forma rápida, será necesario llevar a cabo (fase "online"):

6. Obtener los atributos del caso actual a clasificar y aplicarles la transformación reflejada en la ecuación (5.15).
7. Quedarse con el valor del atributo en las nuevas coordenadas.
8. Mediante una técnica de búsqueda por bisección en el array ordenado, buscar el caso de aprendizaje cuyo atributo sea lo más parecido al del caso actual. El índice de este elemento será guardado como elemento inicial ("start") y final ("end") de un rango[5].
9. Calcular la medida de similitud entre ambos casos.
10. Tomar del array ordenado el elemento anterior al primero del rango o el posterior al último del rango, el que sea más cercano al actual (esto se puede calcular comparando la diferencia en valor absoluto entre el atributo del caso actual y el de los casos del array). Ajustar el rango por delante o por detrás.
11. Calcular la similitud entre el elemento recién encontrado del array y el caso a clasificar. Con ello, actualizar el cálculo de probabilidad de pertenecer a una de las clases.
12. Tentativamente, considerar que el resto de casos de aprendizaje perteneciesen a la clase actualmente minoritaria y con la misma similitud que la última encontrada.

[5] El rango guarda los índices del primer y el último caso de aprendizaje que son vecinos próximos del caso a clasificar. Un ejemplo de rango sería: [123 131].

13. En este hipotético escenario: ¿cambiaría la clase que actualmente es la más probable?
    - En caso negativo la búsqueda ha terminado (ya que aunque todos los demás casos estuviesen igual de próximos y orientaran la clasificación hacia la otra clase, sería insuficiente para cambiar la decisión).
    - En caso afirmativo el algoritmo continúa, pasándose a ejecutar de nuevo el punto 10 de esta enumeración.

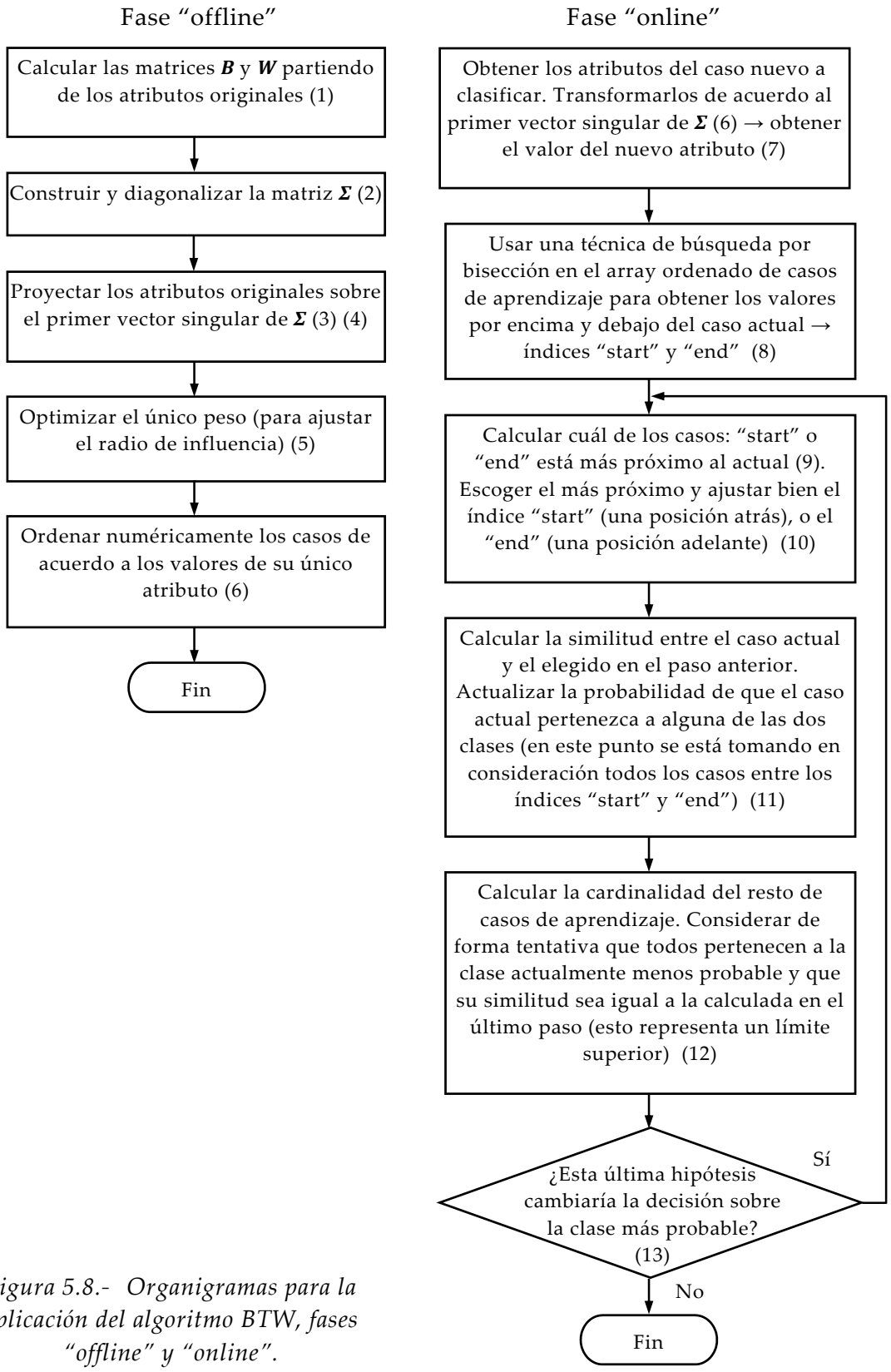


Figura 5.8.- Organigramas para la aplicación del algoritmo BTW, fases "offline" y "online".

Aunque la base de casos posea varios miles, decenas de miles o centenares de miles de instancias, en muy pocas iteraciones el algoritmo encuentra la clasificación deseada (sin tener que calcular la distancia a todos y cada uno de los casos de la base de casos).

La única parte que depende del número de casos de aprendizaje es la búsqueda en el array ordenado y presenta una complejidad de tipo $\mathcal{O}(\log N)$. El resto de operaciones tienen un coste similar al del cálculo de la distancia a otro caso de la base de casos; con la ventaja de que aquí, con el cálculo de unas pocas distancias y similitudes, se establece ya la clase definitiva.

5.3.1 Aplicación en escenarios con múltiples clases

La dirección del espacio de atributos en la que mejor se diferencian dos clases es propia para cada pareja de clases. En este algoritmo, y considerando que el algoritmo subyacente LDA se basa en clases "teóricamente" normales, el combinar todo el resto de clases en una sola para confrontarla a otra ("una vs. resto"), se alejaría conceptualmente más del funcionamiento para el que está diseñado el LDA.

Por esta razón, en este algoritmo se ha elegido una variante del método "una vs. una"[6] que consiste en:

- Obtener la probabilidad de que el caso a clasificar pertenezca a cada una de las clases de cada una de estas parejas.
- Ir sumando dichas probabilidades a una variable totalizadora para cada clase.
- Al final, elegir las dos clases más probables.
- Entre ellas dos decidir cuál es la clase más probable.

[6] Vendría a ser el equivalente a unas elecciones en las que la votación es a dos vueltas.

5.4 Resultados experimentales

Para comprobar la eficacia del algoritmo propuesto se ha experimentado con 11 escenarios tradicionales en el mundo de la inteligencia artificial, obtenidos de la base de datos de la UCI [4] y que se describen en el Anexo A1.

La bondad de la clasificación del algoritmo BTW se ha comparado con sus algoritmos relacionados: LDA, QDA, 1-NN y *k*-NN.

			Tasa de aciertos en %					
	J	***A***	**LDA**	**QDA**	**1-NN**		***k*-NN**[7]	**BTW**
Flores del Iris	3	4	98,00	96,67	96,00	(19)	98,00	97,33
Australian Credit	2	14	86,09	80,72	79,86	(24)	85,94	**86,52**
Breast Cancer	2	9	95,99	95,85	95,14	(8)	97,00	96,85
German Credit	2	24	78,90	79,40	66.10	(16)	71,60	75,50
Glass	6	9	66,36	63,55	73,36	(2)	73,36	**74,30**
Heart	2	13	84,81	87,78	57,41	(25)	68,52	84,81
Image Segmentation	7	19	91,64	48,01	96,71	(1)	96,71	96,10
Satellite Image	6	36	82,45	84,55	89,35	(8)	90,65	87,85
Shuttle Control	7	9	94,53	93,65	99,88	(2)	99,88	97,84
Vehicle Silhouettes	4	18	79,78	91,37	70,45	(6)	72,81	82,39
WaveForms	3	21	88,33	93,67	79,00	(24)	83,00	90,33

Tabla 5.1.- Comparativa de resultados entre el algoritmo BTW y sus relacionados.

Se puede apreciar que el algoritmo BTW se comporta habitualmente de forma intermedia entre el LDA y el *k*-NN, tendiendo a aproximarse al mejor de ellos y en algunos casos superándolo.

Pensamos que este resultado es muy interesante, ya que se ha conseguido un algoritmo de búsqueda de vecinos próximos que no solo clasifica bien, sino que condensa toda la información de un caso en un único atributo, lo cual permitirá, además, trabajar en escenarios que requieran su ejecución en tiempo real.

Así pues, el algoritmo BTW se podrá aplicar en escenarios donde existan un número pequeño de clases y atributos y donde se necesite obtener la clasificación en un tiempo determinista y acotado superiormente. Además comprime la información de muchas fuentes en un único atributo (típico de la fusión sensorial), lo que ayudará a su implementación en equipos sencillos

[7] Los algoritmos 1-NN y k-NN se pueden llevar a cabo con los datos originales o con los datos normalizados de forma que para cada atributo la media sea 0 y la desviación estándar 1. De las dos posibilidades siempre se ha elegido la que mejor clasificación proporcionaba.

(microprocesadores con poca memoria y de bajo coste). Muchas aplicaciones futuras que deban interaccionar en tiempo real con entornos no definidos[8] (conducción automática de vehículos...) tendrán estas características.

5.5 Conclusiones

En el algoritmo BTW, en vez de emplear la tradicional e isótropa métrica euclídea, se plantea realizar una transformación de los datos de acuerdo a la dirección principal de separación de clases proporcionada por el algoritmo LDA, seguida de una clasificación de tipo ∞-NN, donde además se optimiza el radio de influencia para la determinación de la vecindad de un caso.

El problema de trabajar con múltiples clases se resuelve repitiendo la clasificación de un caso bajo cada combinación de parejas de clases y posteriormente quedándose con la clase más frecuentemente repetida ("una vs. una").

El algoritmo presenta dos fases, una primera "offline", que es más costosa en tiempo de ejecución y que, proyecta y prepara los casos de aprendizaje para clasificar los futuros casos nuevos de una forma "online" realmente rápida.

El resultado final es un algoritmo de clasificación donde los atributos quedan fusionados en solo uno, lo cual permite buscar en tiempo real los vecinos de uno dado y por tanto estimar su clase por votación mayoritaria ponderada.

Las prestaciones, en relación a su precisión para clasificar casos nuevos, son competitivas con las de sus dos algoritmos de referencia (LDA y *k*-NN), presentando mejoras en cuanto al tiempo empleado en la búsqueda.

De esta forma se cumple el objetivo indicado en la sección 5.2.1.

En muchos aspectos se podría considerar a BTW un algoritmo de búsqueda aproximada de vecinos, tal como el "locality-sensitive hashing" (LSH) de Andoni et al. [26]. La principal ventaja de BTW frente a LSH es que este último busca un conjunto de líneas para proyectar los datos de forma "ad hoc" para cada problema (particionando un espacio de A dimensiones mediante esferas posicionadas aleatoriamente), mientras que BTW emplea un enfoque bien fundamentado (como el LDA) para esa función de "hashing" que debería ser "sensible a la proximidad". Es más, la métrica BTW se puede considerar que está diseñada para buscar "vecinos próximos preferentemente de su misma clase".

[8] Simulando el comportamiento humano.

6 Propuesta de una métrica local: LOM

Mientras que en la sección anterior se estudiaba el efecto de una métrica global en el comportamiento del algoritmo *k*-NN para la clasificación de casos, en esta se presentarán las aportaciones originales de esta tesis sobre el uso de una métrica de naturaleza local que mejore la precisión de la clasificación.

El algoritmo de clasificación LOM basa la práctica totalidad de su funcionamiento en la métrica del mismo nombre, la cual modifica la medida de distancias en función de la dirección en que se viaja en el espacio de los atributos. Se abandona la métrica euclídea para usar una de tipo riemanniano.

En este capítulo se comienza con la descripción del algoritmo LOM y su métrica, sigue con la implementación detallada del mismo, y finaliza con el análisis de los resultados experimentales obtenidos para un caso sintético. Como colofón se calcula el factor de magnificación del elemento diferencial de volumen que implica el uso de esta métrica.

6.1 El algoritmo LOM

El algoritmo LOM [141] se basa en una métrica que intenta adecuar, de forma local, el concepto de qué puntos están cercanos o lejanos a uno que se desea clasificar. LOM adapta la medida de distancias de forma que estas crezcan más lentamente si la trayectoria que une dos puntos está orientada en la dirección paralela al plano tangente a la frontera de separación entre clases, mientras que esta aumentará más rápidamente en la dirección perpendicular.

La función que proporciona la frontera de separación entre clases es aportada por un algoritmo cualquiera de clasificación de casos (que no hace falta que sea muy preciso) y se puede considerar como un conocimiento "a priori".

La métrica resultante posee solo dos parámetros libres que pueden ser optimizados mediante un procedimiento de validación cruzada[1].

[1] Y que en algún caso se reduce a un solo parámetro.

6.1.1 Objetivo

Desarrollar una métrica para una geometría riemanniana con el objetivo de que al recorrer un segmento de longitud diferencial, la distancia sea diferente según la orientación de ese segmento. Esta métrica permitirá mejorar la bondad del algoritmo *k*-NN al seleccionar un vecindario más óptimo para la clasificación.

6.1.2 Enfoque

La métrica LOM propone una parametrización que conduzca a una nueva geometría para el espacio de atributos, cuyo tensor sea siempre definido positivo, con la intención de que esto mejore el error empírico de clasificación de un algoritmo *k*-NN. Para ello se basa en una función de decisión, obtenida previamente de los datos empíricos, y en la hipótesis de que el coste debe ser mayor al moverse en la dirección del gradiente y menor en el plano perpendicular a él.

6.1.3 Trabajos relacionados

La métrica LOM se fundamenta en la geometría riemanniana desde la perspectiva de orientarla en las direcciones que mejor distinguen una pareja de clases. Esta idea tiene su origen en dos enfoques expuestos en el estado del arte: de una parte VSM (sección 4.4.1) y LaMaNNa (sección 4.4.4), de la otra las transformaciones cuasiconformes (sección 4.4.7).

VSM y LaMaNNA son esencialmente algoritmos de escalado de los ejes de coordenadas. Mientras que VSM aplica esta transformación de forma global, LaMaNNa lo hace localmente. No obstante, tal como se ha comentado en el estado del arte, la idea inicial del segundo de los algoritmos era orientar la geometría en las direcciones marcadas por la superficie de separación de las clases.

Las transformaciones cuasiconformes, basándose en una geometría riemanniana inducida previamente por el "*kernel*", buscan modificar localmente esta geometría para "ensancharla" uniformemente en las proximidades de la frontera de separación de clases.

En relación a estos enfoques, la métrica LOM propuesta aporta:

- A diferencia de VSM y LaMaNNa, una orientación de las direcciones discriminantes, no obligatoriamente según los ejes de coordenadas. Y comparativamente con LaMaNNa, el ser una verdadera métrica y no una función de distancia.
- Frente a los algoritmos basados en transformaciones cuasiconformes, se propone una estrategia de orientación preferencial de las direcciones en las que se ensancha y se contrae la geometría, además

se hace un énfasis especial en orientar el diseño de LOM partiendo de la geometría y finalizando en su formulación matemática.

6.1.4 Descripción de la métrica LOM

En el ámbito de la clasificación de nuevos casos, el primer objetivo de esta investigación ha sido crear una métrica que permita definir qué es, para un punto, "estar cerca" de otro.

Antes de proporcionar una definición formal, quizás sea bueno aportar una visión intuitiva de lo que persigue una métrica de este tipo.

El equivalente a moverse en el espacio de atributos con mayor facilidad en unas direcciones que en otras puede asociarse a las dificultades que tienen los montañeros al moverse en ese medio. Las subidas y bajadas, arroyos… se pueden asociar a mayores esfuerzos (y retrasos) al abordar una ruta. En el lenguaje de la métrica de esta tesis, este mayor o menor esfuerzo se traduciría como "distancias más largas o más cortas".

Imagínese el lector un paisaje de media montaña como el que se aprecia en la Figura 6.1. Cualquiera estará de acuerdo que para ir lo más cómodamente del punto A al B no se sigue una ruta "directa" (en el sentido euclídeo) y se cruza la falda de la montaña (como se indica en la ruta de rayas y puntos), sino que se rodea por la ruta de trazos discontinuos. De hecho las carreteras se trazan siguiendo esta lógica.

Aunque la ruta sea más larga, el "esfuerzo" por la ruta de trazos discontinuos es mucho menor, y a efectos prácticos la "distancia efectiva" es más corta.

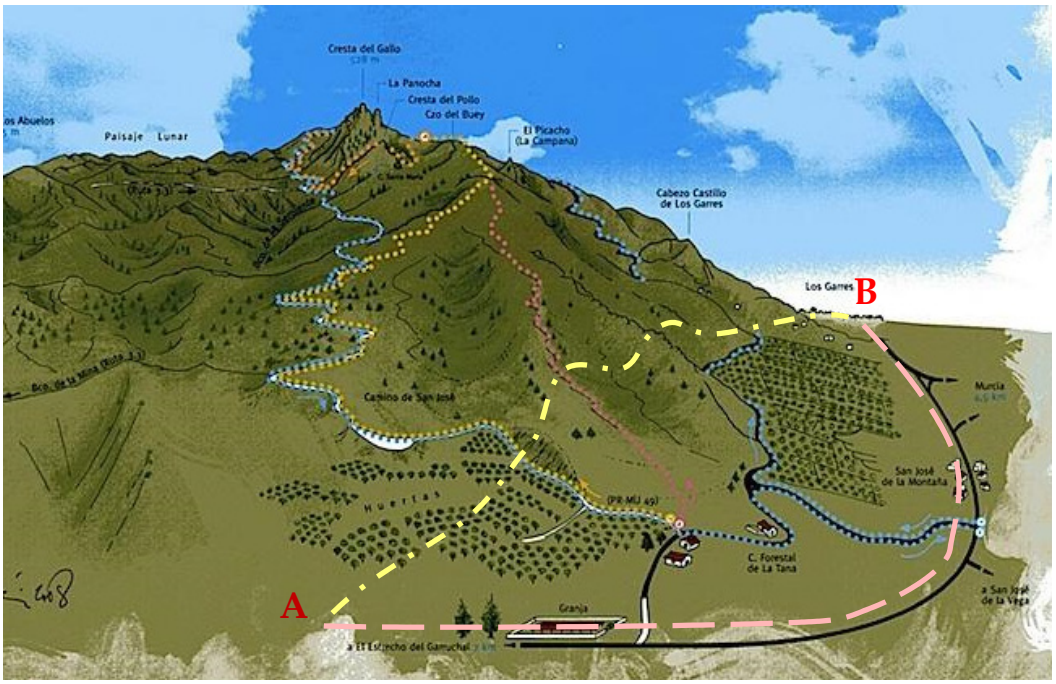


Figura 6.1.- Senderos en una travesía de montaña y búsqueda de la ruta óptima.

No es esta la primera vez que se utiliza la tercera dimensión para explicar el funcionamiento práctico de una métrica riemanniana; en las exposiciones habituales de la teoría de la relatividad, para explicar el efecto de una masa sobre la curvatura de un espacio bidimensional, se recurre también a esta tercera dimensión.

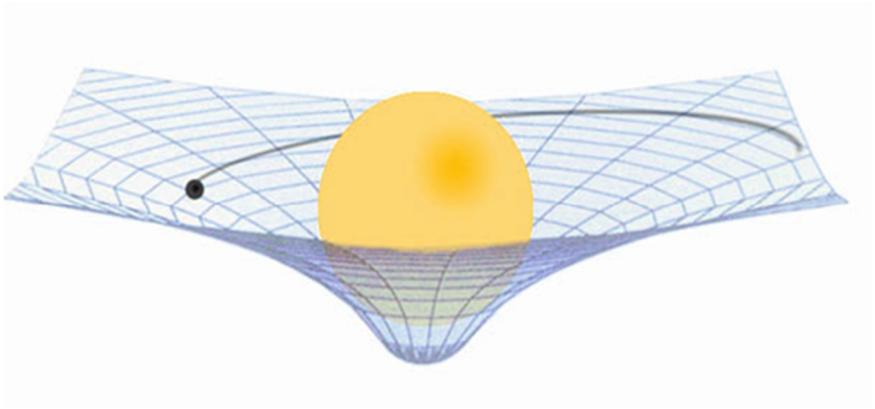


Figura 6.2.- Representación de la distorsión del espacio causado por una masa (dando lugar a una geometría de Riemann) empleando la tercera dimensión.

Volviendo a la problemática de la clasificación (y debido a la distribución peculiar de los datos para un determinado problema) en las proximidades de un caso algunas direcciones dirigirán a zonas donde abundan elementos de su misma clase, mientras que otras apuntarán hacia áreas con elementos mezclados de distintas clases. Determinar dichas direcciones es de fundamental importancia a la hora de proponer una métrica que mida distancias de forma óptima.

La propuesta de esta tesis es una métrica que presente solamente dos grupos de direcciones en un espacio *A*-dimensional:

- Una perpendicular a la superficie que separa las clases. Se propone considerar el gradiente de la función de separación como método válido para determinar esta dirección.
- El conjunto de todas las direcciones perpendiculares a la anterior, es decir, el plano perpendicular al anterior gradiente.

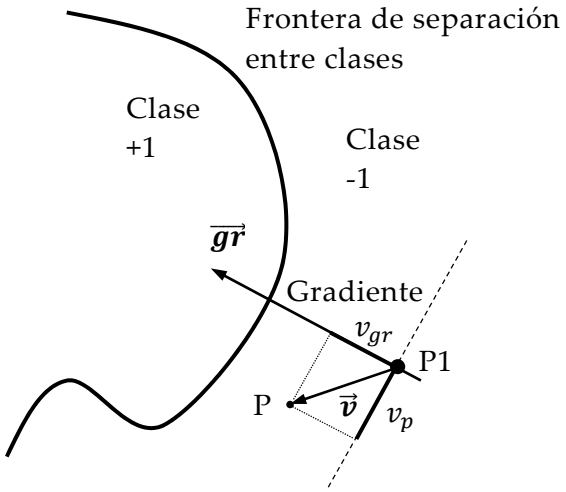


Figura 6.3.- Direcciones principales (o discriminantes) para la métrica LOM en el punto P1.

Posteriormente, a cada uno de estos dos grupos de direcciones se le asignará un peso, de forma que se pueda calcular la "distancia" entre dos puntos en función de una ponderación de las normas de las proyecciones del segmento que une ambos puntos sobre cada uno de estos dos grupos de direcciones. La ecuación que se propone para calcular esta distancia en la nueva métrica es:

$$d^2 = \frac{v_{gr}^2}{r_m^2} + \frac{v_p^2}{r_M^2} \tag{6.1}$$

donde:

v_{gr} es la norma de la proyección del segmento que une los dos puntos (cuya distancia se desea evaluar) $\vec{v}$, sobre la dirección del gradiente de la superficie de separación de clases: $\overrightarrow{gr}$.

v_p es la norma de la proyección del anterior segmento sobre el plano perpendicular al gradiente.

El cálculo de estas dos magnitudes es muy sencillo de realizar. En primer lugar v_{gr}:

$$v_{gr} = \vec{v} \,.\, \overrightarrow{gr} = \langle \boldsymbol{v}, \boldsymbol{gr} \rangle \tag{6.2}$$

donde:

$\overrightarrow{gr}$ es el vector gradiente de la función de separación entre clases (normalizado).

El operador '.' o el operador $\langle\ ,\ \rangle$ representa el producto interior de ambos vectores.

Por otra parte, el cálculo de v_p se realizará mediante la siguiente ecuación:

$$v_p^2 = \|\vec{v}\|^2 - v_{gr}^2 \tag{6.3}$$

Así pues, a la distancia contribuyen las componentes en dirección del gradiente y su perpendicular en forma ponderada, dividiéndolas entre sendos coeficientes denominados r_m y r_M (también elevados al cuadrado para dotarles de significado físico).

r_m es el denominado "radio menor". Viene a reflejar que en la dirección del gradiente, la longitud de la proyección se dividirá entre un coeficiente pequeño; y que, por tanto, cualquier "separación" en esta dirección contribuirá a un gran "alejamiento" entre los puntos.

r_M es el denominado "radio mayor". Sirve para ponderar la norma de la proyección en el plano perpendicular al gradiente. De esta forma, las variaciones en esta dirección contribuirán en menor medida a la distancia.

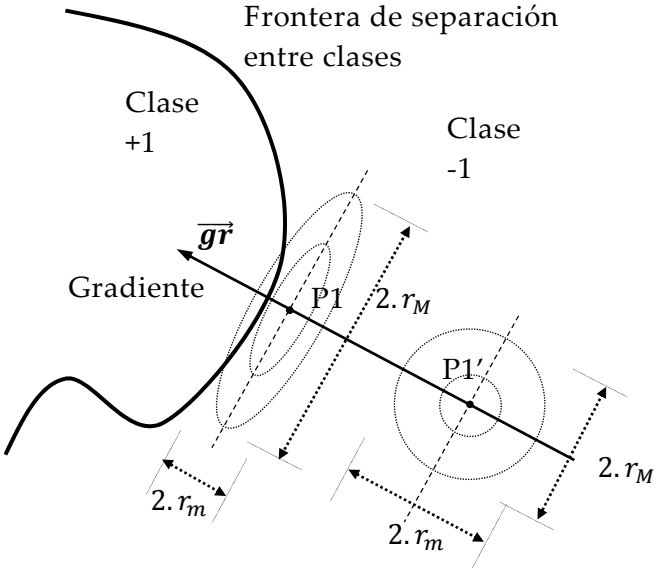


Figura 6.4.- Curvas de puntos equidistantes para los casos P1 y P1′.

En la Figura 6.4 se puede apreciar cómo se modifican las curvas (elipses) que señalan la posición de los puntos equidistantes a uno dado (curvas isométricas) de acuerdo a los valores elegidos para r_m y r_M.

Para no generar un número excesivo de parámetros a proporcionar (o estimar), se va a considerar que los radios mayor y menor para todo el espacio de atributos se obtendrán mediante la siguiente fórmula:

$$r_m = r/\left(1 + \tau\, e^{-f^2(x)}\right)$$
$$r_M = r\left(1 + \tau\, e^{-f^2(x)}\right) \qquad (6.4)$$

donde:

r : es un parámetro a proporcionar (u optimizar) que fija la escala global de distancias del problema.

τ : es un parámetro a proporcionar (u optimizar) que amplifica el radio mayor y disminuye el radio menor según el punto se aproxima a la frontera de separación entre clases. Sirve para controlar la "excentricidad" deseada para las curvas isométricas según la función de decisión se acerca a la

frontera de separación entre clases. Es un número mayor que 0 y habitualmente en el rango de 1 a 5.

$f(\boldsymbol{x})$: es la función que separa dos clases (y cuyo valor depende del punto $\boldsymbol{x}$ del espacio de atributos donde se evalúe). En la frontera entre clases su valor será 0, y según se aleja de ella tomará valores positivos para una clase y negativos para la otra.

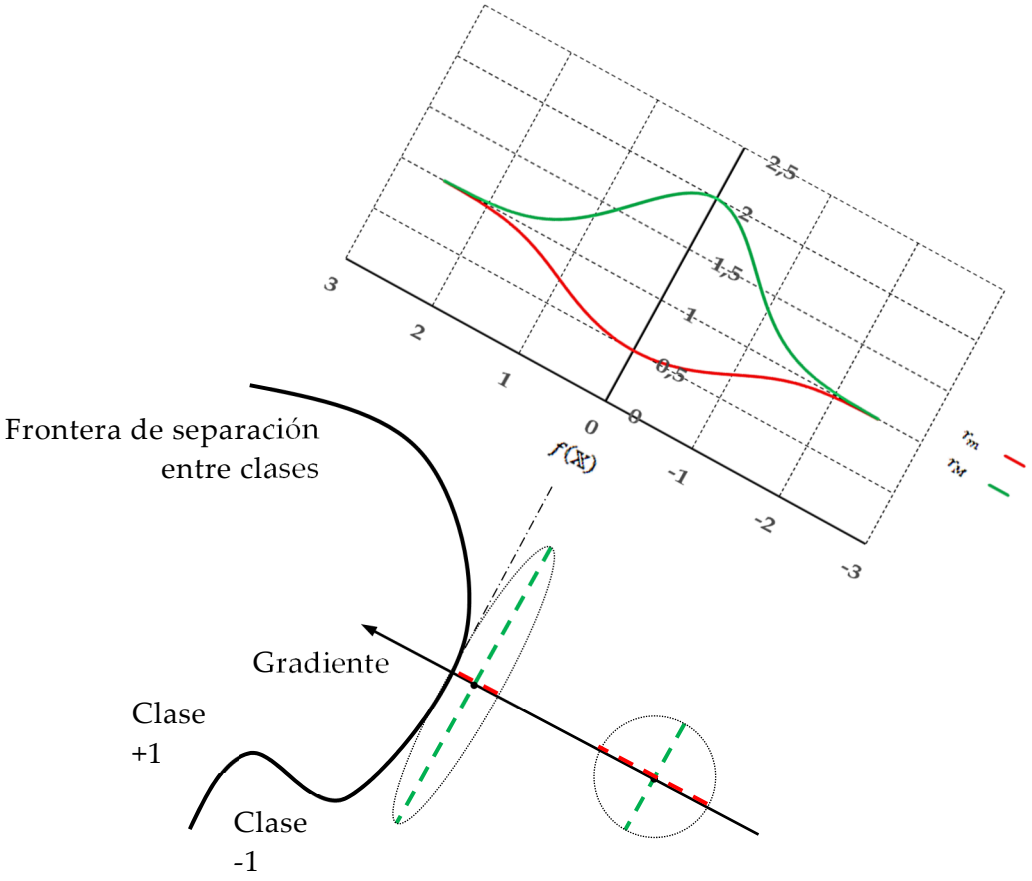


Figura 6.5.- Evolución de los parámetros r_m y r_M según cambia el valor de $f(\boldsymbol{x})$ (para τ=2).

Para ser usada en la métrica LOM no es necesario que la función de decisión devuelva valores acotados en un rango, simplemente sirve con que en la frontera de separación la función devuelva el valor 0, y que su valor absoluto se incremente según se aleja de ella.

Cuando el punto está próximo a la frontera de separación entre clases $f(\boldsymbol{x})$ valdrá 0 y por tanto el radio mayor será $(1+\tau)^2$ veces el radio menor. En aquellas zonas del espacio de atributos donde $f(\boldsymbol{x})$ sea grande en valor absoluto el radio mayor será aproximadamente igual al radio menor.

El factor τ controla el efecto de la excentricidad de las elipses según estas se aproximan al curva de $f(\boldsymbol{x}) = 0$. Tal como se puede apreciar en la Figura 6.6, un valor grande de τ pronuncia los efectos de la métrica LOM generando elipses más excéntricas, mientras que valores pequeños hacen que se parezca más a la métrica euclídea.

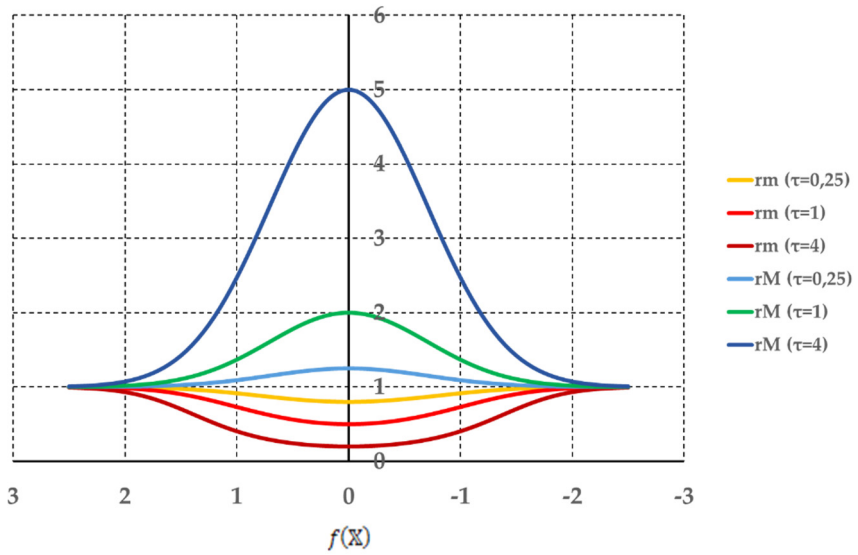


Figura 6.6.- Evolución de r_m y r_M según cambia el valor de τ.

De esta forma, la métrica depende solo de dos parámetros que serán ajustables para cada escenario: r y τ.

La ventaja, y el problema, que subyace a esta métrica es que tanto el valor como la orientación del gradiente de la función de decisión van variando en cada punto del espacio de atributos. Es una ventaja en cuanto a que la métrica se va adaptando localmente a la forma de la superficie de separación de clases; pero se genera el inconveniente de que para calcular correctamente la distancia habría que ir aplicando un cálculo infinitesimal considerando cada elemento diferencial del segmento que une dos puntos.

En muchos estudios del estado del arte se obvia este problema y se considera que el gradiente es único y constante en toda la longitud del segmento que une dos puntos P_i y P_f (inicial y final), lo cual puede ser aceptable si los puntos están muy próximos, pero que deja de serlo cuando estos se alejan entre sí o la superficie de decisión varía muy rápidamente (en relación con la distancia que separa a ambos).

Otro problema de estos planteamientos es que, al considerar un único valor del gradiente, la distancia entre el punto P_i y P_f deja de ser simétrica: distancia (P_i, P_f) ≠ distancia (P_f, P_i). Con lo cual la medida de la distancia deja de ser una métrica; e incluso los "*kernels*" derivados de esta simplificación no cumplen con la condición de dar lugar a matrices definidas positivas. Por lo tanto no le son aplicables las condiciones requeridas para su uso en los habituales algoritmos de optimización SMO de las máquinas de soporte vectorial. A pesar de todo, la racionalidad de su uso se basa en que cuando dos puntos están muy separados su distancia es generalmente muy grande, su similitud tiende a 0 y por tanto la contribución que aportan en el algoritmo de clasificación es pequeña.

La métrica LOM no sufre estos problemas ya que emplea distintos valores del gradiente a lo largo del recorrido que une los puntos.

6.1.5 Propiedades de la métrica LOM

Es bien conocido que la norma elevada al cuadrado de un elemento diferencial (elemento de línea) en un espacio A-dimensional euclídeo tendrá una representación tal como:

$$ds^2 = dx_1^2 + dx_2^2 + \cdots + dx_A^2 \tag{6.5}$$

El correspondiente vector para dicho elemento diferencial será:

$$\boldsymbol{ds} = (dx_1, dx_2, \ldots, dx_A)^T \tag{6.6}$$

Si el gradiente de la función de decisión $f(\boldsymbol{x})$ (una vez normalizado) en un determinado punto de ese espacio es:

$$\boldsymbol{gr} = \frac{\nabla f(\boldsymbol{x})}{\|\nabla f(\boldsymbol{x})\|} = (gr_1, gr_2, \ldots, gr_A)^T \tag{6.7}$$

la proyección del elemento diferencial en la dirección del gradiente será, por tanto, su producto escalar:

$$ds_{gr} = \langle \boldsymbol{ds}, \boldsymbol{gr} \rangle = \sum_{a=1}^{A} gr_a . dx_a \tag{6.8}$$

Y su norma elevada al cuadrado:

$$\begin{aligned} ds_{gr}^2 = gr_1^2 . dx_1^2 + gr_2^2 . dx_2^2 + \cdots + gr_A^2 . dx_A^2 + 2gr_1 . gr_2 . dx_1 . dx_2 + \\ + 2gr_1 . gr_3 . dx_1 . dx_3 + \cdots + 2gr_{A-1} . gr_A . dx_{A-1} . dx_A \end{aligned} \tag{6.9}$$

Expresada en forma matricial resulta:

$$ds_{gr}^2 = (dx_1 \;\; dx_2 \;\; \ldots \;\; dx_A) \begin{bmatrix} gr_1^2 & gr_1 . gr_2 & \ldots & gr_1 . gr_A \\ gr_1 . gr_2 & gr_2^2 & \ldots & gr_2 . gr_A \\ \ldots & \ldots & \ldots & \ldots \\ gr_1 . gr_A & gr_2 . gr_A & \ldots & gr_A^2 \end{bmatrix} \begin{bmatrix} dx_1 \\ dx_2 \\ \ldots \\ dx_A \end{bmatrix} \tag{6.10}$$

La norma elevada al cuadrado de la proyección del elemento diferencial ds sobre el plano perpendicular al gradiente (de forma similar a lo conceptualizado en (6.3)) será:

$$ds_p^2 = ds^2 - ds_{gr}^2 \tag{6.11}$$

Y en forma matricial:

$$ds_p^2 = (dx_1 \;\; dx_2 \;\; \ldots \;\; dx_A) \begin{bmatrix} 1 - gr_1^2 & -gr_1 . gr_2 & \ldots & -gr_1 . gr_A \\ -gr_1 . gr_2 & 1 - gr_2^2 & \ldots & -gr_2 . gr_A \\ \ldots & \ldots & \ldots & \ldots \\ -gr_1 . gr_A & -gr_2 . gr_A & \ldots & 1 - gr_A^2 \end{bmatrix} \begin{bmatrix} dx_1 \\ dx_2 \\ \ldots \\ dx_A \end{bmatrix} \tag{6.12}$$

Para una métrica general, la norma elevada al cuadrado del elemento diferencial de distancia será:

$$ds^2 = (dx_1 \;\; dx_2 \;\; \ldots \;\; dx_A) \; \boldsymbol{G} \begin{bmatrix} dx_1 \\ dx_2 \\ \ldots \\ dx_A \end{bmatrix} \tag{6.13}$$

Para el caso de la métrica LOM, y basándose en las ecuaciones (6.1), (6.10) y (6.12), la matriz $\boldsymbol{G}$ valdrá:

$$\boldsymbol{G} = \begin{bmatrix} \left(\frac{1}{r_m^2} - \frac{1}{r_M^2}\right) gr_1^2 + \frac{1}{r_M^2} & \left(\frac{1}{r_m^2} - \frac{1}{r_M^2}\right) gr_1.gr_2 & \dots & \left(\frac{1}{r_m^2} - \frac{1}{r_M^2}\right) gr_1.gr_A \\ \left(\frac{1}{r_m^2} - \frac{1}{r_M^2}\right) gr_1.gr_2 & \left(\frac{1}{r_m^2} - \frac{1}{r_M^2}\right) gr_2^2 + \frac{1}{r_M^2} & \dots & \left(\frac{1}{r_m^2} - \frac{1}{r_M^2}\right) gr_2.gr_A \\ \dots & \dots & \dots & \dots \\ \left(\frac{1}{r_m^2} - \frac{1}{r_M^2}\right) gr_1.gr_A & \left(\frac{1}{r_m^2} - \frac{1}{r_M^2}\right) gr_2.gr_A & \dots & \left(\frac{1}{r_m^2} - \frac{1}{r_M^2}\right) gr_A^2 + \frac{1}{r_M^2} \end{bmatrix} \quad (6.14)$$

lo cual es equivalente a:

$$\boldsymbol{G} = \overbrace{\left(\frac{1}{r_m^2} - \frac{1}{r_M^2}\right) \begin{bmatrix} gr_1^2 & gr_1.gr_2 & \dots & gr_1.gr_A \\ gr_1.gr_2 & gr_2^2 & \dots & gr_2.gr_A \\ \dots & \dots & \dots & \dots \\ gr_1.gr_A & gr_2.gr_A & \dots & gr_A^2 \end{bmatrix}}^{\boldsymbol{G_1}} + \overbrace{\frac{1}{r_M^2} \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 1 \end{bmatrix}}^{\boldsymbol{G_2}} \quad (6.15)$$

Del análisis de esta matriz $\boldsymbol{G}$ se puede concluir que:

- La matriz resultante es la suma del producto diádico del vector gradiente con sí mismo ($\boldsymbol{G_1}$) con una matriz diagonal ($\boldsymbol{G_2}$).
- El rango de la matriz $\boldsymbol{G_1}$ es 1.
- Los valores propios de la matriz $\boldsymbol{G_1}$ son todos ceros, excepto uno de ellos que vale:

$$\left(\frac{1}{r_m^2} - \frac{1}{r_M^2}\right) \|gr\|^2 \geq 0 \quad (6.16)$$

  Por tanto, el valor propio más pequeño de la matriz $\boldsymbol{G_1}$ es 0.
- El factor $\left(\frac{1}{r_m^2} - \frac{1}{r_M^2}\right)$ tendrá siempre un valor mayor o igual que 0.
- Los valores propios de la matriz identidad son todos 1. Por tanto, los valores propios para la matriz $\boldsymbol{G_2}$ serán todos iguales al factor $1/r_M^2$ (y, consecuentemente, positivos).

De acuerdo al teorema de Weyl [142], el cual indica que si $\mathcal{B}$ y $\mathcal{C}$ son dos matrices simétricas de dimensión A x A con valores propios:

$$\lambda_1(\mathcal{B}) \leq \lambda_2(\mathcal{B}) \leq \dots \leq \lambda_A(\mathcal{B}) \quad \text{y} \quad \lambda_1(\mathcal{C}) \leq \lambda_2(\mathcal{C}) \leq \dots \leq \lambda_A(\mathcal{C})$$

respectivamente, y si los valores propios de la matriz resultante de la suma de ambas matrices $\mathcal{B} + \mathcal{C}$ son:

$$\lambda_1(\mathcal{B} + \mathcal{C}) \leq \lambda_2(\mathcal{B} + \mathcal{C}) \leq \dots \leq \lambda_A(\mathcal{B} + \mathcal{C})$$

se cumple que $\forall i\ 1, \ldots, A$:

$$\lambda_i(\mathcal{B}+\mathcal{C}) \geq \begin{cases} \lambda_i(\mathcal{B}) + \lambda_1(\mathcal{C}) \\ \lambda_{i-1}(\mathcal{B}) + \lambda_2(\mathcal{C}) \\ \ldots \\ \lambda_1(\mathcal{B}) + \lambda_i(\mathcal{C}) \end{cases} \qquad \lambda_i(A+\mathcal{C}) \leq \begin{cases} \lambda_i(\mathcal{B}) + \lambda_A(\mathcal{C}) \\ \lambda_{i+1}(\mathcal{B}) + \lambda_{A-1}(\mathcal{C}) \\ \ldots \\ \lambda_A(\mathcal{B}) + \lambda_i(\mathcal{C}) \end{cases} \tag{6.17}$$

Atendiendo a la desigualdad de la izquierda de la ecuación (6.17), como todos los $\lambda_i(\mathcal{B}) \geq 0$ y $\lambda_i(\mathcal{C}) > 0$, se concluye que todos los valores propios de la matriz $\boldsymbol{G}$ de esta métrica serán positivos, y por tanto la matriz será definida positiva, siendo su rango igual a A.

Pensando de forma alternativa al teorema de Weyl, la ecuación (6.15) puede ser interpretada como la regularización de la matriz singular $\boldsymbol{G}_1$ por medio de la suma de una matriz escalar $\boldsymbol{G}_2$.

Los espacios en $\mathbb{R}^A$ en los cuales la distancia elemental se mide mediante una expresión de la forma (6.13), en las que además $\boldsymbol{G}$ es simétrica, diferenciable al menos dos veces y su determinante es distinto de cero, se llaman espacios de Riemann, y $\boldsymbol{G}$ es su tensor fundamental o métrico (tensor covariante de segundo orden). Así pues, la métrica LOM es riemanniana.

6.1.6 Cálculo de la distancia entre dos puntos

En una métrica general, la longitud exacta de un arco de curva entre dos puntos se calcula mediante:

$$L = \int_{\lambda_i}^{\lambda_f} ds = \int_{\lambda_i}^{\lambda_f} \sqrt{\sum_{i,j} g_{ij} dx^i dx^j} \tag{6.18}$$

realizando una integración paramétrica a lo largo de toda la geodésica que une ambos puntos.

En cualquier métrica, la distancia entre dos puntos se debe medir por el camino más corto. Una geodésica es aquella curva que para dos puntos suficientemente próximos, su longitud es mínima entre todas las curvas que unen esos dos puntos[2].

En la sección 3.2.7 se proporcionan más detalles sobre cómo realizar esta integración a lo largo de una geodésica.

6.1.7 La métrica LOM cuando se usa como función de separación una SVM

La métrica LOM necesita de una etapa previa en la que se selecciona una función de separación (que procede de entrenar un algoritmo de clasificación con los casos de aprendizaje); su objetivo es clasificar un caso nuevo en una de las dos clases posibles. En general es una función que asocia a un punto $\boldsymbol{x}$

[2] También se define como aquella en la cual su curvatura es cero.

del espacio de atributos un valor real. En muchos algoritmos si ese valor es positivo se dice que un caso pertenece a una de las clases y si es negativo a la otra.

$$\mathbb{R}^A \xrightarrow{f(\boldsymbol{x})} \mathbb{R} \tag{6.19}$$

Se va a tomar como algoritmo de clasificación una máquina de soporte vectorial (SVM). De acuerdo a lo expuesto en la sección 4.2.3, la expresión de la función de separación para este algoritmo es (ver ecuación (4.39), en este caso no se ha utilizado el operador signo y, para simplificar, α_i^* representa el el producto $\alpha_i y_i$:

$$f_{decision}\,(\,\boldsymbol{x}_k) = \sum_{i=1}^{nSV} \alpha_i^*\, K(\boldsymbol{x}_k, \boldsymbol{x}_i) + b_0 \tag{6.20}$$

donde:

nSV es el número de vectores de soporte.

$\boldsymbol{x}_k$ es el caso que se desea clasificar (vector A-dimensional).

$\boldsymbol{x}_i$ es el i-esimo vector de soporte.

$K(\boldsymbol{x}_k, \boldsymbol{x}_i)$ es el valor de la función "*kernel*" al aplicarla sobre el punto $\boldsymbol{x}_k$ y el i-esimo vector de soporte.

En esta métrica se empleará un "*kernel*" de tipo RBF:

$$K(\boldsymbol{x},\, \boldsymbol{x}_{SV}) = e^{-\frac{\left\|x_a - x_{SV,\,a}\right\|^2}{\sigma^2}}$$

b_0 es el término independiente que ajusta la función de decisión para que en la frontera entre clases su valor sea 0.

El gradiente (sin normalizar) para esta función de clasificación será:

$$Gr_m = \frac{\partial f(\boldsymbol{x}_k)}{\partial x_m} = \sum_{i=1}^{nSV} \alpha_i^*\; K(\boldsymbol{x}_k, \boldsymbol{x}_i).\left(\frac{-2}{\sigma^2}\right)\left(x_{k,m} - x_{i,m}\right) \tag{6.21}$$

La derivada del gradiente sin normalizar con respecto a x_l:

$$\frac{\partial Gr_m}{\partial x_l} = \sum_{i=1}^{nSV} \alpha_i^*\; K(\boldsymbol{x}_k, \boldsymbol{x}_i).\left(\frac{-2}{\sigma^2}\right)\left[\left(\frac{-2}{\sigma^2}\right)\left(x_{k,m} - x_{i,m}\right)\left(x_{k,l} - x_{i,l}\right) + \delta_{ml}\right] \tag{6.22}$$

Y una vez normalizado:

$$gr_j = \frac{Gr_j}{\sqrt{\sum_{i=1}^{A} Gr_i{}^2}} \tag{6.23}$$

La derivada del gradiente normalizado con respecto a x_l:

$$\frac{\partial gr_m}{\partial x_l} = \frac{\frac{\partial Gr_m}{\partial x_l}\sqrt{\sum_{i=1}^{A} {Gr_i}^2} - \frac{Gr_m\left(\sum_{i=1}^{A} Gr_i \frac{\partial Gr_i}{\partial x_l}\right)}{\sqrt{\sum_{i=1}^{A} {Gr_i}^2}}}{\sum_{i=1}^{A} {Gr_i}^2} \tag{6.24}$$

De acuerdo a la métrica propuesta:

$$\frac{\partial\left(\frac{1}{r_M^2}\right)}{\partial x_l} = \frac{-2}{r_M^3}\frac{\partial r_M}{\partial x_l} \qquad \frac{\partial\left(\frac{1}{r_m^2} - \frac{1}{r_M^2}\right)}{\partial x_l} = \frac{2}{r_M^3}\left(\left[\frac{r_M}{r}\right]^4 + 1\right)\frac{\partial r_M}{\partial x_l} \tag{6.25}$$

donde:

$$\frac{\partial r_M}{\partial x_l} = -2\, r\, \tau\, e^{-f^2(\boldsymbol{x}_k)} f(\boldsymbol{x}_k) Gr_l = -2(r_M - r) f(\boldsymbol{x}_k) Gr_l \tag{6.26}$$

De esta forma es posible calcular las derivadas de los elementos del tensor métrico respecto a las distintas coordenadas:

$$\begin{aligned}\frac{\partial g_{ij}}{\partial x_l} = \frac{2}{r_M^3}\left(\left[\left(\frac{r_M}{r}\right)^4 + 1\right] gr_i\, gr_j - \delta_{ij}\right)\frac{\partial r_M}{\partial x_l} + \\ + \left(\frac{1}{r_m^2} - \frac{1}{r_M^2}\right)\left(gr_i \frac{\partial gr_j}{\partial x_l} + gr_j \frac{\partial gr_i}{\partial x_l}\right)\end{aligned} \tag{6.27}$$

Y con estas derivadas parciales se pueden calcular los símbolos de Christoffel, y la geodésica que une dos puntos.

6.2 Implementación

Calcular la distancia entre dos puntos requiere en primer lugar determinar la curva geodésica que los une. Esto se realiza de forma aproximada calculando un número elevado de puntos intermedios a lo largo de ella, para ello:

- O bien se discretiza en un número igual de partes el segmento que une los dos puntos procediendo a continuación a integrar la ecuación diferencial (3.37).
- O bien se establece una rejilla fina que cubra todo el espacio de atributos y se busca el camino más corto entre los dos extremos del segmento (empleando para medir la longitud de las aristas de la rejilla la métrica LOM).

Una vez que se han determinado los distintos puntos intermedios, se procederá a calcular la distancia de estos pequeños segmentos elementales de acuerdo a la métrica LOM y aplicando la ecuación (6.1)[3]. La suma de las longitudes de estos "microsegmentos" proporciona la distancia entre los dos puntos del espacio de atributos.

6.2.1 Integración aproximada de la ecuación diferencial que permite calcular la geodésica entre dos puntos

La integración de una ecuación tal como la recogida en la expresión (3.37) se puede realizar por diversos métodos. Quizás el más sencillo sea discretizar el camino en un número finito de puntos, establecer la ecuación en diferencias equivalente en cada punto, e ir iterando hasta que la ecuación se satisfaga dentro de los márgenes que correspondan a un error preestablecido.

Así pues, a una trayectoria inicialmente propuesta que une los dos puntos se le somete a una discretización uniforme, dividiéndola en P divisiones iguales de tamaño ϵ. A cada uno de los P+1 puntos creados por esta división se los denomina $\boldsymbol{x}_p$, $0 \leq p \leq P$. El punto $\boldsymbol{x}_0$ es el primero, es siempre el mismo y coincide con el punto inicial de la trayectoria. De forma similar, $\boldsymbol{x}_P$ es el último punto y también es fijo. $x_{p,k}$ es la coordenada k del punto $\boldsymbol{x}_p$.

Aproximando la derivada de cada coordenada k respecto al parámetro de la curva mediante el incremento de la coordenada k dividido entre la distancia real entre ambos puntos (diferencia central):

$$\dot{x}_{p,k} \cong \frac{x_{p+1,k} - x_{p-1,k}}{2\epsilon} = \frac{\Delta x_{p,k}}{2\epsilon} \tag{6.28}$$

[3] Para los que se supone que el valor de la función de decisión y su gradiente es prácticamente igual en sus dos extremos.

De la misma forma, la derivada segunda se puede aproximar por:

$$\ddot{x}_{p,k} \cong \frac{\left(x_{p+1,k} + x_{p-1,k} - 2x_{p,k}\right)}{\epsilon^2} \tag{6.29}$$

Sustituyendo en la ecuación diferencial de la geodésica (3.37):

$$\begin{aligned}
&\frac{\left(x_{p+1,k} + x_{p-1,k} - 2x_{p,k}\right)}{\epsilon^2} + \sum_{i,j} \Gamma_{ij}^{k}(\boldsymbol{x}_p)\, \frac{x_{p+1,i} - x_{p-1,i}}{2\epsilon} \frac{x_{p+1,j} - x_{p-1,j}}{2\epsilon} = 0 \\
&\frac{\left(x_{p+1,k} + x_{p-1,k} - 2x_{p,k}\right)}{\epsilon^2} + \\
&\qquad + \frac{1}{4\epsilon^2} \sum_{i,j} \Gamma_{ij}^{k}(\boldsymbol{x}_p)\, \left(x_{p+1,i} - x_{p-1,i}\right)\left(x_{p+1,j} - x_{p-1,j}\right) = 0 \\
&x_{p,k} = \frac{\left(x_{p+1,k} + x_{p-1,k}\right)}{2} + \frac{1}{8} \sum_{i,j} \Gamma_{ij}^{k}(\boldsymbol{x}_p)\, \left(x_{p+1,i} - x_{p-1,i}\right)\left(x_{p+1,j} - x_{p-1,j}\right)
\end{aligned} \tag{6.30}$$

De esta forma es posible calcular una "versión mejorada" de lo que debería valer $x_{p,k}$ para cumplir con la ecuación de la geodésica. Este valor de $x_{p,k}$ recién calculado será el valor para ese punto, y coordenada, que se usará en la próxima iteración.

Así pues, se irá iterando para cada una de las coordenadas y para todos los puntos (excepto el inicial y final que son fijos) en los que se ha discretizado la curva que une los puntos. Cuando se llegue a que la máxima diferencia en valor absoluto entre los diferentes puntos en sucesivas iteraciones sea menor que un valor pequeño fijado de antemano, se habrá obtenido un punto fijo de la ecuación diferencial y se podrá considerar que es la curva geodésica solución.

En la siguiente página se muestra el organigrama a seguir cuando se desea implementar este algoritmo.

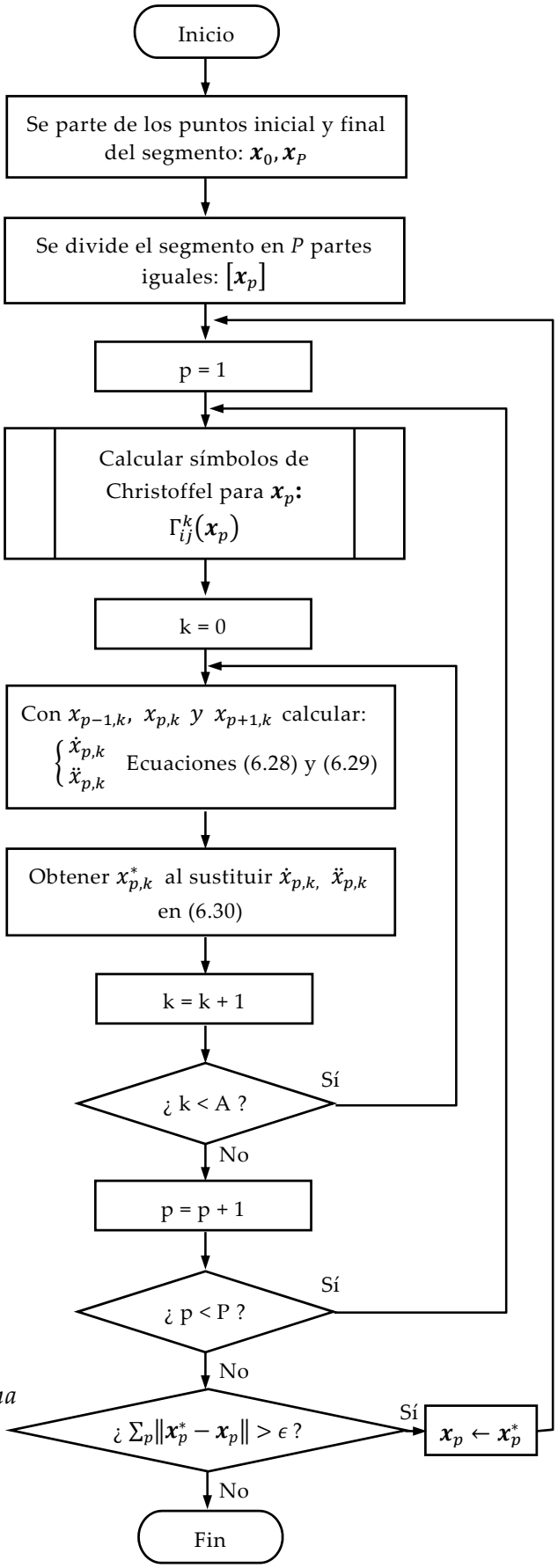


Figura 6.7.- Organigrama para el cálculo de los puntos de una geodésica integrando la ecuación diferencial.

6.2.2 Camino más corto entre dos puntos

Integrar de forma aproximada la ecuación diferencial de la geodésica no garantiza haber encontrado el camino más corto entre esos dos puntos de acuerdo a la métrica propuesta.

Una solución que sí lo garantiza es aplicar la búsqueda del camino más corto mediante un algoritmo bien conocido como puede ser el de Dijkstra [143]:

1. Dicho algoritmo establece primeramente una rejilla que cubre todo el espacio de atributos (en cada intersección de la rejilla se crea un vértice). También se inicializa una cola prioritaria que guarda la distancia calculada para llegar a cada uno de los vértices (en este momento la distancia para llegar a todos los vértices es ∞, excepto al vértice inicial que vale 0).
2. A continuación se procede a ir calculando la distancia a los vértices (V) conectados con el punto inicial mediante distintas aristas (E). Para cada vértice se calcula la distancia total recorrida desde el origen, y cuál es el último vértice por el que se ha llegado a este. En la cola de prioridad van quedando ordenados los vértices, empezando por los de menor distancia al punto origen.
3. A partir de ahora se procede a realizar un procedimiento iterativo, eligiendo el vértice que posee la menor distancia al punto inicial (vértice actual), y se lo extrae de la cola.
    - Se calcula la distancia desde él a todos los vértices que lo conectan mediante una arista (vértices destino).
    - La distancia desde el origen a los vértices destino se obtiene sumando la distancia calculada en el paso anterior a la distancia al origen que posee el vértice actual.
    - Se actualiza la distancia total a uno de esos vértices destino, siempre y cuando la recién calculada sea menor que la que ya poseía hasta este momento. En este caso también se actualizará el camino por el que se llega a este vértice (para indicar que es el vértice actual) y su posición en la cola de prioridad.

    Se repite este tercer paso hasta que en la cola de prioridad no quedan vértices para extraer. En este momento se conoce ya la distancia mínima desde el vértice inicial a cualquier otro en la rejilla.
4. Para conocer cuál es el camino más corto desde un vértice cualquiera al inicial, se comienza por el vértice final deseado y, empleando la información de por cuál nodo se ha llegado a él, se realiza iterativamente un "viaje hacia atrás" de vértice en vértice hasta llegar al punto inicial.

En las dos páginas siguientes se muestran los organigramas a seguir cuando se desea implementar este algoritmo.

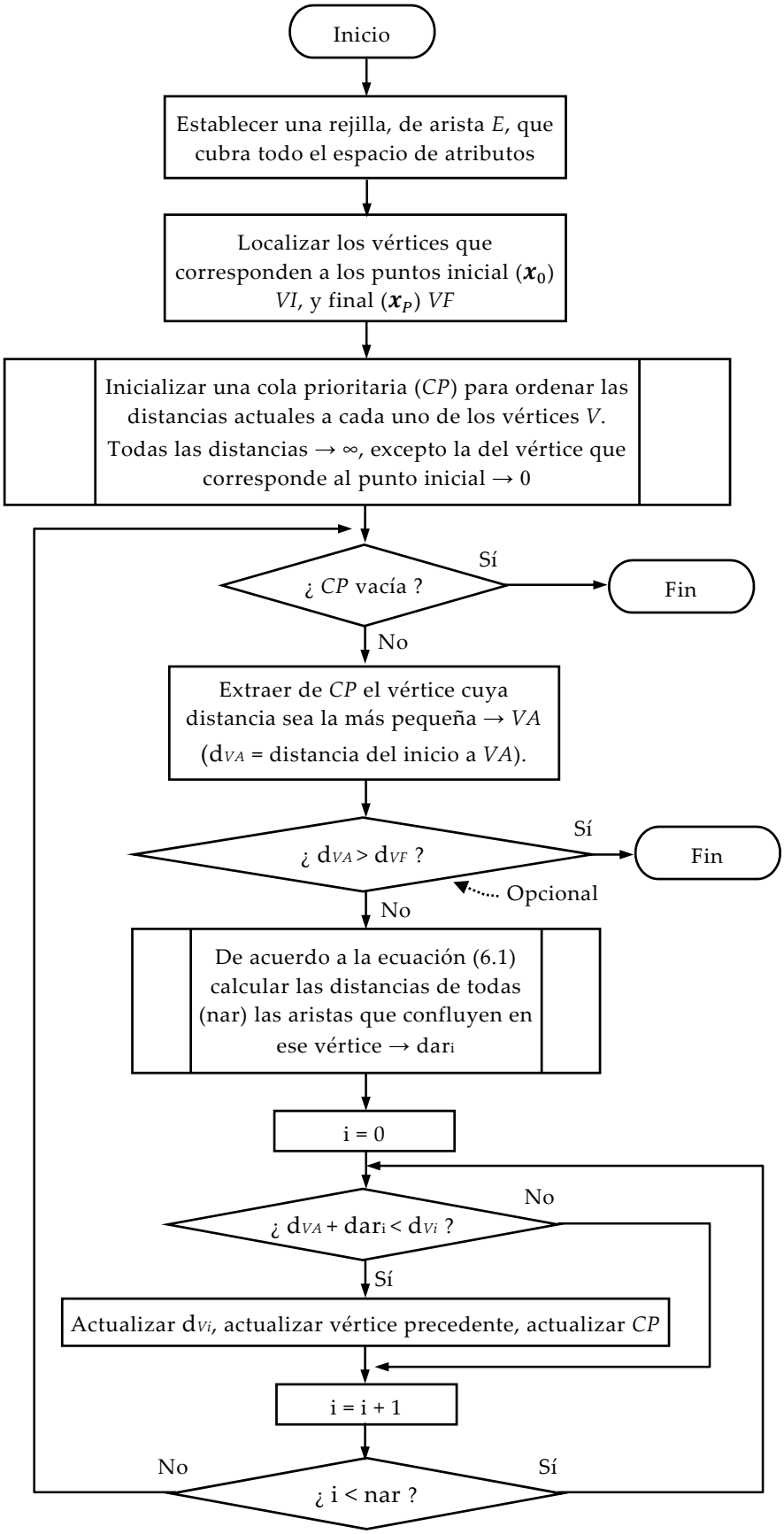


Figura 6.8.- Organigrama para el cálculo de la distancia entre dos puntos empleando el algoritmo de Dijkstra (establecimiento de las distancias en la rejilla).

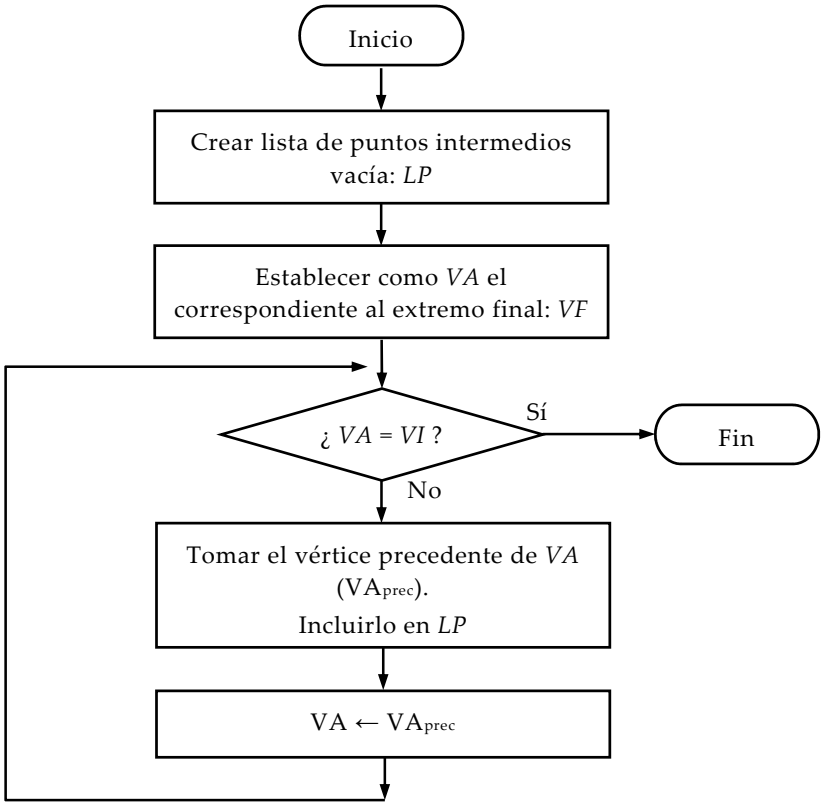


Figura 6.9.- Organigrama para el cálculo de la distancia entre dos puntos empleando el algoritmo de Dijkstra (lista de puntos intermedios de la geodésica).

6.3 Resultados experimentales

En esta sección se va a presentar en primer lugar el problema sintético que se usará en estas pruebas para, a continuación, proporcionar gráficos de los resultados obtenidos al integrar la geodésica que une dos puntos y, para terminar, se mostrarán en forma de tablas y gráficos los resultados obtenidos para la precisión de clasificación.

6.3.1 Problema sintético para probar la métrica LOM

Se ha empleado un problema de prueba sintético para verificar el correcto funcionamiento del algoritmo reseñado en esta sección.

Se buscó un problema de dos atributos que pudiera ser fácilmente reproducible, cuyo número de casos se podría variar a voluntad de forma repetible y con suficiente complejidad para ser interesante. Además debería cumplir con otra de las características habituales en el mundo real: no ser linealmente separable.

Así pues, en una primera fase se determinó generar seiscientos casos para cada una de las dos clases:

- La primera clase (+1) estaba formado por tres bloques, de doscientos casos cada uno, obtenidos aleatoriamente cada uno de ellos de distribuciones gaussianas con medias:

$$\begin{bmatrix}0\\2\end{bmatrix} \quad \begin{bmatrix}2\\0\end{bmatrix} \quad \begin{bmatrix}4\\4\end{bmatrix} \tag{6.31}$$

  y matrices de covarianzas:

$$\begin{bmatrix}0{,}1 & 0\\ 0 & 1\end{bmatrix} \quad \begin{bmatrix}1 & 0\\ 0 & 0{,}1\end{bmatrix} \quad \begin{bmatrix}1 & -1/\sqrt{2}\\ -1/\sqrt{2} & 1\end{bmatrix} \tag{6.32}$$

- La segunda clase (-1) está formada por un único bloque de seiscientos datos procedentes de una distribución gaussiana con media y covarianzas:

$$\begin{bmatrix}2\\2\end{bmatrix} \quad \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} \tag{6.33}$$

El resultado se puede apreciar en la Figura 6.10.

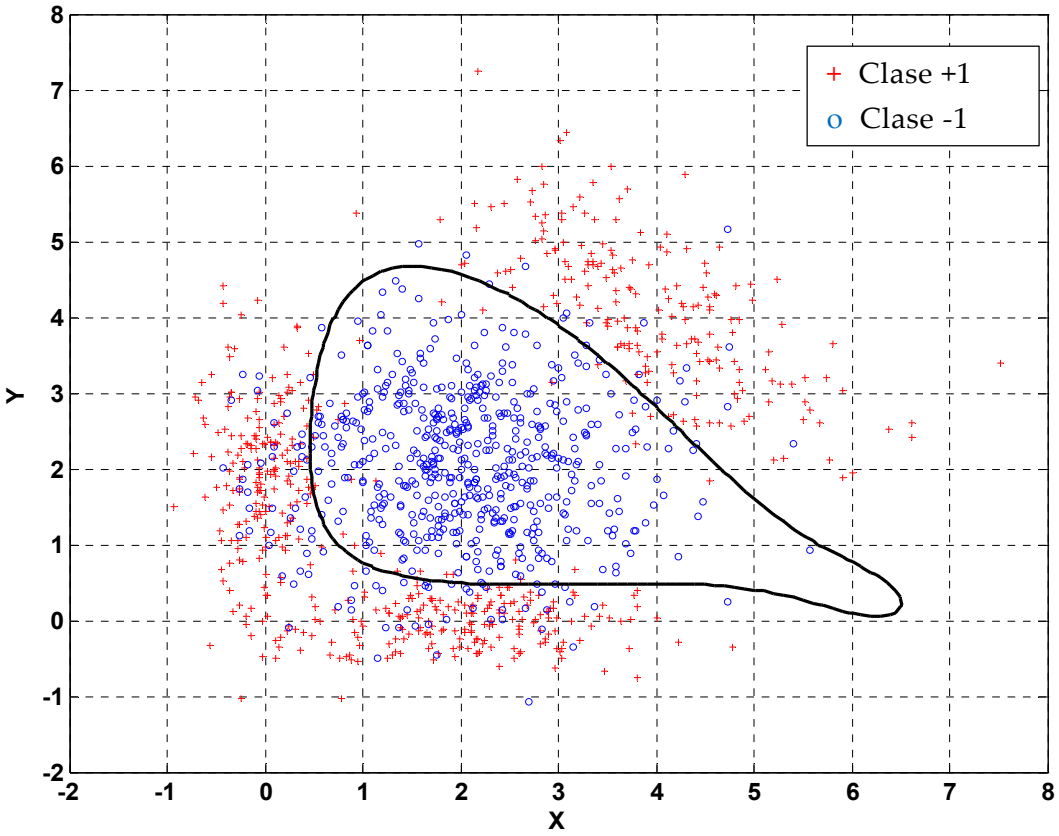


Figura 6.10.- Representación gráfica del problema sintético. Se aprecia una curva cerrada que es la función de decisión obtenida mediante una RBF-SVM con parámetros C=4 y γ=2. [4]

Al ser un problema sintético se puede calcular el error esperable al clasificar un nuevo caso, partiendo del conocimiento de las funciones de distribución que generan las distintas clases. En este problema el error de Bayes sería del 10,22%.

En la Figura 6.11 se muestran las fronteras de separación entre clases si se tuviera la información suficiente para aplicar el teorema de Bayes (en este caso se puede porque se conocen las pdf de las distintas clases que forman este problema).

[4] Donde $\gamma = 1/\sigma^2$.

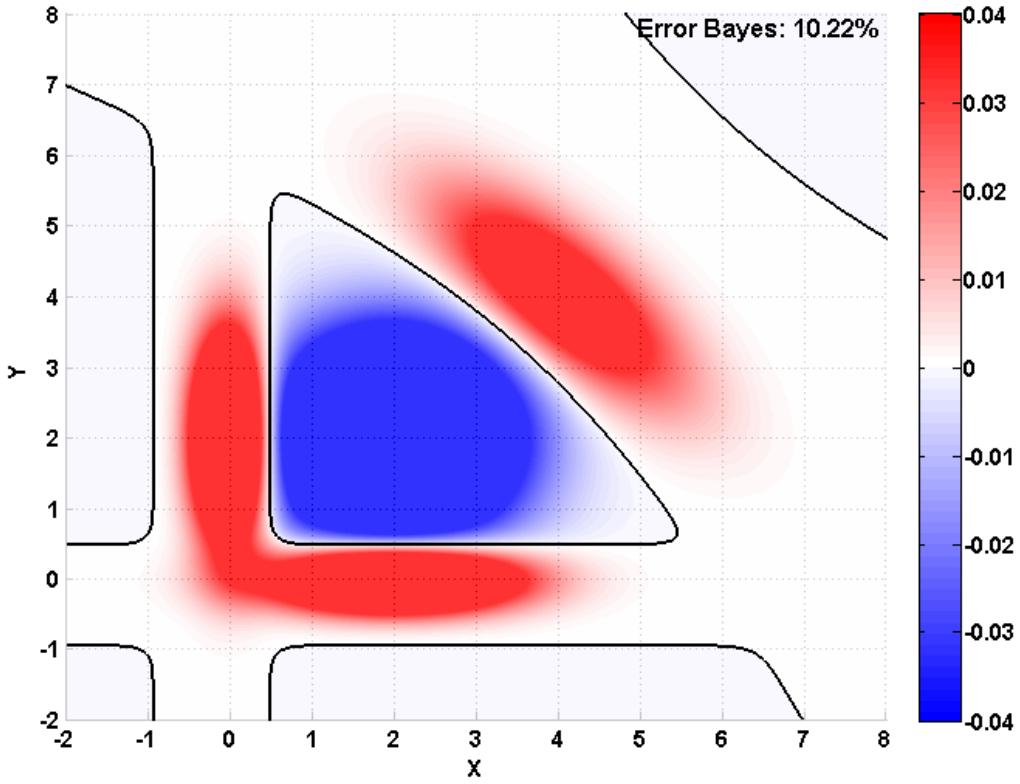


Figura 6.11.- Zonas de separación de las clases de acuerdo a Bayes.

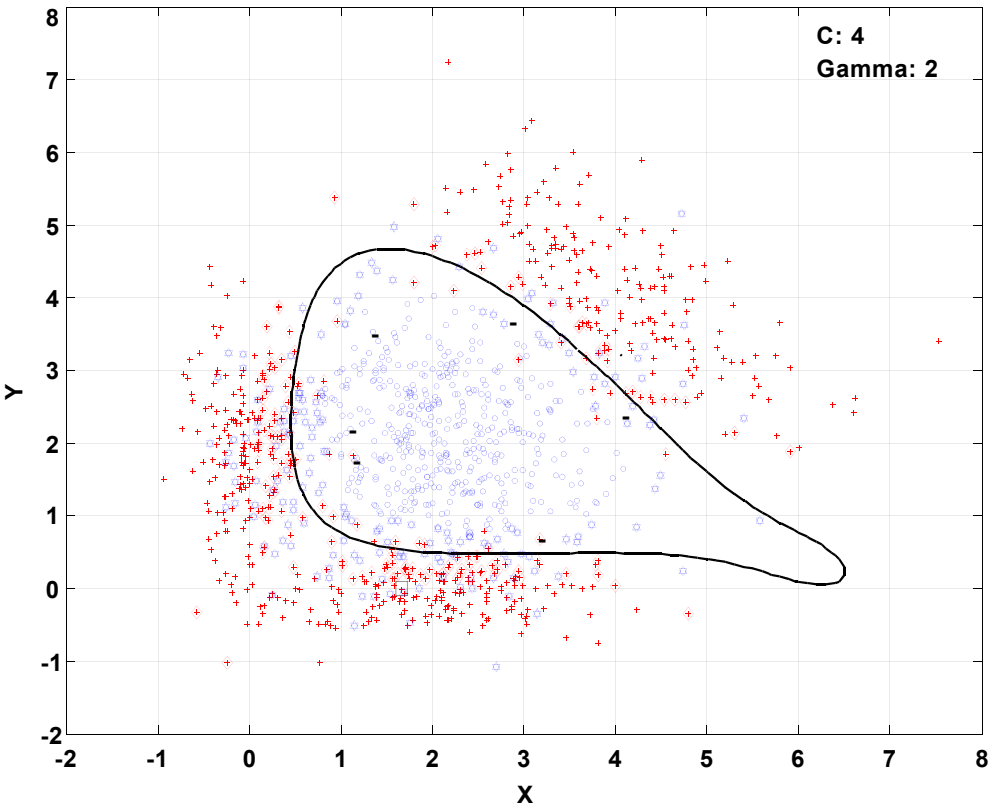


Figura 6.12.- Vectores de soporte para cada clase. Las estrellas corresponden a la clase -1 y los diamantes a la +1.

6.3.2 Función de decisión calculada mediante una RBF-SVM

Mediante la técnica de validación cruzada (dividiendo aleatoriamente los casos de partida en 10 bloques) se ajustan los parámetros de una SVM basada en un kernel RBF. Los parámetros óptimos para esta SVM son: C=4, γ=2, presentando un error medio en la clasificación de 10,75% (empleando 328 vectores de soporte). Así pues, se queda a solo 0,53% de lo que se puede obtener con la clasificación óptima. En la Figura 6.12 se puede apreciar la superficie de separación entre clases y los vectores de soporte necesarios.

6.3.3 Distancia calculada mediante integración de la ecuación diferencial

Empleando la curva de separación entre clases proporcionada por esta SVM se puede definir la métrica y calcular las distancias desde un punto cualquiera (en el caso de las siguientes figuras es el punto (0, 4)) a otro conjunto puntos del plano de acuerdo al método de la integración de la ecuación diferencial de la geodésica. Para estas integraciones se ha utilizado el parámetro $\tau = 1$.

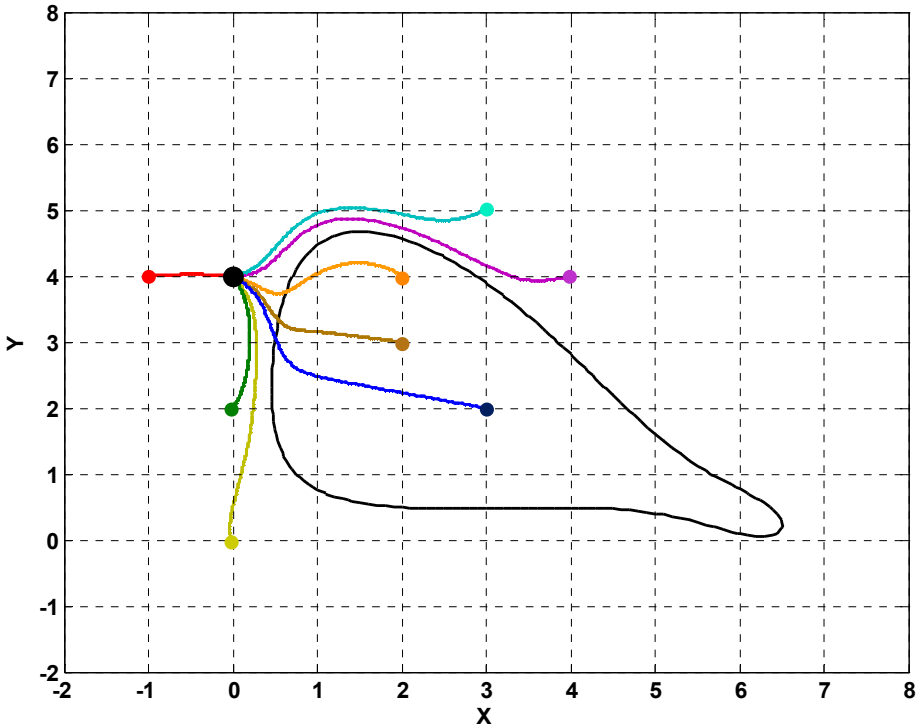


Figura 6.13.- Curvas geodésicas para alcanzar distintos puntos desde el (0, 4) de acuerdo a la métrica LOM.

Se puede apreciar que las líneas rectas tradicionales que unen dos puntos en la métrica euclídea han sido reemplazadas por curvas que siguen perfiles similares a los de la curva de decisión para alcanzar el punto destino.

Esto significa que los caminos más cortos prefieren rodear la superficie de separación entre clases en vez de intentar atravesarla.

Así pues, bajo esta métrica, los puntos que se localizan en trayectorias paralelas a la superficie de separación presentan mayor similitud, y una menor distancia, al caso a ser clasificado.

Los tres principales inconvenientes de esta técnica radican en que una trayectoria geodésica no garantiza que sea el camino más corto, los recursos informáticos necesarios para el cálculo de esta distancia son elevados, y la trayectoria geodésica puede variar dependiendo del camino inicial considerado en la integración de la ecuación diferencial.

6.3.4 Distancia calculada mediante el algoritmo de Dijkstra

Una alternativa que sí garantiza el cálculo de la distancia más corta entre puntos pasa por emplear el algoritmo de Dijkstra.

En esta investigación se ha implementado una variación de dicho algoritmo que emplea una cola de prioridad para acelerar los cálculos y que también permite recorrer caminos en diagonal entre los elementos de la rejilla.

En este caso es posible calcular la distancia entre un punto y el resto de puntos del espacio de atributos; además se puede dibujar las curvas de nivel de los puntos que distan lo mismo de uno dado.

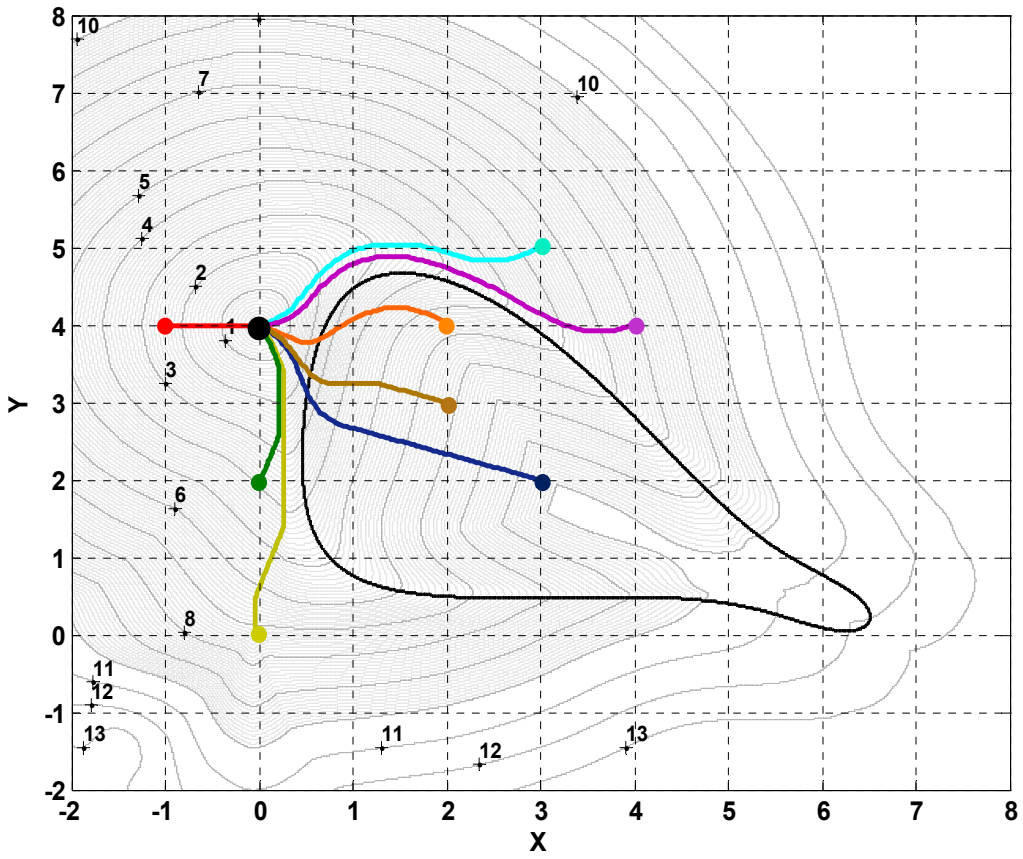


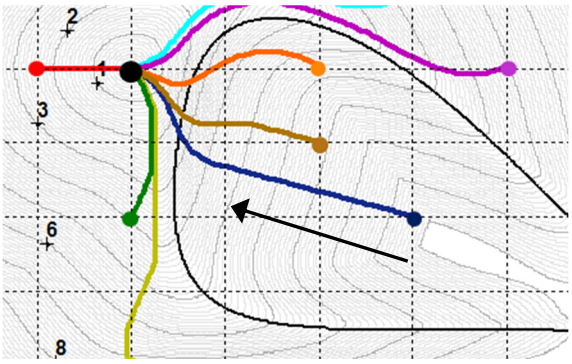
Figura 6.14.- Camino más corto para alcanzar distintos puntos desde el (0,4) empleando la métrica LOM. Se puede ver también las curvas de nivel que unen puntos equidistantes del (0,4).

Los resultados obtenidos se pueden apreciar en la Figura 6.14. Inmediatamente se aprecia que los resultados concuerdan con los obtenidos al integrar la ecuación de la geodésica.

Analizando las curvas de puntos equidistantes se aprecia cómo, en torno al punto inicial de la geodésica (0,4), las curvas de nivel están distorsionadas, manteniendo, más o menos, la forma de una circunferencia en la parte más alejada de la superficie de separación de clases; pero más apiñadas, y adoptando trayectorias paralelas a esta, en la zona más cercana. Esto indica que se ha conseguido el efecto deseado. En las proximidades de la superficie de separación "viajar" en dirección perpendicular a esta es más costoso que en dirección paralela.

Alejándose de dicha superficie de separación, la métrica se convierte de nuevo en euclídea, que era como se propuso.

En las proximidades de la superficie de separación, se puede viajar mucho más espacio sin que la distancia aumente mucho.



Y, por último, se aprecia en la parte interior de la superficie de separación un fenómeno curioso, es como si las curvas de nivel estuvieran "invaginadas" hacia el punto desde donde ha comenzado la geodésica. La explicación es sencilla, para llegar a estos puntos se ha tenido que atravesar perpendicularmente la frontera y ese "costo" (medido como distancia) se mantiene al seguir moviéndose en dicho interior[5]. En cambio a los puntos que están más hacia los laterales se ha llegado a través de curvas geodésicas que viajan mucho tiempo paralelas a la frontera de separación (y por tanto sus distancias son menores).

En la Figura 6.15 se pueden apreciar diversos lugares geométricos de puntos que equidistan de uno central para una métrica LOM con $\tau = 3$. Para esta prueba se ha partido del conjunto de datos de aprendizaje que tiene 180 casos. Con trazos discontinuos se han marcado las fronteras de separación de clases óptimas (de acuerdo a Bayes). Se ha entrenado una SVM con los datos de aprendizaje y los parámetros C=16, γ=0,2; el elipsoide de trazo continuo grueso del centro de la figura representa la función de decisión (la cual se asemeja solo aproximadamente a la verdadera frontera). También se han dibujado mediante unas curvas de trazo continuo fino las curvas que unen los puntos que distan lo mismo de los puntos: (0,17, 2,2), (2, 2), (3, 0), (3, 4), (3, 7) y (7, 0,5).

Se puede apreciar que dichas curvas de nivel se "distorsionan" de acuerdo a lo que se planteó como objetivo de la métrica LOM. Cuanto más se aproximan

[5] Para así cumplir con la desigualdad triangular.

a la frontera de separación entre clases, y más se discrimina entre direcciones, mayor acortamiento se da en la dirección perpendicular a la frontera y más se alarga en dirección paralela a ella. Lejos de la superficie de separación, por ejemplo en el punto (3, 7), la métrica se vuelve euclídea.

Así pues, se comprueba gráficamente que la elección de los vecinos próximos queda mejorada de forma sustancial y que el clasificador que use esta métrica será mucho más robusto.

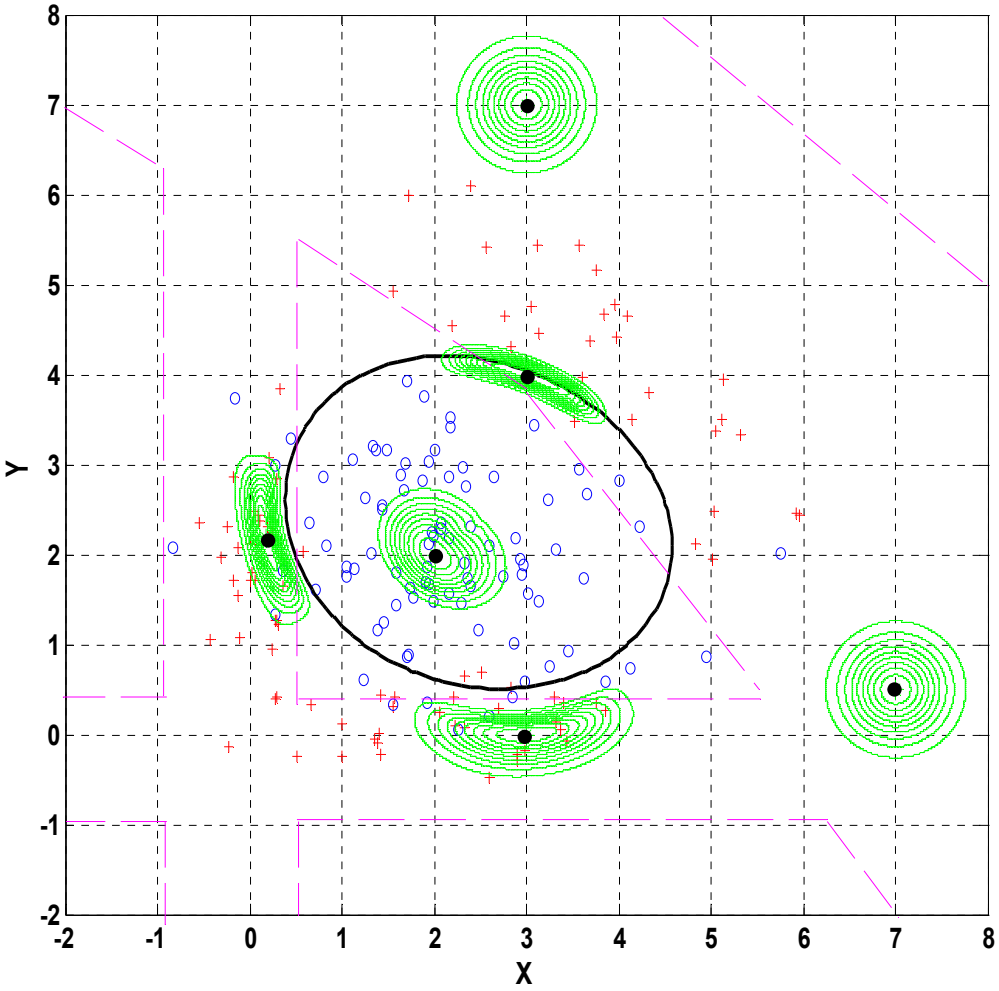


Figura 6.15.- Curvas de equidistancia para los puntos (0,17, 2,2), (2, 2), (3, 0), (3, 4), (3, 7) y (7, 0,5).

Para terminar el análisis gráfico de los resultados, se puede señalar que se ha conseguido el objetivo de que las curvas de nivel de la métrica no sean simples circunferencias, ni elipsoides fijas en un punto, como pasa en los algoritmos más avanzados en la actualidad, sino curvas mucho más adaptadas a cada problema. Se puede observar que recorridos a puntos relativamente cercanos de acuerdo a la métrica euclídea, se convierten en distancias mayores o menores de acuerdo a en qué dirección se viaja desde el punto inicial. La métrica LOM no ha sido solo un artefacto matemático más o menos bien construido. Experimentalmente, LOM responde a las premisas con las que fue diseñada.

6.3.5 Resultados experimentales en el contexto del algoritmo *k*-NN

Para comprobar la efectividad de esta métrica se han preparado cuatro conjuntos de 180, 90, 60 y 30 casos respectivamente del problema sintético descrito al principio de esta sección. También se han generado 6.000 casos como conjunto independiente de casos de test.

Se pretende elaborar un conjunto de pruebas que permitan mostrar que no solamente la precisión de clasificación mejora cuando se usa la métrica LOM, sino que además también se cumple cuando el número de casos de aprendizaje es grande o pequeño.

En primer lugar, con los datos de aprendizaje se entrenan cuatro RBF-SVM. Posteriormente se evalúa la calidad de la clasificación cuando se aplica el algoritmo *k*-NN usando tanto la métrica euclídea como la métrica LOM. Para ello se emplean como casos de aprendizaje los cuatro conjuntos de datos y como evaluador del grado de aciertos los 6.000 casos de test.

En este apartado, para calcular la distancia entre dos puntos mediante la métrica LOM se emplea el método de Dijkstra. También se ha estudiado el efecto de la longitud de la arista (E) en el deterioro de la capacidad de clasificación.

En la Tabla 6.1, y en los cuatro gráficos de la Figura 6.16 se muestran los resultados obtenidos.

Problema	***k*-NN métrica euclídea**	**SVM**	***k*-NN métrica LOM**
180 casos	88,25% (k=13)	88,54%	88,65% (τ=0,2, k=13)
90 casos	87,23% (k=3)	87,53%	89,12% (τ=1,25, k=7)
60 casos	84,40% (k=5)	85,08%	88,98% (τ=1,25, k=13)
30 casos	81,23% (k=1)	82,40%	83,33% (τ=3, k=2)

Tabla 6.1.- Resumen de las distintas precisiones en la clasificación.

En cada uno de los cuatro escenarios (180, 90, 60 y 30 casos de aprendizaje), se aprecia un incremento significativo en la precisión de la clasificación cuando se emplea la métrica LOM, tanto frente a la precisión obtenida con la métrica euclídea, como a la conseguida con la SVM.

El rango de valores de τ para los que la métrica LOM presenta una mejora es relativamente amplio y la longitud de la arista (para el algoritmo de Dijkstra) con la que se puede obtener un cálculo correcto es relativamente grande (del orden de 1/50 del rango total de los atributos).

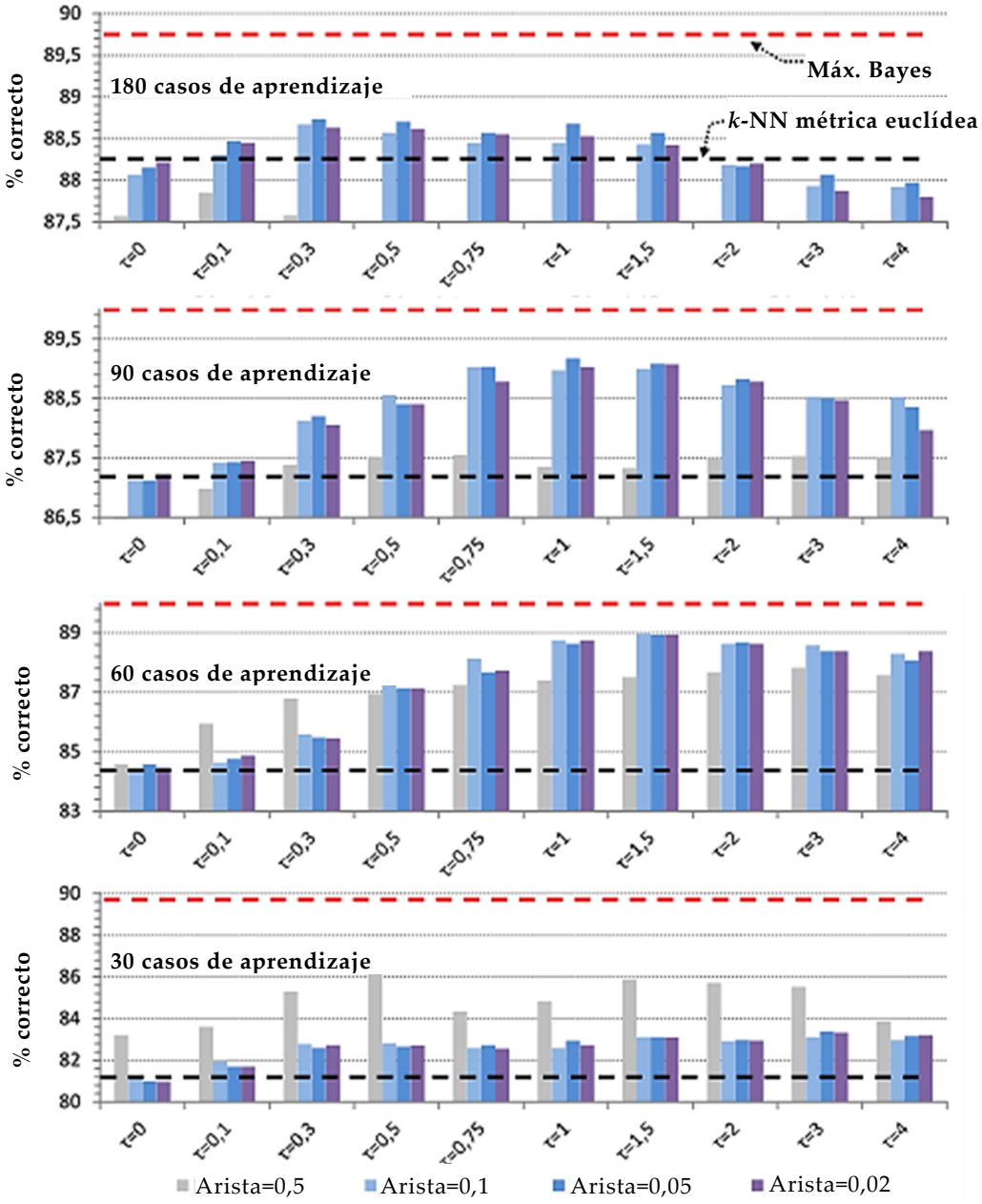


Figura 6.16.- Resultados de la clasificación k-NN para los cuatro problemas de test.

Analizando los resultados que se obtienen con el algoritmo *k*-NN, algunos aspectos se muestran relevantes:

- No es necesario explorar distintos valores para el parámetro *r* (para el algoritmo *k*-NN, *r* es meramente un factor de escala).
- El rango de valores para el parámetro τ que proporciona mejoras significativas es relativamente amplio.
- Incluso para aristas del orden de 0,1 unidades, el cálculo de distancias realizado mediante el algoritmo de Dijkstra es lo suficientemente correcto (hay que tener en cuenta que en el problema sintético, el rango de cada atributo es aproximadamente (-1, 6)).

Aristas más cortas no proporcionan una mejora significativa en la precisión del cálculo de la distancia (solo incrementa el tiempo de cálculo).

- Uno de los inconvenientes del algoritmo de Dijkstra es el tiempo necesario para su cálculo: $O\big((|E| + |V|)log|V|\big)$ empleando una cola de prioridad, siendo $|E|$ el número de aristas y $|V|$ el número de vértices.

  Las soluciones alternativas pasan bien:

  - Por emplear un algoritmo que permita calcular la distancia entre dos puntos cualesquiera, como puede ser el algoritmo de Floyd-Warshall [143]; pero estos algoritmos consumen mucha memoria y se escalan mal a un número de dimensiones mayor que dos.
  - O bien por usar simplificaciones, tal como en la que se está trabajando en la actualidad, que se basan en la integración de la ecuación de la geodésica con un número reducido de puntos intermedios. Esta solución disminuirá la precisión en el cálculo de la distancia entre dos puntos, pero presentará mucha mejor escalabilidad en cuanto al número de atributos.

6.4 El factor de magnificación para la métrica LOM

Como se ha comentado en la sección 4.5, el factor de magnificación se define como $\sqrt{|\boldsymbol{G}|}$.

Para un problema bidimensional el tensor de la métrica LOM sería:

$$\boldsymbol{G} = \begin{bmatrix} \frac{gr_x^2}{r_m^2} + \frac{1 - gr_x^2}{r_M^2} & gr_x\, gr_y \left(\frac{1}{r_m^2} - \frac{1}{r_M^2}\right) \\ gr_x\, gr_y \left(\frac{1}{r_m^2} - \frac{1}{r_M^2}\right) & \frac{gr_y^2}{r_m^2} + \frac{1 - gr_y^2}{r_M^2} \end{bmatrix} \tag{6.34}$$

Se demuestra fácilmente que el determinante de esta matriz es:

$$|\boldsymbol{G}| = \frac{1}{r_m^2 \,.\, r_M^2} \tag{6.35}$$

Con lo que el factor de magnificación sería $1/r_m \,.\, r_M$.

Si se sustituyen los radios menor y mayor por las expresiones utilizadas en la métrica LOM:

$$\begin{aligned} r_m &= \frac{r}{\left(1 + \tau e^{-f^2(\boldsymbol{x})}\right)} \\ r_M &= r \,.\left(1 + \tau e^{-f^2(\boldsymbol{x})}\right) \end{aligned} \tag{6.36}$$

el factor de magnificación sería $1/r^2$.

La orientación de esta magnificación es máxima en la dirección perpendicular a la frontera de separación entre clases y menor en todo el plano tangente a ella.

Aparte de la ventaja demostrada que ofrece el orientar la magnificación en la dirección perpendicular a la frontera de separación, sobre el valor escalar del factor de magnificación se pueden realizar las siguientes observaciones.

- En aquellos algoritmos en los que el parámetro *r* no influye en la clasificación (como puede ser el *k*-NN, donde *r* no pasa de ser un factor de escala), no tiene mucho sentido elaborar discusiones sobre su valor.
- En aquellos otros, como las RBF, donde el parámetro *r* sí tiene relevancia, sería conveniente probar con valores de *r* menores que la unidad para aumentar la ponderación de los elementos más cercanos en las cercanías de la frontera de separación entre clases.

6.5 Conclusiones

El algoritmo LOM combina el uso de la clasificación basada en vecinos próximos con una métrica local diseñada para uniformizar la probabilidad de que un caso pertenezca a una determinada clase en un entorno local del punto a clasificar.

La métrica LOM discrimina la dirección perpendicular a la superficie de separación de clases, previamente calculada mediante los propios casos empíricos, del resto de direcciones contenidas en el plano tangente. A lo largo de la dirección perpendicular se "ensancha" la medida de distancias, mientras que el resto de direcciones ven "contraída" esta distancia.

Bajo estas premisas, se propone una formulación (ecuaciones (6.1) y (6.4)) que aseguran que la métrica LOM es de tipo riemanniana, siendo su tensor siempre definido positivo independientemente de los valores elegidos para los parámetros r y τ.

Para el cálculo de la distancia entre dos puntos en dicha métrica se propone la integración a lo largo de una geodésica, para lo cual se propone una doble alternativa, bien la integración de la ecuación diferencial que la define como geodésica, o bien emplear el método de cálculo de la distancia más corta por el método de Dijkstra.

Los resultados experimentales plasman con claridad que los objetivos de conseguir una métrica con las características marcadas en el objetivo de la sección 6.1.1 se han conseguido. Aunque quedan pendientes más pruebas de su empleabilidad en casos prácticos, la sólida fundamentación teórica de su diseño ayudará a que se cumplan las expectativas.

Al igual que en la métrica BTW, el problema de trabajar con múltiples clases se resolvería repitiendo la clasificación de un caso bajo cada combinación de parejas de clases y posteriormente quedándose con la clase más frecuentemente repetida ("una vs. una").

Uno de los elementos distintivos de LOM es que no se utiliza el enfoque de la optimización de la matriz de la métrica completa, sino que se busca en cada punto una dirección principal local para esta métrica y sobre ella, y su perpendicular, discrimina quiénes son los vecinos más próximos. De esta forma se evita tener que optimizar múltiples parámetros (como se hace en el algoritmo LMNN) y se combate más eficientemente la maldición de la dimensionalidad y el riesgo de sobreaprendizaje.

7 Conclusiones y líneas futuras de trabajo

En las dos últimas décadas se ha pasado de no considerar las métricas como un componente importante de los algoritmos, a ser centro de atención de la literatura científica. Quizás el punto de inflexión fue la introducción de los "*kernels*" como artificios que implícitamente conllevan una transformación geométrica del espacio.

El empleo de métricas distintas de la euclídea permite mejorar las características de los algoritmos de clasificación. Entre estas características se encuentra obviamente la precisión de la clasificación, pero también el tiempo necesario para realizarla, la escalabilidad tanto en número de casos como de atributos, etc.

Dentro de las métricas se pueden distinguir aquellas que se aplican en todo el espacio de los atributos (globales) y las que varían según el punto de aplicación (locales). Existe un paulatino incremento del interés por las métricas locales.

Como se ilustra en la Figura 4.2, y en relación con las métricas habitualmente usadas en las medidas de distancia y similitud entre casos (ver sección 4.1.4), los puntos que equidistan de uno dado adoptan una cierta simetría respecto al punto desde el que se toma la referencia. Desde mi punto de vista esta es una de sus limitaciones más importantes; no todos los atributos son igualmente importantes a la hora de clasificar un caso e, incluso, la combinación concreta de sus atributos puede determinar la relevancia de estos.

En este último capítulo se va a realizar un balance de lo conseguido por las métricas propuestas en esta tesis, pasando revista a la hipótesis de partida y a los objetivos que se plantearon, resumir las características de las métricas BTW y LOM y, por último, se enumerarán las diferentes líneas de trabajo futuro abiertas en esta investigación.

7.1 Validación de la hipótesis

La hipótesis inicial quedó establecida como:

"Sería posible mejorar las prestaciones de los sistemas de clasificación si en vez de emplear una métrica euclídea se utilizase otra medida de distancia que se acomodara a las características del problema en cuestión. Esta nueva medida de distancia podría ser global o local, y debería poder ajustarse a cada problema bien mediante un cierto conocimiento previo y/o mediante técnicas de optimización".

Los resultados, tanto teóricos como experimentales, confirman que las modificaciones en las métricas tradicionales mejoran sustancialmente distintos aspectos de los procesos de clasificación (precisión, velocidad de búsqueda…), adaptando así su uso a escenarios específicos.

BTW y LOM son los dos exponentes fruto de esta tesis que confirman la hipótesis; la primera persigue clasificar de forma correcta y reducir los recursos informáticos necesarios para su implementación (microprocesadores con unos pocos kB de memoria RAM/EEPROM), mientras que la segunda es intensiva en cálculo y procura mejorar la búsqueda de los vecinos próximos a un caso dado. Una es una métrica global y otra local, y en ambas se puede optimizar un único parámetro continuo.

7.2 Consecución de los objetivos

Se ha logrado la validación de la anterior hipótesis por medio de ir cumpliendo los objetivos que se propusieron en un principio y siguiendo la metodología indicada en la sección 1.3.

Revisando los objetivos, se puede concluir:

O1) *Estudio de los métodos de clasificación*

Se ha procedido a un estudio detallado de los principales métodos de clasificación supervisada presentes en el estado del arte, siguiendo la taxonomía reflejada en la Figura 2.1. De ellos surgieron los tres que iban a ser clave en esta tesis: *k*-NN, LDA y SVM. Estos tres fueron analizados con especial profundidad, estudiadas sus variantes y por último codificados en el lenguaje C++.

Esta programación, junto a la de las técnicas de validación cruzada, hace que los resultados experimentales puedan ser comparados.

O2) *Estudio de los fundamentos matemáticos*

Los aspectos matemáticos fueron los más complejos de abordar al principio de la tesis. Pronto se reveló que existían dos grandes conjuntos de conocimiento sin el cual sería imposible abordar el resto de objetivos: la optimización de funciones y las métricas de Riemann.

La optimización es omnipresente en la inteligencia artificial actual. Asociar la técnica correcta a los diferentes tipos de problemas a resolver y conocer los más pequeños detalles (para poder afrontar su programación) es de fundamental importancia. El objetivo se ha cumplido, no solo realizando un estudio teórico, sino implementando en lenguaje C++ las distintas técnicas necesarias para las métricas BTW y LOM.

El estudio de las métricas de Riemann es de naturaleza teórica. Su amplitud y dificultad hace que su dominio completo quede fuera del alcance de una tesis en ciencias de la computación. Ahora bien, todo el tiempo y esfuerzo dedicado a este campo de conocimiento ha reportado importantes beneficios prácticos en la definición de la métrica LOM.

O3) *Estudio del estado del arte*

Por una parte, se han estudiado con detenimiento tanto las métricas tradicionales como las aplicables a un conjunto restringido de problemas. Ha resultado un trabajo ingente (en estudios consultados, se recogen más de 500 métricas distintas), este apartado se cierra con una discusión más en profundidad de aquellas relacionadas con esta tesis y una descripción más somera de otra docena de las más utilizadas, y que permitirían convertir medidas de disimilitud entre información no numérica (textos, bits…) en valores numéricos que podrían así ser empleados en BTW y LOM.

Por otra parte, y bajo la distinción entre métricas globales y locales, se repasan y discuten los artículos de revistas y contribuciones a congresos que fueron los gérmenes directos de las dos métricas que se proponen en esta tesis.

O4) *Propuesta, estudio, justificación e implementación en forma de programa informático de una nueva métrica global*

Partiendo del conocimiento y experiencia adquirida en los tres primeros objetivos, se ha concebido y desarrollado la métrica global BTW. Se consigue una fortísima reducción de la cantidad de información a guardar y del tiempo necesario para recuperar los casos más próximos a uno dado; al mismo tiempo su precisión es competitiva frente a los algoritmos que le sirven de base. Este algoritmo abre nuevas perspectivas sobre los métodos que permiten implementar búsquedas rápidas de vecinos, no solo similares, sino de la misma clase.

O5) *Propuesta, estudio, justificación e implementación en forma de programa informático de una nueva métrica local*

De forma similar se ha concebido también la métrica local LOM. En este caso se consigue modificar la geometría del espacio de los atributos para lograr una mejor precisión de clasificación. La formulación adoptada garantiza que cumpla con los axiomas exigibles a una métrica en todos los puntos del espacio.
Es una métrica alineada con los desarrollos más avanzados del estado del arte en este campo.

7.3 Conclusiones

Tras pasar revista a los objetivos conseguidos, es posible establecer un conjunto de conclusiones generales sobre los resultados de esta tesis.
Los métodos de búsqueda por vecinos próximos no solo proporcionan una clasificación del caso presentado, sino que también enumeran aquellos casos de la base de datos que han servido para tomar esta decisión. Frente a otras técnicas que no facilitan esta información[1], aquí un experto puede juzgar no solo la bondad de la clasificación, sino reutilizar para el presente caso toda la información anexa disponible (modo de reparación de una máquina, tratamientos a administrar a un paciente, soluciones propuestas previamente…) que se aplicaron en los casos similares conocidos. Emplear este tipo de algoritmos facilita el desempeño de los sistemas expertos basados en "Case Based Reasoning" o "Data Mining".

En el estado del arte se aprecia una tendencia paulatina a proponer métricas locales frente a las globales, de cualquier forma ambas coexistirán en un futuro.
En determinados problemas es preferible adoptar una métrica global (LMNN, BTW…) donde los parámetros son iguales en todas las zonas del espacio de atributos. Dan lugar a algoritmos más robustos, ya que pueden utilizar todos los casos para fijar la métrica, y consumen menos recursos en la búsqueda final de los vecinos próximos de uno dado.
Por otra parte es posible querer disponer de una métrica local (LaMaNNa, LOM…) que, para un caso dado, discrimine mejor quiénes son los vecinos próximos de su misma clase; aunque se requiera más tiempo de cálculo. Las métricas locales más avanzadas se apoyan en el conocimiento "a priori" de la función de decisión que separa las clases, aunque habitualmente esta se obtiene de los propios casos empíricos.

[1] Como pueden ser regresiones lineales, LDA, SVM, redes neuronales…

Si nos atenemos a la métrica BTW propuesta en esta tesis, varias son las conclusiones que es posible establecer:

- Proporciona una búsqueda muy rápida de los vecinos próximos de uno dado. Es por tanto aplicable en entornos donde se necesite una respuesta en tiempo real. La previa ordenación de los casos, de acuerdo al único atributo resultante, requiere de un cómputo inicial intensivo; pero acelera de forma importante la posterior búsqueda de los casos próximos a uno dado.
- A diferencia de otros métodos, no busca los vecinos de acuerdo a una métrica euclídea, sino que se orienta a la búsqueda preferentemente de los vecinos de su misma clase. Es pues una métrica orientada a la clasificación.
- No es necesario determinar el número de vecinos óptimos k (en algunas ocasiones muy próximos y en otras muy alejados del caso actual). La estrategia es considerar todos aquellos elementos de la base de casos que no se "separan" mucho del actual, cada uno de ellos ponderado en función de su proximidad al actual.
- Como se puede apreciar en la sección 5.4, y por término medio, la precisión de las predicciones del algoritmo BTW es competitiva tanto con las del algoritmo k-NN como con las del LDA.
- Frente a las posibilidades teóricas del algoritmo LDA, el uso de la métrica BTW con un clasificador k-NN añade un comportamiento no lineal en las clasificaciones, ayudando a mejorar la precisión de los resultados en problemas cuyos atributos siguen distribuciones multimodales.
- La métrica BTW se puede interpretar como la fusión de la información procedente de varios atributos en uno solo (lo cual puede aplicarse, por ejemplo, en entornos que necesiten una fusión sensorial), esto se potencia además porque el algoritmo BTW incrementa la velocidad de búsqueda y reduce los requerimientos de almacenamiento de información.

De forma similar, también es posible establecer un conjunto de conclusiones relacionadas con la métrica LOM:

- Está en línea con las métricas locales más modernas: empleo de una función de decisión previa, uso del gradiente como estimador de la dirección de separación entre clases, análisis desde el punto de vista de la geometría de Riemann…
- Se ha conseguido el gran objetivo geométrico planteado para la métrica LOM: no solo se ensancha el espacio en la dirección paralela a la función de decisión, también se consigue que las curvas de nivel no sean elipses (centradas en el punto origen) que se extienden por todo el espacio de atributos, sino curvas que se adaptan localmente al perfil de la función de decisión.

- Para casos próximos al buscado, esta métrica discrimina correctamente las direcciones preferentes de búsqueda de vecinos de su misma clase.
- El riesgo de sobreaprendizaje se minimiza al introducir solo dos parámetros (uno en el caso *k*-NN) porque la métrica no incrementa de forma importante la flexibilidad del modelo del clasificador.
- El cálculo de distancias usando la métrica LOM se puede considerar como una alternativa teórica para los algoritmos que usan distancias como medidas de disimilitud entre casos (y no solo para *k*-NN). Incrementa la precisión de la clasificación y reduce la maldición de la dimensionalidad. Su mayor inconveniente es la necesidad de un cálculo intensivo para obtener dicha distancia.

Como colofón es posible incluir una reflexión común a ambas métricas, se puede sintetizar como: "BTW necesita de LOM para mejorar su precisión, y LOM necesita de BTW para mejorar su tiempo de cálculo".

Como se ha expuesto en esta tesis, la métrica local mejora la precisión de la clasificación a costa de un procedimiento mucho más complejo para calcular la distancia de un caso a sus vecinos. La forma más simple de minimizar este problema es conocer de antemano cuáles son los candidatos a ser los vecinos más próximos (de forma aproximada e incluyendo también aquellos casos que sean muy dudosos de serlo). Posteriormente se obtendrán las distancias (medidas a lo largo de geodésicas) del caso considerado a los integrantes de este conjunto mucho más reducido de candidatos. De esta forma no solo se logra una mejor escalabilidad en los problemas con gran abundancia de casos, también el cálculo de la distancia geodésica entre puntos más cercanos es más preciso y rápido.

7.4 Líneas futuras de trabajo

Ambas métricas presentan varios aspectos que podrán ser objeto de futuras investigaciones.

Para la métrica BTW:

- En las últimas etapas de esta tesis se ha encontrado que la precisión del algoritmo BTW depende de la "calidad" de la dirección principal elegida para la transformación de coordenadas. Se puede investigar el empleo de otros algoritmos, como podría ser una SVM lineal, para establecer dicha dirección.
- Otro campo es la mejora de las técnicas de optimización para reducir el tiempo empleado en buscar la ponderación ideal, y así acelerar el tiempo del aprendizaje "offline".

- Desarrollo de una técnica que evite el tener que trabajar por parejas de clases. De esta forma se facilitaría que el algoritmo pudiera abordar fácilmente problemas con numerosas clases (por ejemplo, para 10 clases ahora hay que realizar 45 comparaciones entre parejas de clases, lo cual resulta tedioso). La estrategia "una vs. resto" puede ser una primera candidata, pero plantear un problema con más dimensiones espaciales también podría ser explorada en un futuro.
- Estudiar el comportamiento del algoritmo BTW cuando el número de atributos del problema sea muy elevado. Reducir un número elevado de valores por caso a uno solo no parece una estrategia muy prometedora desde el punto de vista de la teoría de información; será necesario investigar si una segunda y/o tercera coordenadas pueden aportar información que ayude a mejorar la clasificación. Las investigaciones de Zhang et al. [116] en torno a la información de los subespacios nulos en el análisis discriminante puede ser un buen punto de partida.

Para la métrica LOM:

- Como se indica frecuentemente en la literatura científica, el camino más corto entre dos puntos en una métrica de Riemann es una geodésica, pero no toda geodésica es el camino más corto. Durante esta investigación, en varias ocasiones se han encontrado dos geodésicas diferentes que conducen de un punto a otro y cuya integración no proporciona la misma distancia. Esto se debe a la disposición de distintos caminos iniciales y al comportamiento de los atractores que gobiernan la integración de la ecuación diferencial. Desde este punto de vista es preferible emplear el algoritmo de Dijkstra para calcular la distancia. Sin embargo, uno de los inconvenientes de dicho algoritmo es el tiempo de cálculo que se requiere. Su complejidad es de orden $O\big((|E| + |V|)log|V|\big)$ cuando se usa una cola de prioridad (donde $|E|$ es la cardinalidad de las aristas y $|V|$ la cadinalidad de los vértices). Uno de los aspectos que requieren mayor investigación es el estudio de la degradación, tanto en precisión como en tiempo de cálculo, del algoritmo de Dijkstra cuando el número de dimensiones es mayor que dos.
- Reducir los tiempos de determinación de la distancia geodésica mediante la integración de la ecuación diferencial es también una línea de futura investigación. Reducir el número de puntos en la discretización de la trayectoria y emplear cómputo paralelo serán temas a abordar.
- Aplicar alguna de las dos técnicas anteriores permitirá explorar su comportamiento en casos reales con múltiples atributos y casos de aprendizaje.

8 Acrónimos y definiciones

1-NN	"*1-Nearest-Neighbor*". Técnica de clasificación basada en atribuir a un caso la misma clase que la que tiene su vecino más próximo.
∞-NN	"*∞-Nearest-Neighbor*". Técnica (propuesta en esta tesis) de clasificar a un caso en función de la proximidad ponderada a todos los casos conocidos.
ANN	"*Artificial Neural Network*". Red neuronal artificial. Es un algoritmo de clasificación y regresión muy popular. Se basa en optimizar los pesos de las conexiones existentes entre nodos; dichos nodos suman el valor de todas sus conexiones entrantes y proporcionan un valor de salida, relacionado de forma no lineal con dicha suma.
BTW	"*Best of Three Worlds*". Acrónimo que se ha dado a uno de los algoritmos desarrollados en esta tesis basado en una métrica global.
CBR	"*Case-Based Reasoning*". Denominación genérica para aquellos algoritmos que obtienen conclusiones de un conjunto de experiencias pasadas, con el fin de identificar situaciones similares y reutilizar las soluciones que fueron exitosas.
COD	"*Curse of Dimensionality*". Se emplea para referirse al problema de la maldición de la dimensionalidad.
CV	"*Cross Validation*". Validación cruzada. Método para estimar la precisión real de una clasificación basándose solamente en casos de aprendizaje.
DANN	"*Discriminant Adaptive Nearest-Neighbor*". Método introducido por Hastie y Tibshirani para mejorar la precisión de los algoritmos de clasificación basados en los vecinos próximos.

ERM	"*Empirical Risk Minimization*". Técnica que minimiza el riesgo de clasificar erróneamente un caso.
FKT	"*Fukunaga-Koontz transform*".
FMS	"*Flexible Manufacturing System*". Es un sistema de fabricación industrial diseñado para producir una gran variedad de piezas en lotes pequeños y con una gran productividad.
IA	"*Inteligencia Artificial*". También conocida por su acrónimo en inglés: AI.
kB	Kilobyte.
k-NN	"*k-Nearest-Neighbor*". Técnica de clasificación basada en atribuir a un caso la misma clase que la que tienen la mayoría de sus *k* vecinos más próximos.
"Kernel"	Función de similitud que permite transformar datos del espacio de atributos ("*input space*") al de las características ("*feature space*").
LaMaNNa	"*Large Margin Nearest Neighbor*". Algoritmo desarrollado por Carlotta Domeniconi y sus colaboradores. Su objetivo es ponderar los distintos atributos de un caso en función de la dirección del gradiente de la frontera de separación entre clases.
LDA	"*Linear Discriminant Analysis*". Es un método de clasificación que intenta buscar un plano en el espacio de los atributos que separe de forma óptima clases cuyos atributos responden a una distribución normal.
LMNN	"*Largest Margin Nearest Neighbor*". Algoritmo desarrollado por Kilian Weinberger y sus colaboradores que pretende obtener una margen mínimo de separación entre casos de distintas clases.
LOM	"*Locally Oriented Metric*". Acrónimo que se ha dado a uno de los algoritmos desarrollados en esta tesis basado en una métrica local.
LOO	"*Leave One Out*", es una técnica de validación cruzada donde se emplean todos los datos, excepto el actual, para predecir la clase del caso actual.

LP "*Linear Programming*". Técnica de optimización con restricciones donde la función a optimizar es lineal.

LSH "*Locality-Sensitive Hashing*". Método de búsqueda rápida de casos similares propuesta por Alexandr Andoni basada en funciones sensibles a la proximidad.

ML "*Machine Learning*". Área de conocimiento de la IA relacionada con las técnicas que permiten incrementar el conocimiento y habilidades de los sistemas de computación.

MSE "*Mean Square Error*". Error cuadrático medio. $SME = \frac{1}{N}\sum_{i=1}^{N}\left(\hat{Y}_i - Y_i\right)^2$

NCA "*Neighbourhood Component Analysis*". Algoritmo de clasificación propuesto por Jacob Goldberger que se basa en optimizar una matriz completa de tipo Mahalanobis.

PCA "*Principal Component Analysis*". Análisis de componentes principales. Es un método que realiza una transformación ortogonal para convertir un conjunto de variables correlacionadas en otro que no presenta esta propiedad.

PDF "*Probability Density Function*". Función de densidad de probabilidad.

PR-CG "*Polak-Ribiere Conjugate Gradient*". Minimización mediante el método del gradiente conjugado, siguiendo el algoritmo de Polak-Ribiere.

QDA "*Quadratic Discriminant Analysis*". Es un método de clasificación que intenta buscar una superficie cónica en el espacio de los atributos que separe de forma óptima clases. Los atributos de ambas clases responden a una distribución normal pero tienen distinta matriz de covarianzas.

QP "*Quadratic Programming*". Técnica de optimización con restricciones donde la función a optimizar es cuadrática.

RAM "*Random Access Memory*". Memoria de acceso directo. Es la memoria del ordenador donde se almacenan los datos que manejan los programas.

RBF	"*Radial Basis Function*". Es una función cuyo valor depende de la distancia al origen, habitualmente se trata de una función exponencial con exponente negativo. La suma ponderada de RBF se usa para aproximar otras funciones, tanto en redes neuronales artificiales como en SVM.
RCA	"*Relevant Component Analysis*". Algoritmo de clasificación desarrollado por Noam Shental y que usa relaciones de equivalencia para optimizar una métrica de tipo Mahalanobis.
SDP	"*Semidefinite Programming*". Técnica de optimización que pretende minimizar una función lineal sujeto a la restricción de que la matriz usada en la función a minimizar pertenezca al cono de matrices semidefinido positivas.
SMO	"*Sequential Minimal Optimization*". Método de optimización para problemas convexos que se utiliza para calcular los vectores de soporte en las SVM.
SOM	"*Self-Organizing Maps*". También denominado mapa autoorganizado, es un tipo de red neuronal que se usa para el aprendizaje no supervisado.
SRM	"*Structural Risk Minimization*". Principio inductivo en IA que pretende equilibrar el error empírico cometido por un modelo frente a su complejidad, en aras a disminuir el riesgo de generalización.
SV	"*Support Vector*". Vector de soporte.
SVD	"*Singular Value Decomposition*". Técnica del álgebra de matrices que permite descomponer una matriz cualquiera como producto de dos matrices ortogonales (no necesariamente una transpuesta de la otra) y una matriz diagonal.
SVM	"*Support Vector Machines*". Máquinas de vectores de soporte. Método de clasificación (y regresión) que intenta reducir el error en las estimaciones futuras por medio de separar lo más posible de las fronteras los puntos que se encuentran más próximos a ellas.
SVM-RBF	Algoritmo de máquina de vectores de soporte que utiliza como "*kernel*" una RBF de tipo exponencial (con exponente negativo).

Valor propio	"*Eigenvalue*". Es el factor escalar que relaciona las normas de un vector propio antes y después de someterle a la transformación lineal que nace de su operador.
VC	Dimensión de "*Vapnik-Chervonenkis*".
Vector propio	"*Eigenvector*". Son los vectores no nulos de un operador lineal que cuando son transformados por el operador dan lugar a un múltiplo escalar de sí mismos.
VSM	"*Variable-kernel Similarity Metric*". Algoritmo desarrollado por David G. Lowe que optimizaba los pesos de los atributos en una clasificación de tipo *k*-NN.

9 Notación

$\in$	Pertenece a…
$\cup$	Unión.
$\cap$	Intersección.
$\forall$	Para todo…
$\exists$	Existe al menos un…
| o :	Tal que…
$\therefore$	Por lo tanto...
δ_{ij}	Es la delta de kronecker. Su valor es 1 cuando coinciden los índices i y j (normalmente se refiere al elemento de una matriz), siendo 0 en caso contrario.
$\nabla f()$	Es el gradiente de la función $f()$. Es una magnitud vectorial formada por las derivadas parciales de primer orden de una función escalar.
$\nabla^2 f()$	Es el Hessiano de la función $f()$. Es una matriz bidimensional formada por las derivadas parciales de segundo orden de una función escalar.
ϵ	Hace referencia a un número suficientemente pequeño.
ρ	Es, en general, una función de medida de distancia asociada a un espacio métrico.
$\boldsymbol{\Sigma}$	Es la matriz de covarianzas de los atributos de los casos.
$\succcurlyeq 0$	Mediante esta expresión se indica que la matriz que le antecede debe ser semidefinida positiva.
$\succ 0$	Mediante esta expresión se indica que la matriz que le antecede debe ser definida positiva.
$\boldsymbol{A} \succcurlyeq \boldsymbol{B}$	Es una contracción para sirve para indicar que la diferencia de esas dos matrices es semidefinida positiva: $(\boldsymbol{A} - \boldsymbol{B}) \in \mathcal{S}_+^n$

$\lvert\ \rvert$	Dependiendo del contexto puede tener distintos significados: • Si se aplica a un valor numérico escalar indica que se obtendrá su valor absoluto. • Si se aplica a un conjunto significa obtener su cardinalidad. • Si se aplica sobre una matriz, será el operador que permite obtener su determinante.
$\lVert\ \rVert$	Norma asociada a un espacio vectorial. Si presenta un superíndice significa que está elevado a esa potencia. Si presenta un subíndice indica el parámetro p de la distancia de Minkowsky que emplea esta norma.
$(\ )^T$	Mediante el superíndice T se indica la transpuesta del vector o matriz.
$[z]_+$	Denota la función de pérdida Hinge. Se evalúa como $max(0,1-z)$. Su uso está extendido en las SVM.
$\{\boldsymbol{x}, y\}_m$	Representa el caso m-ésimo de un problema, donde se conocen tanto sus atributos $\boldsymbol{x}$, como su clase y.
A	Número de atributos que describen un problema. Es también la dimensión del espacio de dichos atributos.
$c_j(\boldsymbol{x}_m)$	Es una función que se evalúa como 1 si el caso m-ésimo pertenece a la clase j-esima, será 0 en caso contrario.
$d(\boldsymbol{x}_m, \boldsymbol{x}_i)$	Medida de distancia entre los casos m-ésimo e i-ésimo.
$\boldsymbol{G}$	Es el tensor de una métrica riemanniana.
g_{ij}	Es un elemento del tensor de la métrica riemanniana.
gr_i	Es el elemento i-ésimo de un vector gradiente.
$\mathcal{H}$	Espacio de Hilbert.
$\boldsymbol{I}$	Matriz unitaria.
J	Número de clases del problema.
$\boldsymbol{J}$	Matriz del jacobiano. Está formada por las derivadas parciales de primer orden de una función vectorial.
$\boldsymbol{M}$	Matriz de una métrica de tipo Mahalanobis.
$\mathbb{N}$	Conjunto de los números naturales.
$\boldsymbol{\mathcal{P}}$	Conjunto de parámetros para un problema de optimización. Es un vector.
p	Se usa para indicar la probabilidad. En el caso de las distancias de Minkowski se usa para indicar el exponente.
$\mathbb{R}$	Conjunto de los números reales.
$\mathbb{R}^+$	Conjunto de los números reales positivos incluyendo el 0.
$\mathbb{R}^N$	Conjunto cartesiano N-dimensional de números reales.

$\mathcal{S}^n$	Representa el conjunto de matrices simétricas de dimensión n.
$\mathcal{S}_+^n$	Representa el conjunto de matrices semidefinidas positivas de dimensión n.
$tr(\ \ )$	Indica el cálculo de la traza de la matriz que se encuentra dentro del paréntesis.
$\mathcal{V}$	Es un espacio vectorial en general.
$\boldsymbol{x}$	Representa, en general, todos los atributos de un caso (es un vector).
$\boldsymbol{x}_m$	Es un vector con todos los atributos del caso m-ésimo. Por extensión, el subíndice indica la posición del elemento en cuestión en una colección ordenada.
$x_{m,i}$	Es el valor del atributo i-ésimo del caso m-ésimo (es un escalar).
$\boldsymbol{x}^{(i)}$	Es el valor de los atributos de un caso en la iteración i-ésima. Por extensión, un superíndice entre paréntesis representa el valor del elemento en cuestión en esa iteración.
$\mathcal{X}$	Espacio original o espacio de los atributos.
y	Es, en general, la clase a la que pertenece un caso.
y_j	Indica que el caso pertenece a la clase j-ésima.
$\mathcal{Z}$	Espacio resultante de una transformación lineal.

10 Referencias bibliográficas

[1] P. McCorduck, Machines Who Think: 25th anniversary edition, Natick, MA: A K Peters, Ltd, 2004.

[2] F. Jäkel, B. Schölkopf and F. A. Wichmann, "Similarity, kernels, and the triangle inequality," *Journal of Mathematical Psychology,* vol. 52, no. 5, pp. 297-303, 2008.

[3] B. Kulis, "International Conference on Machine Learning, Metric Learning Tutorial," 21 06 2010. [Online]. Available: http://www.eecs.berkeley.edu/~kulis/icml2010_tutorial.htm. [Accessed 01 08 2012].

[4] Asuncion, A; Newman, D.J., «UCI Machine Learning Repository,» University of California, Irvine, School of Information and Computer Sciences, 2007. [En línea]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html. [Último acceso: 09 03 2015].

[5] A. Luntz y V. Brailovsky, «On estimation of characters obtained in statistical procedure of recognition,» *Technicheskaya Kibernetica,* 1969.

[6] B. Efron and R. Tibshirani, An Introduction to the Bootstrap, New York: Chapman & Hall, 1996.

[7] J. Friedman, "Another Approach to Polychotomous Classification," Technical report, Stanford University, 1996.

[8] C. M. Bishop, Pattern Recognition and Machine Learning, Springer Verlag, 2006.

[9] H. Chih-Wei, C. Chih-Chung and L. Chih-Jen, "A Practical Guide to Support Vector Classification," National Taiwan University, Taipei , 15 4 2010. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf. [Accessed 1 4 2015].

[10] J. FÄurnkranz, "Round robin classification," *Journal of Machine Learning Research,* vol. 2, pp. 721-747, 2002.

[11] C. Hsu and C. Lin, "A comparison of methods for multi-class support vector machines," *IEEE Transactions on Neural Networks,* vol. 2, pp. 415-425, 2002.

[12] V. N. Vapnik, Statistical learning theory, Wiley, 1998.

[13] R. Rifkin and A. Klautau, "In Defense of One-Vs-All Classification," *Journal of Machine Learning Research,* vol. 5, pp. 101-141, 2005.

[14] R. O. Duda, P. E. Hart and D. G. Stork, Pattern Classification, 2nd Edition, NY: Wiley, 2000.

[15] Š. Raudys, "Taxonomy of Pattern Classification Algorithms," in *Statistical and Neural Classifiers. An Integrated Approach to Design*, London, Springer-Verlag, 2001, pp. 22-75.

[16] T. Hastie, R. Tibshirani and J. Friedman, The Elements of Statistical Learning, 2nd Edition, NY: Springer-Verlag, 2009.

[17] X. Zhu and A. B. Goldberg, Introduction to Semi-Supervised Learning, Morgan & Claypool Publishers, 2009.

[18] E. P. Xing, M. I. Jordan, S. J. Russell and A. Y. Ng, "Distance Metric Learning with Application to Clustering with Side-Information," *Advances in Neural Information Processing Systems,* vol. 15, pp. 505-512, 2002.

[19] A. Bar-Hillel, T. Hertz, N. Shental and D. Weinshall, "Learning distance functions using equivalence relations," in *Proceeedings of the International Conference on Machine Learning*, 2003.

[20] A. Fawzi, O. Fawzi and P. Frossard, "Analysis of classifiers' robustness to adversarial perturbations," 02 03 2015. [Online]. Available: http://dblp.uni-trier.de/rec/bib/journals/corr/FawziFF15. [Accessed 15 06 2015].

[21] S. Chopra, R. Hadsell and Y. LeCun, "Learning a similiarty metric discriminatively, with application," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR-05)*, San Diego, CA, 2005.

[22] H. Samet, Foundations of Multidimensional and Metric Data Structures, San francisco (CA): Morgan Kaufmann, Elsevier, 2006.

[23] H. Lejsek, F. H. Ásmundsson, B. Þ. Jónsson and L. Amsaleg, "Nv-tree: An efficient disk-based index for approximate search in very large high-dimensional collections," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* vol. 31, no. 5, pp. 869-883, 2009.

[24] J. H. Friedman, F. Baskett and L. J. Shustek, "An algorithm for finding nearest neighbors," *IEEE Transactions on Computers,* vol. 24, no. 10, pp. 1000-1006, 1975.

[25] J. L. Bentley and J. H. Friedman, "Data structures for range searching," *ACM Computing Surveys,* vol. 11, no. 4, pp. 397-409, 1979.

[26] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," *Communications of the ACM,* vol. 51, no. 1, pp. 117-122, 2008.

[27] R. A. Finkel and J. L. Bentley, "Quad trees: a data structure for retrieval on composite keys," *Acta Informatica,* vol. 4, no. 1, pp. 1-9, 1974.

[28] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Communications of the ACM,* vol. 18, p. 509–517, 1975.

[29] J. H. Friedman, J. L. Bentley and R. A. Finkel, "An algorithm for finding best matches in logarithmic expected time," *ACM Transactions on Mathematical Software,* vol. 3, no. 3, pp. 209-226, 1977.

[30] I. Jolliffe, Principal Component Analysis, 2nd Ed., Springer Series in Statistics, 2002.

[31] Neos, "Optimization Taxonomy," University of Wisconsin, 2013. [Online]. Available: http://www.neos-guide.org/content/optimization-taxonomy. [Accessed 21 03 2015].

[32] J. Nocedal and S. J. Wright, Numerical Optimization, New York: Springer Science+Business Media, 2006.

[33] D. E. Knuth, The Art of Computer Programming, 3rd Ed., Addison Wesley, 2011.

[34] W. Press, A. Teukolsky, W. Vetterling and B. Flannery, Numerical recipes in C: the art of scientific computing, 2nd Ed., New York: Cambridge University Press, 1992.

[35] C.-W. Hsu, C.-C. Chang and C.-J. Lin, "A Practical Guide to Support Vector Classification," Department of Computer Science, National Taiwan University, Taipei, 2010.

[36] C. Paar, J. Pelzl and B. Preneel, Understanding Cryptography: A Textbook for Students and Practitioners, Springer Verlag, 2010.

[37] D. W. Marquardt, "An Algorithm for Least-Squares Estimation of Nonlinear Parameters," *Journal of the Society for Industrial and Applied Mathematics,* vol. 11, p. 431–441, 1963.

[38] D. E. Rumelhart, G. E. Hinton and R. J. Williams, "Learning representations by back-propagating errors," *nature,* vol. 323, pp. 533-536, 1986.

[39] S. Haykin, Neural Networks And Learning Machines, 3rd Ed., Prentice Hall, 2010.

[40] D. Luenberger and Y. Yinyu, Introduction to Linear and Nonlinear Programming, 3rd ed., New York: Springer, 2008.

[41] L. Vandenberghe and S. Boyd, "Applications of Semidefinite Programming," *Applied Numerical Mathematics,* vol. 29, pp. 283-299, 1999.

[42] A. Man-Cho So and Y. Ye, "Theory of semidefinite programming for Sensor Network Localization," in *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms*, New York, 2005.

[43] L. Tunçel, "Some Applications of Semidefinite Optimization from an Operations Research Viewpoint," Department of Combinatorics and Optimization, Faculty of Mathematics, University of Waterloo, Waterloo, Ontario, 2008.

[44] J. Zhiqiang, "Semi-Definite Programming for Power Output Control in a Wind Energy Conversion System," *IEEE Transactions on Sustainable Energy,* vol. 5, no. 2, pp. 466-475, 2014.

[45] J. Bao, J. F. Forbes and P. J. McLellan, "Robust Multiloop PID Controller Design: A Successive Semidefinite Programming Approach," *Industrial & Engineering Chemistry Research,* vol. 38, no. 9, pp. 3407-3419, 1999.

[46] B. Riemann, Über die Hypothesen, welche der Geometrie zu Grunde liegen (Klassische Texte der Wissenschaft), Berlín: Springer-Verlag, 2013.

[47] A. Pressley, Elementary Differential Geometry, London: Spinger-Verlag, 2010.

[48] M. Berger, A Panoramic View of Riemannian Geometry, Berlín: Springer-Verlag, 2002.

[49] M. P. d. Carmo, Riemannian Geometry, Cambridge, MA: Birkhäuser Boston, 1992.

[50] D. W. Aha and R. L. Goldstone, "Concept learning and flexible weighting," in *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society*, Bloomington, IN, 1992.

[51] M. M. Richter, "Classification and learning of similarity measures," in *Proceedings of the 16th Annual Conference of the "Gesellschaft für Klassifikation e.V." University of Dortmund*, Dortmund, 1993.

[52] W. J. Scheirer, M. J. Wilber, M. Eckmann and T. E. Boult, "Good Recognition is Non-metric," *Pattern Recognition,* vol. 47, no. 8, pp. 2721-2731, 2013.

[53] M. M. Deza and E. Deza, Encyclopedia of Distances 3rd Ed., Springer, 2014.

[54] E. Pekalska, Doctoral Dissertation: Dissimilarity representations in pattern recognition, Delft: Delft University of Technology, 2005.

[55] D. G. Lowe, "Similarity metric learning for a variable-kernel classifier," *Neural Computation,* vol. 7, no. 1, pp. 72-85, 1995.

[56] P. C. Mahalanobis, "On the generalised distance in statistics," *Proceedings of the National Institute of Sciences of India,* vol. 2, no. 1, pp. 49-55, 1936.

[57] T. Pang-Ning, M. Steinbach and V. Kumar, Introduction to Data Mining, Addison-Wesley, 2005.

[58] A. Singhal, "Modern Information Retrieval: A Brief Overview," *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering,* vol. 24, no. 4, pp. 35-43, 2001.

[59] M. R. Daliri, "Chi-square distance kernel of the gaits for the diagnosis of Parkinson's disease," *Biomedical Signal Processing and Control,* vol. 8, no. 1, pp. 66-70, 2013.

[60] J. Zhang, M. Marszalek, S. Lazebnik and C. Schmid, "Local features and kernels for classification of texture and object categories: A comprehensive study," *International Journal of Computer Vision,* 2007.

[61] T. Hastie and R. Tibshirani, "Discriminant adaptive nearest neighbor classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 18, no. 6, p. 607–616, 1996.

[62] C. Domeniconi, J. Peng and D. Gunopulos, "An adaptive metric machine for pattern classification, Proceedings of the 2000 conference on Neural Information Processing Systems (NIPS)," in *Advances in Neural Information Processing Systems 13*, 2000.

[63] C. Domeniconi, Gunopulos, Dimitrios and J. Peng, "Large margin nearest neighbor classifiers," *IEEE Transactions on Neural Networks,* vol. 16, no. 4, pp. 899-909, 2005.

[64] J. R. Bray and J. T. Curtis, "An ordination of upland forest communities of southern Wisconsin," *Ecological Monographs,* vol. 27, pp. 325-349, 1957.

[65] P. J. Somerfield, "Identification of the Bray-Curtis similarity index: comment on Yoshioka," *Marine Ecology Progress Series,* vol. 372, pp. 303-306, 2008.

[66] K. Clarke, P. Somerfield and M. Chapman, "On resemblance measures for ecological studies, including taxonomic dissimilarities and a zero-

adjusted Bray-Curtis coefficient for denuded assemblages," *Journal of Experimental Marine Biology and Ecology,* vol. 330, pp. 55-80, 2006.

[67] G. Lance and T. W. Williams, "Mixed-data classificatory programs, I.) Agglomerative Systems," *Australian Computer Journal,* vol. 1, pp. 15-20, 1967.

[68] G. Jurman, S. Riccadonna, R. Visintainer and C. Furlanello, "Canberra Distance on Ranked Lists," in *Proceedings, Advances in Ranking – NIPS 09 Workshop*, 2009.

[69] S. M. Emran and N. Ye, "Robustness of chi-square and Canberra distance metrics for computer intrusion detection," *Quality and Reliability Engineering International,* vol. 18, pp. 19-28, 2002.

[70] G. J. Székely, M. L. Rizzo and N. K. Bakirov, "Measuring and testing independence by correlation of distances," *The Annals of Statistics,* vol. 35, no. 6, pp. 2769-2794, 2007.

[71] A. Fujita, J. R. Sato, M. A. A. Demasi, M. C. Sogayar, C. E. Ferreira and S. Miyano, "Comparing Pearson, Spearman and Hoeffding's D measure for gene expression," *Journal of bioinformatics and computational biology,* vol. 7, no. 4, pp. 663-684, 2009.

[72] M. Tomassini, L. Vanneschi, P. Collard and M. Clergue, "A Study of Fitness Distance Correlation as a Difficulty Measure in Genetic Programming," *Evolutionary Computation,* vol. 13, no. 2, pp. 213-239, 2006.

[73] R. W. Hamming, "Error detecting and error correcting codes," *Bell System Technical Journal,* vol. 29, no. 2, pp. 147-160, 1950.

[74] A. Tversky, "Features of similarity," *Psychological Review,* vol. 84, no. 4, pp. 327-352, 1977.

[75] D. Kahneman, Thinking, Fast and Slow, London: Penguin, 2011.

[76] M. M. Richter and S. Wess, "Similarity, Uncertainty and Case-Based Reasoning in PATDEX," in *Automated Reasoning*, Springer, 1991, pp. 249-265.

[77] P. Jaccard, "Distribution de la flore alpine dans le bassin des Dranses et dans quelques régions voisines," *Bulletin de la Société Vaudoise des Sciences Naturelles,* vol. 37, pp. 241-272, 1901.

[78] D. J. Rogers and T. T. Tanimoto, "A Computer Program for Classifying Plants," *Science,* vol. 132, no. 3434, pp. 1115-1118, 1960.

[79] P. Willet, "Similarity-based virtual screening using 2D fingerprints," *Drug Discovery Today,* vol. 11, no. 23-24, pp. 1046-1053, 2006.

[80] M. A. Fligner, J. S. Verducci and P. E. Blower, "A Modification of the Jaccard–Tanimoto Similarity Index for Diverse Selection of Chemical Compounds Using Binary Strings," *Technometrics,* vol. 44, no. 2, pp. 110-119, 2002.

[81] C. R. Rao, "The utilization of multiple measurements in problems of biological," *Journal of the Royal Statistical Society,* Vols. Series B, 10, pp. 159-193, 1948.

[82] Z. E. Chay, C. H. Lee, K. C. Lee, J. S. Oon and M. H. Ling, "Russel and Rao Coefficient is a Suitable Substitute for Dice Coefficient in Studying Restriction Mapped Genetic Distances of Escherichia coli," *Computational and Mathematical Biology,* vol. 1, pp. 1-9, 2010.

[83] R. S. Teegavarapu, Floods in a Changing Climate: Extreme Precipitation, Cambridge University Press, 2012.

[84] L. R. Dice, "Measures of the amount of ecologic association between species," *Ecology,* vol. 26, no. 6, 1945.

[85] C. Brun, F. Chevenet, D. Martin, J. Wojcik, A. Guénoche and B. Jacq, "Functional classification of proteins for the prediction of cellular function from a protein-protein interaction network," *Genome Biology,* vol. 5, no. 1, p. R6, 2003.

[86] V. I. Levenshtein, "Binary-codes capable of correcting spurious insertion and deletion of ones," *Problems of Information Transmission,* vol. 1, pp. 8-17, 1965.

[87] E. Brill and R. C. Moore, "An Improved Error Model for Noisy Channel Spelling Correction," in *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, Stroudsburg, PA, 2000.

[88] G. V. Bard, "Spelling-error tolerant, order-independent pass-phrases via the Damerau–Levenshtein string-edit distance metric," in *Proceedings of the Fifth Australasian Symposium on ACSW Frontiers*, Ballarat, Australia, 2007.

[89] M. Li, Y. Zhang, M. Zhu and M. Zhou, "Exploring distributional similarity based models for query spelling correction," in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, Stroudsburg, PA, 2006.

[90] K. A. Majorek, S. Dunin-Horkawicz, K. Steczkiewicz, A. Muszewska, M. Nowotny , K. Ginalski and J. M. Bujnicki , "The RNase H-like superfamily: new members, comparative structural analysis and evolutionary classification," *Nucleic Acids Research,* vol. 42, no. 7, pp. 4160-4179, 2013.

[91] F. J. Damaerau, "A technique for computer detection and correction of spelling errors," *Communications of the ACM (ACM),* vol. 7, no. 3, pp. 171-176, 1964.

[92] S. B. Needleman and C. D. and Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of Molecular Biology,* vol. 48, no. 3, pp. 443-453, 1970.

[93] T. F. Smith and M. S. Waterman, "Identification of common molecular subsequences," *Journal of Molecular Biology,* vol. 147, no. 1, pp. 195-197, 1981.

[94] M. Zhao, W.-P. Lee, E. Garrison and G. Marth, "SSW Library: An SIMD Smith-Waterman C/C++ Library for Use in Genomic Applications," *PLoS ONE,* vol. 8, no. 12, p. e82138, 2013.

[95] R. Yan, D. Xu, J. Yang, S. Walker and Y. Zhang, "A comparative assessment and analysis of 20 representative sequence alignment methods for protein structure prediction," *Nature, Scientific reports.,* vol. 3, no. 2619, pp. 1-9, 2013.

[96] R. Dieny, J. Thevenon, J. Martinez del Rincon and J.-C. Nebel, "Bioinformatics inspired algorithm for stereo correspondence," in *International Conference on Computer Vision Theory and Applications*, Vilamoura - Algarve, 2011.

[97] R. Irving, "Plagiarism and collusion detection using the Smith-Waterman algorithm," Dept of Computing Science, University of Glasgow , 2004.

[98] A. Bar-Hillel, T. Hertz, N. Shental and D. Weinshall, "Learning a Mahalanobis Metric from Equivalence Constraints," *Journal of Machine Learning Research,* vol. 6, p. 937–965, 2005.

[99] N. Shental, T. Hertz, D. Weinshall and M. Pavel, "Adjustment learning and relevant component analysis," in *Proceedings of the Seventh European Conference on Computer Vision (ECCV-02)*, London, 2002.

[100] J. Goldberger, S. Roweis, G. Hinton and R. Salakhutdinov, "Neighbourhood Components Analysis," *Advances in Neural Information Processing Systems,* vol. 17, pp. 513-520, 2005.

[101] J. Friedman, "Flexible metric nearest neighbor classification," Tech. Report. Stanford University, Statistics Department, 1994.

[102] Z. Zhang, J. Kwok and D. Yeung, "Parametric distance metric learning with label information," in *Proceedings of the International Conference on Artificial Intelligence*, 2003.

[103] C. Domeniconi and D. Gunopulos, "Adaptive Nearest Neighbor Classification using Support Vector Machines," in *Advances in Neural Information Processing Systems 14*, 2002.

[104] J. Peng, D. R. Heisterkamp and H. Dai, "Adaptive kernel metric nearest neighbor classification," in *Proceedings. 16th International Conference on Pattern Recognition*, 2002.

[105] J. Peng, D. R. Heisterkamp and H. Dai, "LDA/SVM Driven Nearest Neighbor Classification," *IEEE Transactions on Neural Networks*, vol. 14, no. 4, pp. 940-942, 2003.

[106] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE transactions on Information Theory*, pp. 21-27, 1967.

[107] E. Fix and J. Hodges, "Discriminatory analysis, nonparametric discrimination: Consistency properties," Technical Report 4, USAF School of Aviation Medicine, Randolph Field, Texas, 1951.

[108] D. Michie, D. J. Spiegelhalter and C. C. Taylor, Machine Learning, neural and statistical classification, Englewood Cliffs, NJ: Overseas Press, 2009.

[109] S. Belongie, J. Malik and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, pp. 509-522, 2002.

[110] P. Y. Simard, Y. LeCun and J. Decker, "Efficient pattern recognition using a new transformation distance," in *Advances in Neural Information Processing 6*, vol. 6, San Mateo, CA, Morgan Kaufman, 1993, pp. 50-58.

[111] B. V. Dasarathy, Nearest Neighbor: Pattern Classification Techniques, Hoboken (NJ): IEEE Computer Society, 1990.

[112] J. Liangxiao, C. Zhihua, W. Dianhong and S. Jiang, "Survey of Improving K-Nearest-Neighbor for Classification," in *FSKD 2007. Fourth International Conference on Fuzzy Systems and Knowledge Discovery*, 2007.

[113] N. Bhatia and V. Ashev, "Survey of Nearest Neighbor Techniques," *International Journal of Computer Science and Information Security*, vol. 8, no. 2, pp. 302-305, 2010.

[114] R. D. Short and K. Fukunaga, "The Optimal Distance Measure for Nearest Neighbor Classification," *IEEE Transactions on information theory*, vol. 27, no. 5, pp. 622-627, 1981.

[115] K. Fukunaga and T. E. Flick, "An Optimal Global Nearest Neighbor Metric," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 6, no. 3, pp. 314-318, 1984.

[116] S. Zhang and T. Sim, "Discriminant Subspace Analysis: A Fukunaga-Koontz Approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 29, no. 10, pp. 1732-1745, 2007.

[117] D. Kong and C. Ding, "Pairwise-Covariance Lineal Discriminant Analysis," in *Proceedings of the twenty-eighth AAAI Conference on Artificial Intelligence*, 2014.

[118] V. N. Vapnik, The Nature of Statistical Learning, NY: Springer-Verlag, 1996.

[119] N. Cristianini and J. Shawe-Taylor, An introduction to Support Vector Machines, Cambridge, MA: Cambridge University Press, 2000.

[120] B. Schölkopf and A. J. Smola, Learning with Kernels, Cambridge, MA: MIT Press, 2002.

[121] B. Boser, I. Guyon and V. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the 5th Annual ACM Workshop on Computational*, Pittsburgh, 1992.

[122] D. Wolpert, "The Lack of A Priori Distinctions Between Learning Algorithms," *Neural Computation,* vol. 8, pp. 1341-1390, 1996.

[123] J. C. Platt, "Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines," Microsoft, Technical Report MSR-TR-98-14, Redmond, WA, 1998.

[124] H. V. Nguyen and L. bai, "Cosine Similarity Metric Learning for Face Verification," in *Proccedings of the 10th Asian conference on computer vision (ACCV)*, 2010.

[125] B. Kulis, "Metric Learning: A Survey," *Foundations and Trends in Machine Learning,* vol. 5, no. 4, pp. 287-364, 2012.

[126] S. Shalev-Shwartz, Y. Singer and A. Y. Ng, "Online and batch learning of pseudo-metrics," in *Proceedings of the Twenty First International Conference on Machine Learning (ICML-04)*, Banff, Canada, 2004.

[127] N. Nguyen and Y. Guo, "Metric Learning: A Support Vector Approach," in *Machine Learning and Knowledge Discovery in Databases 5212, Lecture Notes in Computer Science*, Berlin, Springer, 2008, pp. 125-136.

[128] K. Q. Weinberger and L. K. Saul, "Distance Metric Learning for Large Margin Nearest Neighbor Classification," *Journal of Machine Learning Research,* vol. 10, pp. 207-244, 2009.

[129] J. Davis, B. Kulis, P. Jain, S. Sra and I. Dhillon, "Information-theoretic metric learning," in *Proceedings of International Conference on Machine Learning (ICML)*, 2007.

[130] C. Domeniconi, J. Peng and D. Gunopulos, "Locally adaptive metric nearest-neighbor classification," *IEEE Pattern analysis and Machine Intelligence,* vol. 24, no. 9, pp. 1281-1285, 2002.

[131] Q. Chang, Q. Chen and X. Wang, "Scaling Gaussian RBF Kernel Width to Improve SVM Classification," in *Proceedings of the Int. Conference on neural networks and brain, ICNN&B'05*, 2005.

[132] J. J. Naudé, M. A. van Wyk y B. J. van Wyk, «Generalized Similarity Metric Learning for a Variable Kernel Classifier,» *Lecture Notes in Computer Science,* vol. 3138, pp. 788-796, 2004.

[133] S.-I. Amari and S. Wu, "Improving support vector machine classifiers by modifying," *Neural Networks,* vol. 12, pp. 783-789, 1999.

[134] S. Boyd and L. Vandenberghe, Convex optimization, Cambridge University Press, 2004.

[135] V. Zorich, "Quasi-conformal mapping," Encyclopedia of Mathematics, 07 02 2011. [Online]. Available: URL: http://www.encyclopediaofmath.org/index.php?title=Quasi-conformal_mapping&oldid=13506. [Accessed 31 03 2015].

[136] P. Williams, S. Li, J. Feng and S. Wu, "Scaling the Kernel Function to Improve Performance of the Support Vector Machine," in *Advances in Neural Networks, ISNN'05*, 2005.

[137] S. Wu and S.-I. Amari, "Conformal transformation of kernel functions: a data-dependent way to improve support vector machine classifiers," *Neural Processing Letters,* vol. 15, pp. 59-67, 2002.

[138] G. Wu and E. Y. Chang, "Adaptive Feature-space Conformal Transformation for Imbalances-data Learning," in *Proceedings of the 20th Int. Conference on Machine Learning (ICML-2003)*, Washington DC, 2003.

[139] J. M. Revilla and E. Kahoraho, "BTW: a New Distance Metric for Classification," in *Proc. of the International Symposium on Distributed Computing and Artificial Intelligence, DCAI 2012*, Salamanca, 2012.

[140] J. M. Revilla and E. Kahoraho, "A new metric for real-time case based classification and sensor fusion," in *ENMA Scientific 2013, International Conference* , Bilbao, 2013.

[141] J. M. Revilla and E. Kahoraho, "LOM, a Locally Oriented Metric which Improves Accuracy in Classification Problems," in *Proceedings of the Seventh International Conference on Information, Process, and Knowledge Management*, Lisbon, 2015.

[142] H. Lütkepohl, Handbook of matrices, John Wiley & Sons Ltd., 1996.

[143] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, Introduction to Algorithms, 3rd Edition, Cambridge, MA: MIT Press, 2009.

[144] L. Breiman, J. Friedman, R. Olshen and C. Stone, Classification and regression trees, Beltmont, CA.: Wadsworth International Group, 1984.

[145] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annual Eugenics,* vol. II, pp. 179-188, 1936.

[146] J. Quinlan, "Simplifying decision trees," *International Journal of Man-Machine Studies,* vol. 27, pp. 221-234, 1987.

[147] K. Bennett and O. Mangasarian, "Robust Linear Programming Discrimination of Two Linearly Inseparable Sets," *Optimization Methods and Software,* vol. 1, pp. 23-34, 1992.

[148] D. W. Aha and D. Kibler, "Instance-based prediction of heart-disease presence with the Cleveland database," University of California, 1980.

[149] J. Siebert, "Vehicle Recognition Using Rule Based Methods," Turing Institute Research Memorandum, TIRM-87-018, 1987.

[150] P. Zezula, G. Amato, V. Dohnal and M. Batko, Similarity search. The metric space approach, NY: Springer Science+Bussines media Inc., 2006.

[151] B. Silverman, Density Estimation for Statistics and Data Analysis, London: Chapman and Hall, 1986.

[152] R. E. Bellman, Adaptive Control Processes, Princeton University Press, 1961.

[153] P. Pokorny, "Geodesics Revisited," *Chaotic Modeling and Simulation,* pp. 281-298, 2012.

[154] I. Schoenberg, "Metric spaces and positive definite functions," *Transactions of the American Mathematical Society,* vol. 44, pp. 522-536, 1938.

Anexos

A1 Casos de estudio utilizados en esta investigación

Para valorar la bondad de los algoritmos desarrollados es necesario de disponer de un conjunto de casos.

Estos casos se pueden separar en dos grandes bloques:

- Casos sintéticos: son artificiales y han sido generados mediante un programa, con ellos es posible investigar sobre una determinada cualidad de los algoritmos.
- Casos reales: han sido extraídos de escenarios reales, se han elegido aquellos que se usan repetidamente en investigaciones de renombre. Algunos son muy habituales, como los de las bases de datos de la UCI [4], que han sido la principal fuente de esta investigación. Se ha pretendido deliberadamente que la mayor parte de los problemas hayan sido usados en proyectos de renombre como "StatLog" [1] [108], financiado por la Comisión Europea bajo el programa ESPRIT.

A1.1 Problemas sintéticos

Los problemas sintéticos que han sido utilizados en esta tesis son:

- "Waveforms".
- Escuadra diagonal.

A1.1.1 "Waveforms"

Es un problema artificial creado por Breiman, el cual simula la existencia de dos ondas que son sumadas formando tres combinaciones.

Presenta veintiún atributos que están contaminados con ruido. Su interés nace de la gran cantidad de atributos que presentan fuertes correlaciones y la presencia del ruido.

Características:

N. de atributos	21
Tipo de atributos	Continuos
N. de clases	3
N. de casos de aprendizaje	300
N. de casos de prueba	4.700

[1] El título oficial del proyecto es: "The comparative testing of statistical and logical learning algorithms on large-scale applications to classification, prediction and control".

Origen y usos:

Lo introdujo Breiman [144] para estudiar sus árboles de clasificación y regresión.

A1.1.2 Escuadra diagonal

Es un problema bidimensional con dos clases creado para esta investigación. Se ha utilizado en el estudio empírico del algoritmo LOM.

La primera clase está formada por tres bloques obtenidos, aleatoriamente, cada uno de ellos de distribuciones gaussianas con medias:

$$\begin{bmatrix}0\\2\end{bmatrix} \quad \begin{bmatrix}2\\0\end{bmatrix} \quad \begin{bmatrix}4\\4\end{bmatrix}$$

Y matrices de covarianzas:

$$\begin{bmatrix}0{,}1 & 0\\ 0 & 1\end{bmatrix} \quad \begin{bmatrix}1 & 0\\ 0 & 0{,}1\end{bmatrix} \quad \begin{bmatrix}1 & -1/\sqrt{2}\\ -1/\sqrt{2} & 1\end{bmatrix}$$

La segunda clase está formada por un único bloque procedente de una distribución gaussiana con media y covarianzas:

$$\begin{bmatrix}2\\2\end{bmatrix} \qquad \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$$

Se generaron distintos bloques de datos de aprendizaje (de 30, 60, 90, 180 y 1.200 casos) y un gran bloque de datos de test de 6.000 casos.

Características:

N. de atributos	2
Tipo de atributos	Continuos
N. de clases	2
N. de casos de aprendizaje	30, 60, 90, 180, 1.200
N. de casos de prueba	6.000

Origen y usos:

Solo en esta tesis doctoral.

A1.2 Problemas reales

El uso de problemas reales en una investigación acerca los objetivos de esta (resultados que se obtienen) a los condicionantes del mundo práctico en el que se pretenderán usar. Desde este punto de vista su uso es ineludible en un trabajo que se precie.

Los problemas reales que han sido utilizados en esta tesis son:

- Flores del Iris.
- "Australian Credit".
- "Breast Cancer".
- "German Credit".
- "Glass".
- "Heart".
- "Image Segmentation".
- "Satellite Image".
- "Shuttle Control".
- "Vehicle Silhouettes".

A1.2.1 Flores del Iris

El conjunto de datos tabula cuatro características (longitud y anchura de sépalos y pétalos) de tres especies de flores de Iris (Setosa, Versicolor y Virgínica) mediante 150 ejemplos.

Es sin duda el problema de clasificación más empleado a lo largo de las investigaciones. Su popularidad procede de que Fisher [145] lo uso en el año 1936 para ilustrar el funcionamiento del algoritmo que presentaba: el análisis discriminante.

Los resultados que obtienen casi todos los algoritmos de clasificación son muy buenos, es un problema muy simple ya que la especie Setosa se puede separar linealmente de las otras dos; estas últimas no son linealmente separables pero entre ellas hay poco solapamiento.

Es de destacar que entre los atributos que reflejan la longitud y anchura de los pétalos existe una gran correlación.

Características:

N. de atributos	4
Tipo de atributos	Continuos
N. de clases	3
N. de casos de aprendizaje	150
N. de casos de prueba	-

Origen y usos:

Se ha usado en casi todos los estudios importantes de clasificación. Solo se citarán unos pocos:

- Fisher en su estudio del análisis discriminante.
- Duda y Hart [14] en su libro de referencia sobre técnicas de clasificación.

A1.2.2 "Australian Credit"

Es un problema en el que se intenta determinar si a un cliente se le puede conceder una determinada tarjeta de crédito. Para ello la entidad bancaria recopiló catorce atributos de cada cliente y en función de ellos se intenta tomar la decisión. Para garantizar la confidencialidad, tanto el identificador del cliente como el significado de los atributos han sido eliminados de la base de casos.

Es un problema interesante, tanto por la mezcla de atributos continuos y nominales como porque, de diversos estudios, se cree que solo son relevantes los atributos 5º, 8º, 9º, 13º y 14º.

Características:

N. de atributos	14
Tipo de atributos	Continuos (6) y Nominales (8)
N. de clases	2
N. de casos de aprendizaje	690
N. de casos de prueba	-

Origen y usos:

Este problema lo introdujo Quinlan [146] en su artículo sobre el "podado" de árboles de decisión.

Se empleó como problema de referencia en el proyecto europeo "StatLog".

A1.2.3 "Breast Cancer"

Es un problema basado en datos médicos fruto de la colaboración entre los departamentos de informática y ciencias clínicas de la Universidad de Wisconsin entre los años 1989 y 1992.

Los atributos se obtienen de imágenes digitalizadas de una biopsia mamaria (en esas fotos se describen las características de los núcleos de las células).

Las imágenes fueron procesadas mediante programación lineal para obtener de cada núcleo celular 9 atributos tales como radio, perímetro, desviación estándar de los tonos de gris,... [147].

En este problema existe una proporción de 2/3 a 1/3 entre los casos clasificados como benignos y malignos.

Características:

N. de atributos	9
Tipo de atributos	Continuos
N. de clases	2
N. de casos de aprendizaje	699
N. de casos de prueba	-

Origen y usos:

En su origen estuvo ligado a la investigación médica y a métodos de clasificación basados en árboles de decisión.

Posteriormente se ha usado en otro tipo de algoritmos: redes neuronales artificiales (ANN), máquinas de vectores de soporte,...

A1.2.4 "German Credit"

El objeto de este problema es clasificar de una forma rápida a los posibles clientes de un banco para saber si son candidatos para la concesión de un crédito.

Los datos originales presentaban numerosos atributos categóricos y simbólicos. La Universidad de Strathclyde convirtió dichos atributos a formato numérico para facilitar su uso.

Entre dichos atributos se encuentran los bienes actuales, historia crediticia pasada, estado civil, tipo de trabajo,...

En su formato original también disponía de una matriz de riesgo (para de ella derivar una función de pérdida).

Características:

N. de atributos	24
Tipo de atributos	Valores enteros
N. de clases	2
N. de casos de aprendizaje	1.000
N. de casos de prueba	-

Origen y usos:

El origen de los datos es el Instituto de estadística y econometría de la Universidad de Hamburgo.

Se empleó como problema de referencia en el proyecto europeo "StatLog".

A1.2.5 "Glass"

El análisis y clasificación de restos de vidrio encontrados en escenarios de crimen es una técnica habitual en los departamentos policiales. Mediante ella se puede determinar el origen de un determinado fragmento y su posible procedencia.

Los principales atributos que componen este problema son la composición química del fragmento de vidrio y su índice de refracción.

Los algoritmos de clasificación deben tratar de determinar a cuál de las 6 posibles clases pertenece.

Características:

N. de atributos	9
Tipo de atributos	Continuos
N. de clases	6
N. de casos de aprendizaje	214
N. de casos de prueba	-

Origen y usos:

El origen de los datos es una recopilación realizada por Vina Spiehler, de la compañía "Diagnostic Products Corporation".

Es uno de los problemas clásicos en los estudios de clasificación, ha sido utilizado con clasificadores de tipo *k*-NN, ANN, SVM, árboles de decisión, algoritmos genéticos,...

A1.2.6 "Heart"

Su objetivo es pronosticar una enfermedad cardiaca basándose en un conjunto de datos fácilmente recopilables.

Los principales atributos de este problema se derivan de los contenidos típicos de una anamnesis médica y de sus posteriores pruebas analíticas: edad, sexo, presión sanguínea, tasa de colesterol, comportamiento ante el ejercicio,...

En esta investigación se ha usado la misma versión que se empleó en el proyecto europeo "StatLog".

Características:

N. de atributos	13
Tipo de atributos	Continuos
N. de clases	2
N. de casos de aprendizaje	270
N. de casos de prueba	-

Origen y usos:

Su primera versión nació de la fusión de cuatro bases de datos del Instituto de cardiología de Budapest, los hospitales universitarios de Zurich y Basilea, el centro médico de veteranos de Long Beach y la Fundación Clínica de Cleveland, todas ellas procesadas, analizadas y fusionadas por David W. Aha [148].

Posteriormente, el número de atributos se redujo de 76 a 13 y se limitó a los casos presentes en la base de datos de Cleveland.

Se empleó como problema de referencia en el proyecto europeo "StatLog".

A1.2.7 "Image Segmentation"

Es un problema de clasificación de imágenes de exteriores. Se proporcionan 19 atributos (medias de colores RGB, saturación,…) de cada imagen de 3x3 píxeles, con el objetivo de conocer cuál es su fuente. Hay 7 posibles fuentes: ladrillos, cielo, follaje, cemento, ventanas, caminos y hierba.

Características:

N. de atributos	19
Tipo de atributos	Continuos
N. de clases	7
N. de casos de aprendizaje	2.310
N. de casos de prueba	-

Origen y usos:

Los datos los ha proporcionado el Grupo de Visión de la Universidad de Massachusetts.

Se empleó como problema de referencia en el proyecto europeo "StatLog".

A1.2.8 "Satellite Image"

El problema parte de imágenes de la tierra tomadas por un satélite de la NASA, donde cada píxel representa aproximadamente un área de 80x80 metros. Se tomó una pequeña zona de la imagen original (82x100 píxeles) y se la dividió en regiones de 3x3 píxeles; de cada píxel se proporciona como atributo los valores de la intensidad de la emisión en cuatro bandas espectrales (verde, rojo y dos en el infrarrojo cercano) codificados como valores en el rango [0 255]. Así pues, para cada caso hay 36 atributos y una clase entre las siete posibles (suelo rojo, algodón, suelo gris, suelo gris húmedo, suelo con vegetación, mezcla de clases, suelo gris muy húmedo).

Los atributos que ocupan las posiciones 17, 18, 19 y 20 son los que corresponden al píxel central.

Características:

N. de atributos	36
Tipo de atributos	Continuos
N. de clases	7
N. de casos de aprendizaje	4.435
N. de casos de prueba	2.000

Origen y usos:

Los datos originales del Landsat fueron comprados a la NASA por el "Australian Centre for Remote Sensing" de la Universidad de Nueva Gales del Sur, en Australia.

Se empleó como problema de referencia en el proyecto europeo "StatLog".

A1.2.9 "Shuttle Control"

Son datos proporcionados por la NASA que están relacionados con la posición de fuentes de radiación en el trasbordador espacial. Se dispone de 9 atributos y 7 clases cuyos nombres son: "Rad Flow", "Fpv Close", "Fpv Open", "High", "Bypass", "Bpv Close".

Aproximadamente el 80% de los casos corresponden a la clase 1, por lo tanto este sería el límite inferior de precisión para cualquier algoritmo.

Es un problema en el que los mejores algoritmos consiguen tasas de acierto superiores al 99%, ya que los atributos se suponen libres de ruido y se proporcionan gran cantidad de casos.

Su dificultad radica en que las distribuciones de los atributos son multimodales, este problema es fácil para los algoritmos de tipo árbol de clasificación.

Es posible que los datos estén ordenados temporalmente, aunque esto no se tiene en cuenta para la clasificación.

Características:

N. de atributos	9
Tipo de atributos	Continuos
N. de clases	7
N. de casos de aprendizaje	43.500
N. de casos de prueba	14.500

Origen y usos:

Los datos proceden de Jason Catlett del "Basser Department of Computer Science", en la Universidad de Sydney, Australia.
Se empleó como problema de referencia en el proyecto europeo "StatLog".

A1.2.10 "Vehicle Silhouettes"

Es una base de casos que tiene como objetivo el reconocer objetos de tres dimensiones por medio de sus representaciones bidimensionales (siluetas). Para ello toma cuatro tipos de vehículos (autobús de doble plataforma, furgoneta Chevrolet, Saab 9000 y Opel Manta 400) y de sus imágenes se obtienen 18 atributos ("circularidad" de la imagen, varianza escalada a lo largo de los ejes mayor y menor,…).
La gran dificultad proviene en que los casos son visiones de los vehículos tomadas desde distintos ángulos, con lo cual su semejanza sí es aparente para el ser humano, pero muy difícil para algoritmos genéricos de clasificación.

Características:

N. de atributos	18
Tipo de atributos	Continuos
N. de clases	4
N. de casos de aprendizaje	846, 761
N. de casos de prueba	-

Origen y usos:

Los casos fueron recogidos por JP Siebert [149] en el "Turing Institute" para sus estudios sobre clasificación basada en reglas.
Se empleó como problema de referencia en el proyecto europeo "StatLog".