Universidad de Valladolid

Facultad de Ciencias

Trabajo Fin de Máster

Máster en Matemáticas

Aspectos Matemáticos en Máquinas de Vectores Soporte (SVM)

Autor: Álvaro Valera Blanco

Tutor/es: Luis Ángel García Escudero

Índice general

Capítulo 1

Introducción al aprendizaje y aprendizaje supervisado

El Aprendizaje Automático, es una disciplina compartida entre la Informática, las Matemáticas y la Estadística que puede considerarse englobada bajo un epígrafe más general de Inteligencia Artificial, cuyo objetivo es desarrollar técnicas que permitan que las computadoras "aprendan". Existen dos tipos de Aprendizaje Automático: el supervisado y no supervisado. Cuando se desea "predecir" variables a partir de aprender del comportamiento en datos proporcionados sobre una muestra patrón o de entrenamiento, es lo que se conoce como aprendizaje supervisado. Si no se dispone de ninguna muestra patrón o de entrenamiento, se habla de aprendizaje no supervisado y el objetivo es encontrar patrones o grupos de interés aprendiendo directamente del conjunto de datos.

El Aprendizaje Estadístico juega un papel muy importante en muchas áreas como las finanzas y la industria y, en general, es uno de los pilares de la Ciencia de Datos (Data Science).

El objetivo principal del aprendizaje supervisado es crear o estimar alguna función que sea capaz de poder predecir el valor de nuevas observaciones. Para ello lo que hace es generalizar lo que ha aprendido del conjunto de datos de entrenamiento a una nueva situación que no se haya visto anteriormente. Existen varios métodos dentro del aprendizaje supervisado en función del valor de salida, de clasificación si es una etiqueta de clase y de regresión si es un valor numérico.

Diversos campos de las Matemáticas han sido útiles dentro del Aprendizaje Automático. En este trabajo nos centraremos en algunas de las aportaciones matemáticas a los *Support Vector Machines*.

Los *Support Vector Machines*, conocidos en castellano como máquinas de vectores de soporte (SVM), es una técnica de clasificación de aprendizaje supervisado que tiene su origen en los años 90 con Vladimir Vapnik, y co-autores, y ha ido ganando popularidad desde entonces en muchos campos como la Estadística, la Minería de Datos y la Inteligencia Artificial. Esto se debe a sus sólidos fundamentos teóricos, ya que utiliza conceptos bien establecidos y desarrollados de la teoría de optimización. Esta rama de las Matemáticas es muy importante para el aprendizaje supervisado y los *Support Vector Machines* en particular.

Lo que se trata de hacer es una predicción a partir de un problema de optimización geométrica, el cual se puede escribir como un problema de optimización cuadrático conve-

xo con restricciones lineales, que por lo general, puede resolverse mediante procedimientos no lineales. Para ello se pretende buscar un hiperplano de separación óptimos tras, en la mayoría de las ocasiones, llevar el problema a espacios de dimensiones mayores donde los datos transformados puedan estar mejor separados.

Originariamente los SVM's se presentaron como una técnica para resolver problemas de clasificación binaria. Actualmente, se pueden utilizar para resolver otro tipo de problemas, por ejemplo, multiclase, de regresión o en detección de anomalías.

El objetivo de este Trabajo de Fin de Máster (TFM) es analizar numerosos aspectos de carácter más matemático que subyacen bajo la metodología SVM. En primer lugar, en los Capítulos 2 y 3, nos centraremos en la clasificación binaria, sobre todo en los aspectos computacionales tanto del caso linealmente separable como del no separable linealmente. A continuación, en el Capítulo 4, veremos cómo si los datos no son separables linealmente, podemos pasar a dimensiones superiores donde sí serán más fácilmente separables y utilizaremos lo que se conoce como "kernel trick". En el siguiente Capítulo 5, veremos un algoritmo, que se ha desarrollado relativamente hace poco, que sirve para resolver los problemas de optimización relacionados con la solución de los SVMs. En el Capítulo 6, se mostrara la implementación práctica de los SVM con el lenguaje de programación R. Por último, en el Capítulo 7, trataremos alguna extensión de los SVMs, junto con algunas conclusiones y posibles direcciones futuras.

Para el desarrollo de este TFM se ha llevado a cabo un proceso de recopilación y análisis de información procedente de distintas fuentes que abordan esta temática. Los libros escogidos han sido [1, 2, 3, 4]. La elección de estos libros se ha basado en su reconocimiento como obras de referencia en el área de estudio, cada uno aportando una perspectiva valiosa sobre algunos aspectos clave de los SVM. Al unificar y dar coherencia a los fragmentos seleccionados de cada uno de estos libros, se ha tratado de establecer un hilo conductor común, buscando asegurar cierta consistencia en la presentación de contenidos. Se han realizado análisis comparativos e identificado los puntos clave en los que los libros seleccionados proporcionaban una explicación más clara o ilustrativa y estos fragmentos se han integrado en la memoria del TFM tratando de ofrecer una perspectiva matemáticamente razonada del SVM.

Capítulo 2

Caso separable linealmente (*Hard-SVM*)

Un problema de aprendizaje binario de dos clases es un problema en el que se busca encontrar un procedimiento para clasificar elementos en dos grupos diferentes. En este tipo de problemas se tiene una variable objetivo, o variable a predecir, binaria, que puede tomar dos valores posibles y que en el SVM se suele codificar como 1 o -1. En el contexto de Aprendizaje Automático, el problema de optimización binaria a menudo se refiere a la clasificación binaria, donde se busca encontrar un modelo que pueda separar dos clases diferentes de datos. Supongamos que tenemos disponible un conjunto de datos de aprendizaje, es decir, sea

$$\mathcal{C} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n) : \mathbf{x}_i \in \mathbb{R}^p, y_i \in \{-1, 1\}\} \tag{2.1}$$

un conjunto de observaciones de entrenamiento, donde cada par $(\mathbf{x}_i, y_i)$ se conoce como *observación*, siendo el vector $\mathbf{x}_i$ un conjunto de p características, también conocido como vector de variables predictoras e y_i la clase a la que pertenece dicha observación, la variable que tenemos que predecir.

El problema de clasificación binaria en el cual nos centraremos en los próximos capítulos se basa en usar el conjunto de observaciones $\mathcal{C}$ para construir una función $f : \mathbb{R}^p \to \mathbb{R}$ tal que

$$Q(\mathbf{x}) = \text{sign}(f(\mathbf{x})) \tag{2.2}$$

sea un clasificador. La función separadora f nos permite por tanto, separar cada nuevo punto $\mathbf{x}$ en las dos clases existentes, dependiendo si $Q(\mathbf{x})$ es 1, es decir, si $f(\mathbf{x}) \geq 0$, o es -1, si $f(\mathbf{x}) < 0$.

Existen dos tipos de conjuntos de observaciones de entrenamiento $\mathcal{C}$ que nos permitirán realizar distintos tipos de clasificación. Puede suceder que todas las observaciones del conjunto se puedan separar mediante un hiperplano, es decir, cada una está en el semiplano de la clase correspondiente, entonces diremos que el conjunto de observaciones es *separable*, o que no pueden separarse todas las observaciones, por tanto diremos que el conjunto es *no separable*.

En este capítulo nos centraremos en el caso linealmente separable. Primero veamos la definición de hiperplano:

Definición 2.1. Un *hiperplano* es un subespacio afín plano de dimensión $p-1$ y, de forma analítica, está definido por

$$\left\{\mathbf{x} = (x_1, ..., x_p)^T : \ \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p = 0\right\}, \tag{2.3}$$

para los parámetros $\beta_0, \beta_1, \ldots, \beta_p$ en $\mathbb{R}$.

De manera equivalente, se puede escribir en forma matricial como

$$\left\{\mathbf{x} : \ \beta_0 + \beta^T \mathbf{x} = 0\right\} \tag{2.4}$$

Propiedades 2.2. *Algunas propiedades que presentan los hiperplanos son las siguientes*

- *Para dos puntos cualquiera del hiperplano* $\mathbf{x}_1$ *y* $\mathbf{x}_2$ *se cumple* $\beta^T(\mathbf{x}_1 - \mathbf{x}_2) = 0$ *y por tanto* $\beta^* = \beta/\|\beta\|$ *es el vector normal al hiperplano.*
- *Para cualquier punto* $\mathbf{x}_0$ *del hiperplano, se tiene* $\beta^T \mathbf{x}_0 = -\beta_0$.
- *La distancia de cualquier punto* $\mathbf{x}$ *del espacio* S *al hiperplano viene dada por*

$$\beta^{*T}(\mathbf{x} - \mathbf{x}_0) = \frac{1}{\|\beta\|}(\beta^T \mathbf{x} + \beta_0) = \frac{1}{\|\beta\|} f(\mathbf{x}), \tag{2.5}$$

  Por tanto $f(\mathbf{x})$ *es proporcional a la distancia desde* $\mathbf{x}$ *al hiperplano definido por* $f(\mathbf{x}) = 0$.

A lo largo de la historia, han sido varios los algoritmos que han ido apareciendo para resolver el problema de la separación lineal. Un antecedente muy importante de los SVM es el "perceptrón" propuesto en 1956 por Frank Rosenblatt, que discutiremos en el Apéndice A.

Decimos que un conjunto $\mathcal{C}$ como el que se proporciona en (2.1), es *separable linealmente* si existe un hiperplano que es capaz de separar dicho conjunto clasificando correctamente todas las observaciones en función del lado del hiperplano en el queden las observaciones. Un hiperplano de este tipo será conocido como *hiperplano de separación* y proporciona una función lineal

$$f(\mathbf{x}) = \beta^T \mathbf{x} + \beta_0, \tag{2.6}$$

donde $\beta \in \mathbb{R}^p$ y $\beta_0 \in \mathbb{R}$, que nos permite clasificar nuevas observaciones en función del signo de $f(\mathbf{x})$ como se ha comentado en (2.2).

En consecuencia, para todas las observaciones del conjunto de entrenamiento $\mathcal{C}$, se deben cumplir las siguientes restricciones

$$f(\mathbf{x}_i) = \beta^T \mathbf{x}_i + \beta_0 > 0 \quad \text{si } y_i = 1, \tag{2.7}$$

y

$$f(\mathbf{x}_i) = \beta^T \mathbf{x}_i + \beta_0 < 0 \quad \text{si } y_i = -1. \tag{2.8}$$

Ambas ecuaciones (2.7) y (2.8) pueden escribirse de forma única como

$$y_i(\beta^T \mathbf{x}_i + \beta_0) = y_i f(\mathbf{x}_i) > 0 \quad \text{para todo } i = 1, \ldots, n. \tag{2.9}$$

Este hiperplano de separación de las dos clases si existe, no suele ser único, existe un número infinito de dichos hiperplanos. Esto lo podemos observar en la Figura 2.1, ya que un hiperplano puede desplazarse un poco hacia arriba o hacia abajo y rotar sin llegar a tocar ninguna de las observaciones.

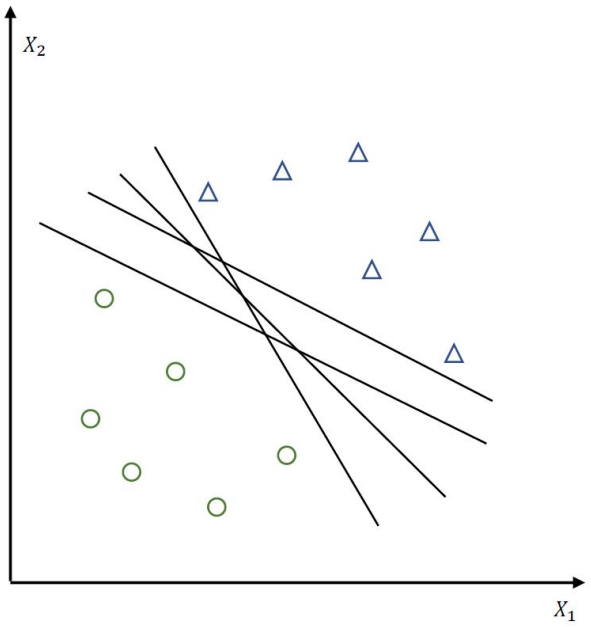

Figura 2.1: Múltiples hiperplanos separando dos clases linealmente separables

Un método para elegir el mejor hiperplano que separe las observaciones de entre esos infinitos que hay, es elegir aquel que esté más alejado de todas las observaciones. Para ello se define el "margen" M como la mínima distancia entre el hiperplano de separación y la observación más próxima a él de las dos clases.

Este hiperplano que separa a las dos clases y maximiza la distancia a las observaciones más cercanas de cada una de ellas se conoce como "hiperplano de margen máximo" o también como "hiperplano de separación óptimo". A esta regla de clasificación se la conoce como *Hard-SVM*.

Por hipótesis hemos supuesto que el conjunto de observaciones, $\mathcal{C}$, es linealmente separable, por tanto, existen β y β_0 tales que se cumplen las siguientes desigualdades para cualquier observación del conjunto de entrenamiento,

$$\beta^T \mathbf{x}_i + \beta_0 \geq +1, \quad \text{si } y_i = +1 \tag{2.10}$$

$$\beta^T \mathbf{x}_i + \beta_0 \leq -1, \quad \text{si } y_i = -1. \tag{2.11}$$

Aquellas observaciones que cumplan la igualdad en (2.10), pertenecen al hiperplano H_{+1} : $\beta^T \mathbf{x}_i + \beta_0 = 1$, y aquellas que cumplan la igualdad en (2.11) están en H_{-1} : $\beta^T \mathbf{x}_i + \beta_0 = -1$. A estas observaciones que caen en alguno de estos dos hiperplanos, es decir, están justo en el margen del *Hard-SVM* se las conoce como *vectores soporte*, del inglés "support vectors", y veremos como van a jugar un papel muy importante en los SVM's.

El *Hard-SVM* proporciona una solución única al problema del hiperplano de separación como se puede observar en la Figura 2.2. Además se puede ver que en este conjunto de datos hay 3 vectores soporte.

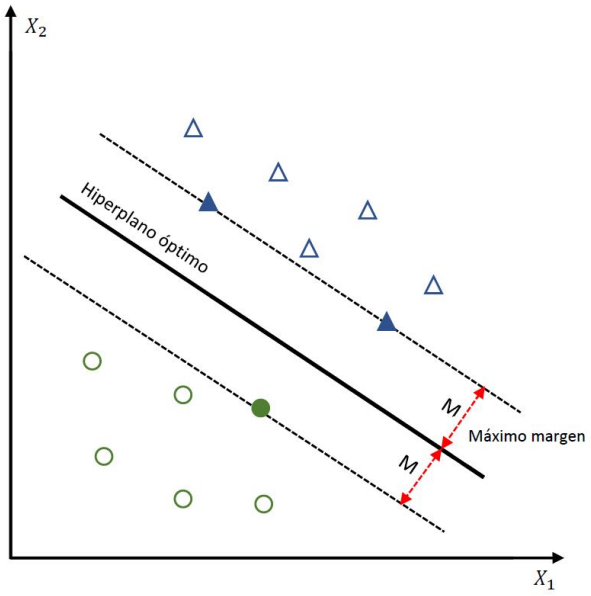

Figura 2.2: *Hard-SVM.* 3 vectores soporte marcados en color más oscuro.

A continuación vamos a caracterizar esta regla de clasificación.

Teorema 2.3. *Un hiperplano de separación es de margen máximo si y solo si está a la misma distancia de la observación más cercana de cada una de las clases.*

Demostración. Por reducción al absurdo, sean los pares $(\mathbf{x}_i, y_i)$ y $(\mathbf{x}_j, y_j)$, donde $y_i = +1$ e $y_j = -1$, las observaciones más cercanas de cada una de las clases al hiperplano de separación óptimo. Supongamos que la distancia del vector de variables predictoras $\mathbf{x}_i$ al hiperplano es mayor que la distancia correspondiente del vector $\mathbf{x}_j$ a este. Esto quiere decir que se puede acercar el hiperplano hacia la observación $\mathbf{x}_i$ una distancia tal que la distancia del hiperplano a esa observación sea menor que antes y además siga siendo mayor que la distancia a la observación $\mathbf{x}_j$. Esto es una contradicción, porque se puede aumentar el tamaño del margen cuando hemos supuesto por hipótesis que era máximo. □

La idea de este hiperplano es que nos pueda proporcionar un mejor rendimiento de clasificación para datos de un nuevo conjunto test, pues maximiza el margen entre las dos clases en los datos de entrenamiento. A este margen entre las dos clases lo denotaremos por $L = 2M$.

Veamos ahora la formulación matemática de la regla de clasificación *Hard-SVM*. El hiperplano de separación óptimo que buscamos para el *Hard-SVM* es la solución del siguiente problema de optimización,

$$\begin{aligned} \max_{\beta,\beta_0} \quad & M \\ \text{sujeto a:} \quad & \|\beta\| = 1 \\ & y_i \left(\mathbf{x}_i^T \beta + \beta_0\right) \geq M, \; i = 1, \ldots, n \end{aligned} \tag{2.12}$$

o expresado de manera equivalente,

$$\begin{aligned} \max_{\beta,\beta_0,\|\beta\|=1} \quad & M \\ \text{sujeto a:} \quad & y_i \left(\mathbf{x}_i^T \beta + \beta_0\right) \geq M, \; i = 1, \ldots, n \end{aligned} \tag{2.13}$$

El conjunto de las restricciones del problema (2.13) asegura que todas las observaciones distan del hiperplano de separación definido por los parámetros β y β_0 al menos M, y se busca maximizar ese valor de M y los parámetros asociados del hiperplano.

La expresión de la distancia de un punto al hiperplano definida en (2.5) como $|f(\mathbf{x})|/\|\beta\|$, junto con la desigualdad (2.9), nos permite obtener la siguiente desigualdad que deben cumplir todos las observaciones del conjunto de entrenamiento,

$$\frac{y_i f(\mathbf{x}_i)}{\|\beta\|} \geq M,\ i = 1, \ldots, n, \tag{2.14}$$

donde $f(\mathbf{x}_i) = \mathbf{x}_i^T \beta + \beta_0$.

Se puede deducir de esta desigualdad (2.14) que encontrar el hiperplano de separación óptimo es equivalente a encontrar el valor de β que maximice el margen. Esta se puede escribir de forma equivalente como

$$y_i f(\mathbf{x}_i) \geq M\|\beta\|,\ i = 1, \ldots, n. \tag{2.15}$$

Para cualquier β y β_0 que satisfacen la expresión (2.15), todo múltiplo suyo también satisface dicha desigualdad, es decir, existen infinitas soluciones de la forma $\lambda(\beta^T \mathbf{x} + \beta_0)$ con $\lambda \in \mathbb{R}$ que representan el mismo hiperplano. Para eliminar este problema de llegar a varias soluciones, se fija de forma arbitraria y sin pérdida de generalidad que

$$M\|\beta\| = 1, \tag{2.16}$$

para obtener solo una única solución.

De esta igualdad se puede concluir que maximizar el margen es equivalente a minimizar la norma de β y por tanto la condición (2.15) queda como

$$y_i \left(\mathbf{x}_i^T \beta + \beta_0\right) = y_i f(\mathbf{x}_i) \geq 1,\ i = 1, \ldots, n. \tag{2.17}$$

Podemos, por tanto, reescribir las desigualdades (2.10) y (2.11) como una única (2.17). De aquí se puede ver que los vectores soporte con respecto al hiperplano de separación óptimo son aquellos cuyo margen es igual a 1, es decir que cumplen la igualdad en (2.17).

Por todo esto el problema de optimización (2.13) para la búsqueda del hiperplano de separación óptimo para la regla *Hard-SVM* es equivalente a un problema de optimización de tipo cuadrático que se define de la siguiente forma.

$$\begin{aligned} \min_{\beta, \beta_0} \quad & \frac{1}{2}\|\beta\|^2 \\ \text{sujeto a:} \quad & y_i \left(\mathbf{x}_i^T \beta + \beta_0\right) \geq 1, \quad i = 1, \ldots, n. \end{aligned} \tag{2.18}$$

Nótese que es equivalente minimizar $\|\beta\|$ y $\|\beta\|^2/2$. De esta segunda forma, nos permitirá posteriormente simplificar la notación, obteniendo expresiones más compactas.

Para resolver este problema vamos a hacer uso de la teoría de optimización. Hagamos un pequeño recordatorio de algunas ideas de ella.

Un problema general de optimización se puede presentar de la siguiente forma:

Definición 2.4. Dadas las funciones f, g_i, $i = 1, \ldots, k$ y h_i, $i = 1, \ldots, m$ definidas en el dominio $\Omega \subseteq \mathbb{R}^n$, el problema de optimización denominado *primal* se define como

$$\begin{aligned} \text{mín} \quad & f(\mathbf{x}), \quad \mathbf{x} \in \Omega \\ \text{sujeto a:} \quad & g_i(\mathbf{x}) \leq 0, \quad i = 1, \ldots, k, \\ & h_i(\mathbf{x}) = 0, \quad i = 1, \ldots, m. \end{aligned} \tag{2.19}$$

donde f se conoce como la *función objetivo*, y las otras relaciones se denominan respectivamente, restricciones de *desigualdad* e *igualdad*. Al valor óptimo de la función objetivo se le llama valor del problema de optimización.

Dado que los problemas de minimización se pueden convertir en problemas de maximización cambiando el signo de la función objetivo $f(\mathbf{x})$, la elección de la minimización en este problema (2.19) no es ninguna restricción. De manera similar, cualquier restricción en dicho problema puede reescribirse con la desigualdad contraria.

Se define como *región factible* a aquella región del dominio donde se satisfacen todas las restricciones y se denota por

$$R_F = \{\mathbf{x} \in \Omega : \, g(\mathbf{x}) \leq 0, h(\mathbf{x}) = 0\}. \tag{2.20}$$

La solución del problema de optimización primal (2.19) es un punto $\mathbf{x}^* \in R_F$ que cumple que no existe ningún otro punto $\mathbf{x} \in R_F$ tal que $f(\mathbf{x}) < f(\mathbf{x}^*)$. A este punto se le conoce como *mínimo global*. Un punto $\mathbf{x}^* \in \Omega$ es un *mínimo local* de $f(\mathbf{x})$ si existe $\varepsilon > 0$ tal que $f(\mathbf{x}) \geq f(\mathbf{x}^*)$, para todo $\mathbf{x} \in \Omega$ con $\|\mathbf{x} - \mathbf{x}^*\| < \varepsilon$.

En función de cómo sea la función objetivo y las restricciones, existen diferentes problemas de optimización.

Definición 2.5. Un problema de optimización se dice que es *lineal* si la función objetivo y todas sus restricciones son funciones lineales. Se dice que es *cuadrático* cuando la función objetivo es cuadrática y las restricciones son lineales.

Con esta definición, podemos decir que el problema de optimización (2.18) es un problema cuadrático.

Además veamos ahora una clase restringida de problemas de optimización. Para ello, primero definimos lo que es un conjunto convexo y una función convexa.

Definición 2.6. Un conjunto D en un espacio vectorial $\mathcal{S}$ de dimensión p es *convexo* si para dos vectores cualquiera $\mathbf{u}, \mathbf{v} \in D$, el segmento que une $\mathbf{u}$ con $\mathbf{v}$ está contenido en D, es decir, para cualquier $\alpha \in [0, 1]$ se cumple $\alpha\mathbf{u} + (1 - \alpha)\mathbf{v} \in D$.

Dado el conjunto convexo D, una función $f : D \to \mathbb{R}$ es *convexa* si para todos $\mathbf{u}, \mathbf{v} \in D$ y $\alpha \in [0, 1]$,

$$f(\alpha\mathbf{u} + (1 - \alpha)\mathbf{v}) \leq \alpha f(\mathbf{u}) + (1 - \alpha) f(\mathbf{v}).$$

Una propiedad importante de las funciones convexas es que todo mínimo local $\mathbf{x}^*$ de la función es mínimo global. Esto es cierto ya que para cualquier $\mathbf{x} \neq \mathbf{x}^*$, por la definición de mínimo local, existe un $\alpha > 0$ suficientemente pequeño tal que

$$f(\mathbf{x}^*) \leq f(\mathbf{x}^* + \alpha(\mathbf{x} - \mathbf{x}^*)).$$

Como f es una función convexa, se tiene además

$$f(\mathbf{x}^* + \alpha(\mathbf{x} - \mathbf{x}^*)) = f(\alpha\mathbf{x} + (1-\alpha)\mathbf{x}^*)) \leq \alpha f(\mathbf{x}) + (1-\alpha)f(\mathbf{x}^*).$$

Juntando ambas ecuaciones y reordenando los términos, se concluye que $f(\mathbf{x}^*) \leq f(\mathbf{x})$ para cualquier $\mathbf{x}$, por tanto se concluye que $f(\mathbf{x}^*)$ es un mínimo global de f.

Esta propiedad de las funciones convexas es la que hace que los problemas de optimización sean manejables cuando las funciones y conjuntos involucrados son convexos.

Definición 2.7. Un problema de optimización se dice que es *convexo* si la función objetivo y las restricciones son convexas en el conjunto Ω.

Otra propiedad importante de las funciones convexas es que para todo $\mathbf{w} \in \mathcal{S}$ podemos construir un vector tangente a f en $\mathbf{w}$ que se encuentra siempre debajo de la función f. Si f es una función diferenciable, la tangente es la función lineal $g(\mathbf{u}) = f(\mathbf{w}) + \langle \nabla f(\mathbf{w}), \mathbf{u} - \mathbf{w} \rangle$, donde $\nabla f(\mathbf{w})$ es el gradiente de f en $\mathbf{w}$, conocido como el vector de derivadas parciales de f, $\nabla f(\mathbf{w}) = \left(\frac{\partial f(\mathbf{w})}{\partial w_1}, \ldots, \frac{\partial f(\mathbf{w})}{\partial w_p}\right)$. Esto es, para funciones convexas diferenciables se tiene que para todo $\mathbf{u}$ se cumple

$$f(\mathbf{u}) \geq f(\mathbf{w}) + \langle \nabla f(\mathbf{w}), \mathbf{u} - \mathbf{w} \rangle. \tag{2.21}$$

Esta desigualdad la trataremos en capítulos posteriores para el caso de funciones no diferenciales.

Si f es una función escalar diferenciable, hay una forma sencilla de comprobar si es convexa.

Lema 2.8. *Sea $f : \mathbb{R} \to \mathbb{R}$ una función escalar dos veces diferenciable, y sean f' y f'' sus derivadas primera y segunda respectivamente. Son equivalentes las 3 condiciones siguientes:*

1. *f es convexa.*
2. *f' es monótonamente decreciente.*
3. *f'' es no negativa.*

Demostración. Para demostrar que las tres condiciones son equivalentes, vamos a demostrar las siguientes implicaciones $1 \Rightarrow 2 \Rightarrow 3$ y $3 \Rightarrow 2 \Rightarrow 1$.

Comencemos por la primera implicación, vamos a ver $1 \Rightarrow 2$. Supongamos que f es convexa. Entonces, para todo $x, y \in \mathbb{R}$ y todo $\lambda \in [0, 1]$ se cumple que

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y).$$

Tomando la derivada con respecto a λ, obtenemos

$$f'(\lambda x + (1-\lambda)y)(x - y) \leq f(x) - f(y).$$

Dividiendo ambos lados por $x - y$, obtenemos:

$$f'(\lambda x + (1-\lambda)y) \leq \frac{f(x) - f(y)}{x - y}.$$

Lo cual implica que f' es monótonamente decreciente.

Veamos ahora $2 \Rightarrow 3$. Supongamos que f' es monótonamente decreciente. Entonces, para todo $x \in \mathbb{R}$ y todo $h > 0$ se cumple que

$$\frac{f'(x+h) - f'(x)}{h} \leq 0$$

Tomando el límite cuando h tiende a 0 por la derecha, obtenemos que $f''(x) \leq 0$. Si tomamos el límite cuando h tiende a 0 por la izquierda, obtenemos que $f''(x) \geq 0$. Por lo tanto, $f''(x) = 0$, lo que implica que f'' es no negativa.

Para demostrar $3 \Rightarrow 2$, supongamos que f'' es no negativa. Entonces, para todo $x, y \in \mathbb{R}$ con $x < y$, aplicando el teorema del valor medio obtenemos que existe un $c \in (x, y)$ tal que

$$f'(y) - f'(x) = f''(c)(y - x) \geq 0.$$

Lo cual implica que f' es monótonamente decreciente.

Para terminar la demostración, nos queda ver que $2 \Rightarrow 1$. Supongamos que f' es monótonamente decreciente. Entonces, para todo $x, y \in \mathbb{R}$ y todo $\lambda \in [0, 1]$ se cumple que

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y).$$

Tomando la derivada con respecto a λ, obtenemos

$$f'(\lambda x + (1 - \lambda)y)(x - y) < f(x) - f(y).$$

Como f' es monótonamente decreciente, entonces $f'(\lambda x + (1-\lambda)y) \geq f'(y)$. Por lo tanto:

$$f'(y)(x - y) \leq f(x) - f(y).$$

Lo cual implica que f es convexa, y en consecuencia, hemos demostrado que las tres condiciones son equivalentes. □

Además, las siguientes afirmaciones muestran como la composición de una función escalar convexa con una función lineal proporcionan una función vectorial convexa.

Lema 2.9. *Supongamos que $f : \mathbb{R}^d \to \mathbb{R}$ se puede escribir como $f(\mathbf{w}) = g(\langle \mathbf{w}, \mathbf{x} \rangle + y)$ para algún $\mathbf{x} \in \mathbb{R}^d, y \in \mathbb{R}$. Si la función g es convexa, entonces f es convexa.*

Demostración. Sean $\mathbf{w}_1, \mathbf{w}_2 \in \mathbb{R}^d$ y $\alpha \in [0, 1]$. Tenemos

$$\begin{aligned} f\left(\alpha \mathbf{w}_1 + (1-\alpha)\mathbf{w}_2\right) &= g\left(\langle \alpha \mathbf{w}_1 + (1-\alpha)\mathbf{w}_2, \mathbf{x} \rangle + y\right) \\ &= g\left(\alpha \langle \mathbf{w}_1, \mathbf{x} \rangle) + (1-\alpha)\langle \mathbf{w}_2, \mathbf{x} \rangle) + y\right) \\ &= g\left(\alpha\left(\langle \mathbf{w}_1, \mathbf{x} \rangle + y\right) + (1-\alpha)\left(\langle \mathbf{w}_2, \mathbf{x} \rangle + y\right)\right) \\ &\leq \alpha g\left(\langle \mathbf{w}_1, \mathbf{x} \rangle + y\right) + (1-\alpha) g\left(\langle \mathbf{w}_2, \mathbf{x} \rangle + y\right) \end{aligned}$$

donde la última desigualdad viene dada por ser la función g convexa. □

Por último, vamos a ver que el máximo de una función convexa es convexa y la suma ponderada de funciones convexas con pesos no negativos, también es convexa

Lema 2.10. *Sean las funciones convexas $f_i : \mathbb{R}^p \to \mathbb{R}$ para $i = 1, \ldots r$. Las siguientes funciones de $\mathbb{R}^p$ a $\mathbb{R}$ son convexas:*

1. $g(\mathbf{x}) = \text{máx}_{i=1,\ldots,r} f_i(\mathbf{x})$

2. $g(\mathbf{x}) = \sum_{i=1}^{r} w_i f_i(\mathbf{x})$, *donde* $w_i \geq 0$ *para todo* i.

Demostración. Para la primera afirmación se tiene

$$\begin{aligned} g(\alpha u + (1-\alpha)v) &= \max_{i=1,\ldots,r} f_i(\alpha u + (1-\alpha)v) \\ &\leq \max_{i=1,\ldots,r} [\alpha f_i(u) + (1-\alpha) f_i(v)] \\ &\leq \alpha \max_{i=1,\ldots,r} f_i(u) + (1-\alpha) \max_{i=1,\ldots,r} f_i(v) \\ &= \alpha g(u) + (1-\alpha) g(v). \end{aligned}$$

Para la segunda,

$$\begin{aligned} g(\alpha u + (1-\alpha)v) &= \sum_{i=1}^{r} w_i f_i(\alpha u + (1-\alpha)v) \\ &\leq \sum_{i=1}^{r} w_i \left[\alpha f_i(u) + (1-\alpha) f_i(v)\right] \\ &= \alpha \sum_{i=1}^{r} w_i f_i(u) + (1-\alpha) \sum_{i=1}^{r} w_i f_i(v) \\ &= \alpha g(u) + (1-\alpha) g(v). \end{aligned}$$

□

Una vez hemos repasado todas estas definiciones y proposiciones, que nos serán de utilidad a lo largo del TFM, vamos a presentar una técnica de la teoría de optimización conocida como los multiplicadores de Lagrange. Para ello vamos a restringirnos a la clase de problemas de optimización que son convexos y cuadráticos.

Dado el problema de optimización primal (2.19), es decir, aquel que tiene restricciones tanto de igualdad como desigualdad, se define la *función de Lagrange* como

$$\begin{aligned} L_P(\mathbf{x}, \alpha, \mu) &= f(\mathbf{x}) + \sum_{i=1}^{k} \alpha_i g_i(\mathbf{x}) + \sum_{i=1}^{m} \mu_i h_i(\mathbf{x}) \\ &= f(\mathbf{x}) + \alpha g(\mathbf{x}) + \mu h(\mathbf{x}) \end{aligned} \tag{2.22}$$

donde $\alpha = (\alpha_1, \ldots, \alpha_k)$ tal que $\alpha_i \geq 0$ para $i = 1, \ldots, k$ y $\mu = (\mu_1, \ldots, \mu_m)$ se conocen como los *multiplicadores de Lagrange*.

A partir de esta función de Lagrange, que es la función objetivo, junto con las funciones de restricción que es una única, podemos definir el *problema dual* de la siguiente forma

$$\begin{aligned} \max \quad & \theta(\alpha, \mu), \\ \text{sujeto a:} \quad & \alpha \geq 0, \end{aligned} \tag{2.23}$$

donde $\theta(\alpha, \mu) = \inf_{\mathbf{x} \in \Omega} L_P(\mathbf{x}, \alpha, \mu)$.

La teoría de optimalidad proporciona un resultado fundamental en la relación entre el problema primal y el problema dual, asegurando bajo ciertas hipótesis que resolver uno u otro es equivalente, además de proporcionar algún corolario muy utilizado. La

principal ventaja de transformar un problema primal en su dual es que en los problemas de optimización que trataremos, asociados al SVM, suele ser notablemente más fácil resolver el problema dual que el primal. A continuación, mostramos un teorema conocido como el *Teorema de la Dualidad Débil*, que nos da una de las relaciones fundamentales entre ambos problemas, proporcionando la relación existente entre las soluciones de los dos problemas.

Teorema 2.11 (de la Dualidad Débil)**.** *Sea* $\mathbf{x} \in \Omega$ *una solución factible del problema primal* (2.19) *y* (α, μ) *una solución factible del problema dual asociado. Entonces* $f(\mathbf{x}) \geq \theta(\alpha, \mu)$.

Demostración. De la definición de $\theta(\alpha, \mu)$ para $\mathbf{x} \in \Omega$, se tiene

$$\begin{aligned} \theta(\alpha, \mu) &= \inf_{\mathbf{y} \in \Omega} L_P(\mathbf{y}, \alpha, \mu) \\ &\leq L_P(\mathbf{x}, \alpha, \mu) \\ &= f(\mathbf{x}) + \alpha g(\mathbf{x}) + \mu h(\mathbf{x}) \leq f(\mathbf{x}), \end{aligned} \tag{2.24}$$

ya que como $\mathbf{x}$ es una solución factible para el problema primal, implica que cumple todas sus restricciones, es decir, $g(\mathbf{x}) \leq 0$ y $h(\mathbf{x}) = 0$ y (α, μ) es solución factible para el problema dual implica $\alpha \geq 0$. □

Este teorema, como se ha comentado anteriormente, proporciona dos corolarios muy útiles e interesantes. El primero establece que el problema dual está acotado superiormente por el problema primal.

Corolario 2.12. *El valor óptimo del dual está acotado superiormente por el valor óptimo del primal, es decir*

$$\sup\{\theta(\alpha, \mu) : \alpha \geq 0\} \leq \text{ínf}\{f(\mathbf{x}) : g(\mathbf{x}) \leq 0, h(\mathbf{x}) = 0\}.$$

Este otro corolario, permite obtener las soluciones de ambos problemas cuando se cumplen determinadas condiciones.

Corolario 2.13. *Si* $f(\mathbf{x}^*) = \theta(\alpha^*, \mu^*)$ *con* $\alpha^* \geq 0$ *y* $g(\mathbf{x}^*) \leq 0$, $h(\mathbf{x}^*) = 0$, *entonces* $\mathbf{x}^*$ *y* (α^*, μ^*) *son soluciones del problema primal y dual respectivamente. En este caso se cumple que* $\alpha_i^* g_i(\mathbf{x}^*) = 0$ *para* $i = 1, \ldots, k$.

Demostración. Dado que los valores son iguales, la secuencia de desigualdades en la ecuación (2.24) debe convertirse en igualdades. En particular la última desigualdad solo puede ser igualdad si $\alpha_i^* g_i(\mathbf{x}^*) = 0$ para todo i. □

Estamos ahora en condiciones de enunciar el *Teorema de Karush-Kuhn-Tucker* que proporciona las condiciones suficientes, que denotaremos como *condiciones KKT*, para que un vector $\mathbf{x}^*$ sea una solución óptima del problema de optimización primal.

Teorema 2.14 (de Karush-Kuhn-Tucker)**.** *Dado el problema de optimización cuadrático convexo* (2.19)*, la solución óptima* $(\mathbf{x}^*, \alpha^*, \mu^*)$ *existe si y solo si se satisfacen las siguientes*

condiciones

$$\begin{aligned}
\frac{\partial L_p(\mathbf{x}^*, \alpha^*, \mu^*)}{\partial \mathbf{x}} &= 0, \\
\frac{\partial L_p(\mathbf{x}^*, \alpha^*, \mu^*)}{\partial \mu} &= 0, \\
\alpha_i^* g_i(\mathbf{x}^*) &= 0,\ i = 1, \ldots, k, \\
g_i(\mathbf{x}^*) &\leq 0,\ i = 1, \ldots, k, \\
\alpha_i &\geq 0,\ i = 1, \ldots, k.
\end{aligned} \tag{2.25}$$

Supongamos ahora un caso particular en que el problema primal cuadrático convexo no tiene restricciones de igualdad, es decir,

$$\begin{aligned}
\text{mín} \quad & f(\mathbf{x}), \quad \mathbf{x} \in \Omega \\
\text{sujeto a:} \quad & g_i(\mathbf{x}) \leq 0, \quad i = 1, \ldots, k.
\end{aligned} \tag{2.26}$$

Para este problema cuadrático convexo, se define la función de Lagrange como

$$L_P(\mathbf{x}, \alpha) = f(\mathbf{x}) + \sum_{i=1}^{k} \alpha_i g_i(\mathbf{x}) \tag{2.27}$$

donde $\alpha = (\alpha_1, \ldots, \alpha_k)$, $\alpha_i \geq 0$ para $i = 1, \ldots, k$, conocidos como los *multiplicadores de Lagrange*. A partir de esta función de Lagrange, se puede definir como hemos hecho anteriormente, para el caso general, el problema dual como

$$\begin{aligned}
\text{máx} \quad & \theta(\alpha), \\
\text{sujeto a:} \quad & \alpha \geq 0,
\end{aligned} \tag{2.28}$$

donde $\theta(\alpha) = \inf_{\mathbf{x} \in \Omega} L_P(\mathbf{x}, \alpha)$.

Para este problema (2.26), las condiciones KKT que tiene que cumplir un punto $\mathbf{x}^*$ para que sea solución del problema primal vienen dadas por el siguiente resultado.

Teorema 2.15. *Dado el problema de optimización primal* (2.26) *convexo. Si existen las constantes* $\alpha_i \geq 0$, $i = 1, \ldots, k$ *(denominadas multiplicadores de Lagrange) tales que*

$$\begin{aligned}
\frac{\partial L_p(\mathbf{x}^*, \alpha^*, \mu^*)}{\partial \mathbf{x}} = 0 \Leftrightarrow \frac{\partial f(\mathbf{x}^*)}{\partial x_j} + \sum_{i=1}^{k} \frac{\partial g_i(\mathbf{x}^*)}{\partial x_j} &= 0, \quad j = 1, \ldots, n \\
\alpha_i^* g_i(\mathbf{x}^*) &= 0, \quad i = 1, \ldots, k.
\end{aligned} \tag{2.29}$$

entonces $(\mathbf{x}^*, \alpha^*)$ *es solución óptima del problema.*

La primera condición, conocida como primera condición KKT, surge como consecuencia de la definición de la función $\theta(\alpha)$ como el inferior de la función de Lagrange, pues este se haya en el punto donde las derivadas parciales con respecto a $\mathbf{x}$ son cero. La segunda, se denomina condición complementaria, y garantiza que los óptimos de los problemas dual y primal coincidan, $\theta(\alpha^*) = f(\mathbf{x}^*)$, pues esto implica que todos los sumandos del sumatorio de la función de Lagrange serán cero.

El Teorema de Karush-Kuhn-Tucker nos proporciona las condiciones que se tienen que cumplir para resolver el problema primal a partir del problema dual. Primero se

construye a partir del problema primal la función de Lagrange. Se aplica la primera condición de KKT a esta función que hace desaparecer todas las variables primales de esta. Además como hemos comentado antes, este paso es equivalente a calcular el inferior de la función de Lagrange. De esto, se obtiene la función de Lagrange del problema dual, que depende única y exclusivamente de los multiplicadores de Lagrange. De la primera condición KKT puede surgir alguna restricción adicional para los multiplicadores de Lagrange, las variables duales.

Por último, para obtener la solución del problema primal basta con sustituir el resultado obtenido al resolver el problema dual, en las relaciones obtenidas al aplicar la primera condición KKT.

Retomando nuestro problema de optimización cuadrática convexo. Hemos visto un resultado que dice que todo problema de optimización primal con función objetivo y restricciones estrictamente convexas tiene forma dual. En nuestro caso estas hipótesis se cumplen, ya que tanto la función objetivo como las restricciones son funciones convexas, ya que la función objetivo es cuadrática y las restricciones de desigualdad lineales, es decir, es un problema de optimización cuadrático y convexo y por tanto admite dual. Para ello, haremos uso de la función de Lagrange, que en nuestro problema primal ha de ser minimizada con respecto a β y β_0, y es

$$L_P(\beta, \beta_0, \alpha) = \frac{1}{2}\|\beta\|^2 - \sum_{i=1}^{n} \alpha_i \left[y_i \left(\mathbf{x}_i^T \beta + \beta_0 \right) - 1 \right], \tag{2.30}$$

donde $\alpha_i \geq 0, i = 1, \ldots, n$, son los multiplicadores de Lagrange, que se denotan por α, siendo este el vector de multiplicadores de Kuhn-Tucker.
Se puede observar que el signo menos del segundo sumando de esta función de Lagrange se debe a que en las restricciones de (2.18) están expresadas de la forma $g(\mathbf{x}) \geq 0$ en lugar de $g(\mathbf{x}) \leq 0$.

Derivando (2.30) con respecto a las variables β y β_0, que son las variables sobre las que se optimiza el problema primal, e igualando a cero, se tiene la primera condición de KKT.

$$\begin{aligned} 0 &= \frac{\partial L_P(\beta, \beta_0, \alpha)}{\partial \beta} = \beta - \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i, \\ 0 &= \frac{\partial L_P(\beta, \beta_0, \alpha)}{\partial \beta_0} = -\sum_{i=1}^{n} \alpha_i y_i, \end{aligned} \tag{2.31}$$

es decir, las ecuaciones (2.31) nos permiten expresar β en términos de α_i y establecer una restricción adicional para estos multiplicadores de Lagrange, esto es,

$$\begin{aligned} \beta &= \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i, \\ \sum_{i=1}^{n} \alpha_i y_i &= 0. \end{aligned} \tag{2.32}$$

Si aplicamos la segunda condición de KKT, la condición complementaria, se tiene la siguiente igualdad

$$\alpha_i (1 - y_i (\mathbf{x}_i^T \beta + \beta_0)) = 0 \quad i = 1, \ldots, n. \tag{2.33}$$

Queremos construir el problema dual. Para ello, reescribimos (2.30) como

$$L(\beta, \beta_0, \alpha) = \frac{1}{2}\|\beta\|^2 - \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i^T \beta - \beta_0 \sum_{i=1}^{n} \alpha_i y_i + \sum_{i=1}^{n} \alpha_i. \tag{2.34}$$

A continuación, sustituimos las igualdades (2.32) en (2.34), lo que hace que el tercer término de la parte de la derecha se anule y se obtenga la función de Lagrange que depende únicamente de los multiplicadores de Lagrange,

$$L(\alpha) = \frac{1}{2}\left(\sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i\right)\left(\sum_{j=1}^{n} \alpha_j y_j \mathbf{x}_j\right) - \left(\sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i\right)\left(\sum_{j=1}^{n} \alpha_j y_j \mathbf{x}_j\right) + \sum_{i=1}^{n} \alpha_i,$$

$$L(\alpha) = -\frac{1}{2}\left(\sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i\right)\left(\sum_{j=1}^{n} \alpha_j y_j \mathbf{x}_j\right) + \sum_{i=1}^{n} \alpha_i.$$

Se obtiene así la función objetivo del problema dual,

$$L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j. \tag{2.35}$$

De esta manera, podemos definir el problema dual, que trata de encontrar un α^* que maximice la función (2.35) añadiendo las restricciones de no negatividad asociadas a los multiplicadores de Lagrange junto con la igualdades de (2.32), es decir

$$\begin{aligned} \text{máx} \quad & L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j, \\ \text{sujeto a:} \quad & \sum_{i=1}^{n} \alpha_i y_i = 0, \\ & \alpha_i \geq 0 \quad i = 1, \ldots, n. \end{aligned} \tag{2.36}$$

que es conocido también como *problema dual de Wolfe*.

Este problema dual de Wolfe puede reescribirse en notación matricial de la siguiente forma, donde se busca el α tal que

$$\begin{aligned} \text{máx} \quad & L_D(\alpha) = \mathbf{1}_n^T \alpha - \frac{1}{2} \alpha^T \mathbf{H} \alpha, \\ \text{sujeto a:} \quad & \alpha \geq 0,\ \alpha^T \mathbf{y} = 0. \end{aligned} \tag{2.37}$$

donde $\mathbf{y} = (y_1, \ldots, y_n)^T$ y $\mathbf{H} = (H_{ij})$ es una matriz cuadrada de tamaño $n \times n$ tal que $H_{ij} = y_i y_j (\mathbf{x}_i^T \mathbf{x}_j)$.

Se puede resolver el problema dual utilizando técnicas de programación cuadrática, las cuales comentaremos en posteriores capítulos. Además, en general, este problema suele ser más sencillo de resolver. Esto se debe a que para el problema dual, el número de variables que hay que encontrar es proporcional al tamaño de la muestra n, mientras que para el primal lo es a la dimensionalidad p. Esta propiedad es importante, pues implica que el coste computacional asociado a la resolución del problema dual es menor y por tanto más factible, incluso en problemas con dimensiones altas.

Una vez que se obtiene la solución del problema dual, α^*, para obtener la solución del problema primal a partir de esta es un proceso sencillo.

Primero basta con sustituir dicha solución α^* en (2.32) para obtener el vector de pesos óptimo β^*,

$$\beta^* = \sum_{i=1}^{n} \alpha_i^* y_i \mathbf{x}_i. \tag{2.38}$$

Usando la condición complementaria de KKT en (2.33), dado que los multiplicadores de Lagrange son no negativos, esto es, $\alpha_i \geq 0$. Si $\alpha_i > 0$, entonces el segundo factor de la parte de la izquierda tendrá que ser nulo,

$$y_i(\mathbf{x}_i^T \beta^* + \beta_0^*) = 1, \tag{2.39}$$

es decir, aquellas observaciones que tienen asociado un multiplicador $\alpha_i > 0$, satisfacen la igualdad en (2.17) y por tanto son los vectores soporte. En consecuencia, se puede afirmar que el hiperplano de separación óptimo se construye únicamente a partir de una combinación lineal de los vectores soporte, debido a que las demás observaciones del conjunto de entrenamiento tienen asociado un multiplicador nulo, $\alpha_j = 0$. Sea $\mathcal{SV} \subset \{1, \ldots, n\}$ el subconjunto de índices que identifican a los vectores soporte, y en consecuencia a los multiplicadores de Lagrange no nulos. Se tiene que el vector de pesos del hiperplano β^* dado por (2.38), viene dado por la suma únicamente de las observaciones que son vectores soporte, esto es

$$\beta^* = \sum_{i \in \mathcal{SV}} \alpha_i^* y_i \mathbf{x}_i. \tag{2.40}$$

En otras palabras, β^* solo depende de vectores soporte, $\{\mathbf{x}_i : i \in \mathcal{SV}\}$. En muchas aplicaciones, el número de vectores soporte puede ser muy pequeño en comparación con el número total de observaciones del conjunto de datos y, en cualquier caso, los vectores soporte proporcionan toda la información necesaria para determinar el *Hard-SVM*.

Queda por determinar el valor del parámetro β_0^* para obtener la solución del problema primal. Para ello, lo podemos estimar resolviendo (2.33) para uno de los vectores soporte. Es decir, podemos despejar β_0^* de (2.39),

$$\beta_0^* = y_{\text{sv}} - \mathbf{x}_{\text{sv}}^T \beta^*, \tag{2.41}$$

donde $(\mathbf{x}_{\text{sv}}, y_{\text{sv}})$ representa un punto soporte sv $\in \mathcal{SV}$, correspondiente al vector de variables predictoras junto con el valor de su clase, es decir, cualquier observación que tiene asociado un multiplicador de Lagrange α_i distinto de cero.

En general, en la práctica suele ser más robusto obtener β_0^* del promedio de todos los vectores soporte. Así la expresión (2.41) se puede sustituir por esta otra

$$\beta_0^* = \frac{1}{N_{\mathcal{SV}}} \sum_{j \in \mathcal{SV}} (y_j - \mathbf{x}_j^T \beta^*) \tag{2.42}$$

siendo $\mathcal{SV}$ el conjunto de vectores soporte y $N_{\mathcal{SV}}$ el cardinal de dicho conjunto.

Si además queremos obtener una expresión de β_0^* a partir de la solución obtenida en el problema dual, α^*, basta con sustituir (2.32) en (2.42).

De todo esto se sigue que el hiperplano de separación óptimo (*hard-SVM*) se puede escribir como

$$f^*(\mathbf{x}) = \beta^{*T}\mathbf{x} + \beta_0 = \sum_{i \in \mathcal{SV}} \alpha_i^* y_i \mathbf{x}^T \mathbf{x}_i + \beta_0^*. \tag{2.43}$$

Para resolver este *Hard-SVM* se puede observar de la ecuación (2.43) que únicamente los vectores soportes son relevantes en el cálculo, mientras que las demás observaciones que no son vectores soporte no juegan ningún papel en la resolución del problema de optimización. Esta descripción de la solución en términos de los vectores soporte, parece sugerir que el hiperplano de separación óptimo se centra más en los vectores más complicados de clasificar (por ser más "fronterizos") y es más robusto para los errores de especificación del modelo.

La regla de clasificación *Hard-SVM* viene dada por

$$Q(\mathbf{x}) = \text{sign}(f^*(\mathbf{x})). \tag{2.44}$$

Si $j \in \mathcal{SV}$, por (2.43) se cumple la siguiente igualdad

$$f^*(\mathbf{x}_j) = \sum_{i \in \mathcal{SV}} \alpha_i^* y_i y_j (\mathbf{x}_j^T \mathbf{x}_i) + y_j \beta_0^* \tag{2.45}$$

Por tanto, la norma al cuadrado del vector de pesos β^* del hiperplano de separación óptimo es

$$\begin{aligned} \|\beta^*\|^2 &= \sum_{i \in \mathcal{SV}} \sum_{j \in \mathcal{SV}} \alpha_i^* \alpha_j^* y_i y_j (\mathbf{x}_i^T \mathbf{x}_j) \\ &= \sum_{j \in \mathcal{SV}} \alpha_j^* \sum_{i \in \mathcal{SV}} \alpha_i^* y_i y_j (\mathbf{x}_i^T \mathbf{x}_j) \\ &= \sum_{j \in \mathcal{SV}} \alpha_j^* (1 - y_j \beta_0^*) \\ &= \sum_{j \in \mathcal{SV}} \alpha_j^* \end{aligned} \tag{2.46}$$

donde usamos (2.45) en la tercera linea y (2.32) en la última igualdad. De aquí podemos ver que el hiperplano de margen máximo tiene como margen máximo entre las dos clases $L = 2M = 2/\|\beta^*\|$, donde

$$\frac{1}{\|\beta^*\|} = \left(\sum_{j \in \mathcal{SV}} \alpha_j^* \right)^{-1/2}. \tag{2.47}$$

Como comentario final, resaltaremos que la definición del problema dual y la del hiperplano de separación óptimo dependen del producto escalar de los vectores soporte. Esto es una propiedad, que se utilizará más tarde, para calcular hiperplanos de separación óptimos en espacios de dimensiones mayores transformando las variables de entrada.

Capítulo 3

Caso no separable linealmente (*Soft-SVM*)

En la práctica, el problema planteado en el capítulo anterior no tiene demasiado interés, ya que en general, los conjuntos de datos que aparecen en la práctica habitual del aprendizaje supervisado no tienen todas las observaciones separadas linealmente. Es decir, las clases se solapan en el espacio de las características y no pueden ser separadas linealmente por un hiperplano.

En este capítulo, trataremos de extender el concepto tratado en el capítulo anterior de hiperplano de separación de margen óptimo, es decir, el *hard-SVM*, para conseguir un hiperplano que "separe" lo mejor posible las clases, usando lo que se conoce como *margen suave*. Para este tipo de problemas, Vapnik y Cortes [5] expusieron que es imprescindible permitir que algunas observaciones no cumplan las restricciones del margen formuladas en el problema primal (2.18) del caso de datos separables, es decir, suavizar el grado de separabilidad del conjunto de entrenamiento permitiendo que haya errores en la clasificación de algunas observaciones del conjunto de entrenamiento. Esta generalización se conoce como *soft-SVM*.

Dadas estas circunstancias, existen dos posibilidades para una observación de este tipo. Puede caer al otro lado del hiperplano, es decir en la clase incorrecta, o caer dentro del margen asociado a su clase, de acuerdo a la frontera definida desde el hiperplano de separación. Ambas observaciones se dice que son *no separables*, siendo la primera mal clasificada y la segunda clasificada de forma correcta. Esto lo podemos observar en la Figura 3.1, donde $\mathbf{x}_j$ pertenece al primer caso y $\mathbf{x}_k$ al segundo, respectivamente.

Desde el punto de vista de la formulación, una observación no separable es aquella que no cumple las restricciones del problema (2.13).

Sea un conjunto de observaciones de entrenamiento $\mathcal{C}$ no separables. La idea para abordar este problema en el que las clases se solapan es maximizar M, pero permitir que algunos puntos puedan quedar en el lado incorrecto del margen. Para ello, definimos las *variables de holgura*, $\{\xi_i : i = 1, \ldots, n\}$, que serán posteriormente utilizadas para cuantificar el número de observaciones no separables que se van a permitir. Un ejemplo de esto, podemos observarlo en la Figura 3.1

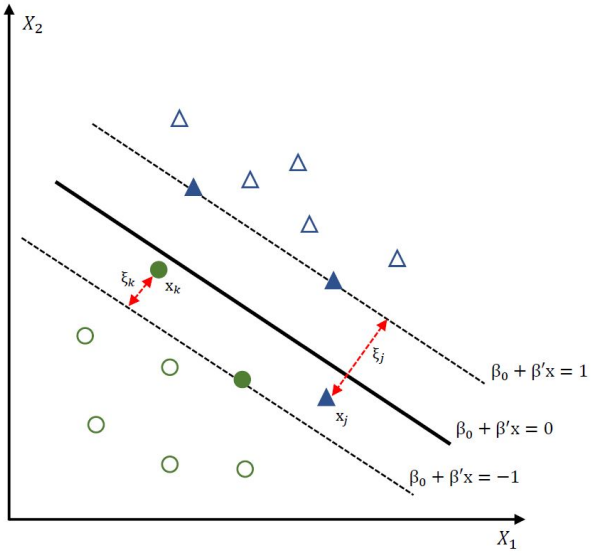

Figura 3.1: Ejemplos que están en el lado contrario del margen de seguridad junto a sus correspondientes variables de holgura ξ_i.

Hay dos formas para introducir estas variables de holgura en las restricciones del problema (2.13),

$$y_i \left(\mathbf{x}_i^T \beta + \beta_0\right) \geq M - \xi_i, \tag{3.1}$$

o

$$y_i \left(\mathbf{x}_i^T \beta + \beta_0\right) \geq M(1 - \xi_i), \tag{3.2}$$

con $\xi_i \geq 0$ para todo $i \in \{1, ..., n\}$ y $\sum_{i=1}^{n} \xi_i \leq K$, siendo K una constante a elegir.

En función de qué restricción de las dos tomemos llegaremos a distintas soluciones. La primera desigualdad (3.1), parece más natural, pues mide la superposición de la distancia real con respecto al margen, mientras que la segunda (3.2), mide la superposición de la distancia relativa, que cambia con el ancho del margen, M. Sin embargo, la primera da como resultado un problema de optimización no convexo, mientras que la segunda da uno convexo, así pues a partir de ahora utilizaremos la desigualdad (3.2), que conducirá a lo que se conoce como *soft-SVM*.

Se puede entender el valor de la variable de holgura, ξ_i, en la desigualdad (3.2), como la cantidad proporcional por la cual la predicción de la observación $\mathbf{x}_i$, $f(\mathbf{x}_i) = \mathbf{x}_i^T \beta + \beta_0$, está en el lado equivocado de su margen, es decir, la desviación de la separabilidad con respecto al margen de la clase correspondiente a dicha observación. Por lo tanto, al acotar la suma de estas variables de holgura, $\sum_{i=1}^{n} \xi_i$, limitamos la cantidad proporcional total por la cual las predicciones caen en el lado equivocado de su margen. En función del valor de la variable de holgura, se pueden clasificar las observaciones no separables, si la variable de holgura está entre 0 y 1 estará bien clasificado, mientras que una mala clasificación ocurre cuando $\xi_i > 1$, luego limitando el sumatorio anterior a K, limitamos el número total de observaciones de entrenamiento mal separadas a K, es decir, cuanto mayor sea el valor de esta constante, mayor será el número de observaciones no separables. Para eliminar este problema de llegar a varias soluciones, se fija de forma arbitraria, como hemos hecho

en el capítulo anterior en la igualdad (2.16), que $M\|\beta\| = 1$. En consecuencia, podemos reescribir la condición (3.2) como

$$y_i \left(\mathbf{x}_i^T \beta + \beta_0\right) \geq 1 - \xi_i, \tag{3.3}$$

con $\xi_i \geq 0$ para todo $i \in \{1, ..., n\}$ y $\sum_{i=1}^{n} \xi_i \leq K$.

Una vez que hemos suavizado las restricciones, según (3.3), podemos escribir el problema equivalente a (2.18), como

$$\begin{aligned} \min_{\beta,\beta_0} \quad & \frac{1}{2}\|\beta\|^2 \\ \text{sujeto a:} \quad & y_i \left(\mathbf{x}_i^T \beta + \beta_0\right) \geq 1 - \xi_i, \ i = 1, \ldots, n, \\ & \xi_i \geq 0, \ i = 1, \ldots, n, \\ & \sum_{i=1}^{n} \xi_i \leq K, \ \text{para } K \text{ constante.} \end{aligned} \tag{3.4}$$

Este problema de optimización es cuadrático con restricciones de desigualdad lineales, y por tanto es un problema de optimización convexo. Sin embargo, ya no basta con plantear como único objetivo maximizar el margen, es decir, minimizar la norma de los coeficientes β, pues lo lograríamos fácilmente clasificando erróneamente muchas observaciones. Para no tener este problema, tenemos que incluir de alguna manera en la función objetivo los errores de clasificación que comete el hiperplano de separación. Para ello, se define el siguiente problema de optimización

$$\begin{aligned} \min_{\beta,\beta_0} \quad & \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{n} \xi_i \\ \text{sujeto a:} \quad & y_i \left(\mathbf{x}_i^T \beta + \beta_0\right) \geq 1 - \xi_i, \ i = 1, \ldots, n, \\ & \xi_i \geq 0, \ i = 1, \ldots, n. \end{aligned} \tag{3.5}$$

donde C se denomina *parámetro de regularización* o parámetro de coste y reemplaza a la restricción de la constante K del problema (3.4). Este valor que se elige, permite controlar el grado en que influye el término del coste de observaciones no separables en la minimización de la norma de β, es decir, permite regular el compromiso entre el grado de sobreajuste del clasificador final y la proporción del número de observaciones mal clasificadas, la complejidad y el error de entrenamiento.

Elecciones de valores pequeños de C permitirán valores de ξ_i más grandes. Es decir estaríamos admitiendo una anchura de margen óptimo más grande, pero a costa de admitir más observaciones dentro del margen e, incluso, clasificándolas mal. Además, esto produce modelos más complejos, pues va a depender de un mayor número de vectores soporte. El caso extremo, es decir, cuando $C \to 0$, implicaría que todas las observaciones podrían estar mal situadas dentro del margen y el número de observaciones mal clasificadas sería máximo, pues las holguras ξ se podrían hacer arbitrariamente grandes.

Sin embargo, una elección de un valor muy grande de C fuerza valores de ξ_i muy pequeños y, en consecuencia, la anchura del margen óptimo será pequeño. El modelo será más simple, pero podría estar sobreajustado a los datos de entrenamiento. El caso de

datos separable se corresponde cuando estamos en el límite, es decir, cuando $C \to \infty$, pues estaríamos considerando que todas las observaciones son separables, esto es $\xi_i \to 0$. No existe en la actualidad ningún resultado que pruebe cual es el valor óptimo para C dado un conjunto de datos de entrenamiento.

El hiperplano obtenido para este caso en el que las observaciones no son linealmente separables y que es solución del problema (3.5), se conoce como *hiperplano de separación de margen suave*, o también como hemos dicho anteriormente, *soft-SVM*. Este problema de optimización también es convexo, por tanto para buscar la solución usaremos de nuevo los multiplicadores de Lagrange.

Como se ha explicado en el Capítulo 2, la función de Lagrange primal para el problema (3.5), por (2.22), es

$$L_p(\beta, \beta_0, \xi, \alpha, \mu) = \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i\left[y_i\left(x_i^T\beta + \beta_0\right) - (1-\xi_i)\right] - \sum_{i=1}^{n}\mu_i\xi_i, \quad (3.6)$$

que se quiere minimizar con respecto a β, β_0, ξ_i, siendo $\alpha_i, \mu_i, i = 1, \ldots, n$ los multiplicadores de Lagrange. En este caso aparecen dos familias de multiplicadores de Lagrange pues hay dos familias de restricciones en el problema (3.5). Además, los signos del tercer y cuarto sumando son negativos pues ambas restricciones son del tipo $g(x) \geq 0$ en lugar de $g(x) \leq 0$, como ocurre en el problema (2.19).

Aplicando las primeras condiciones de KKT, obtenemos

$$\frac{\partial L}{\partial \beta} \equiv \beta^* - \sum_{i=1}^{\pi}\alpha_i y_i \mathbf{x}_i = 0 \quad (3.7)$$

$$\frac{\partial L_p}{\partial \beta_0} \equiv \sum_{i=1}^{n}\alpha_i y_i = 0 \quad (3.8)$$

$$\frac{\partial L_p}{\partial \xi_i} = C - \alpha_i - \mu_i = 0,\ i = 1, \ldots, n \quad (3.9)$$

además de las restricciones de no negatividad, $\alpha_i, \mu_i, \xi_i \geq 0$ para todo i.

Las condiciones complementarias de KKT en este caso son

$$\alpha_i\left[y_i\left(\mathbf{x}_i^T\beta + \beta^*\right) - (1-\xi_i)\right] = 0,\ i = 1, \ldots, n, \quad (3.10)$$

$$\mu_i\xi_i = 0,\ i = 1, \ldots, n. \quad (3.11)$$

También se cumple la restricción del problema (3.4),

$$y_i\left(\mathbf{x}_i^T\beta + \beta^*\right) - (1-\xi_i) \geq 0,\ i = 1, \ldots, n. \quad (3.12)$$

La función objetivo de nuestro problema dual se puede obtener eliminando las variables primales de la función de Lagrange (3.6). Para ello reescribimos esa función como

$$L(\beta, \beta_0, \xi, \alpha, \mu) = \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i y_i \mathbf{x}_i^T\beta - \beta_0\sum_{i=1}^{n}\alpha_i y_i + (1-\xi_i)\sum_{i=1}^{n}\alpha_i - \sum_{i=1}^{n}\mu_i\xi_i, \quad (3.13)$$

A continuación, sustituimos en (3.13), la igualdad (3.7) y agrupamos,

$$L(\beta,\beta_0,\xi,\alpha,\mu) = \frac{1}{2}\left(\sum_{i=1}^{n}\alpha_i y_i \mathbf{x}_i\right)\left(\sum_{j=1}^{n}\alpha_j y_j \mathbf{x}_j\right) - \left(\sum_{i=1}^{n}\alpha_i y_i \mathbf{x}_i\right)\left(\sum_{j=1}^{n}\alpha_j y_j \mathbf{x}_j\right) + \\ + \sum_{i=1}^{n}\alpha_i + \sum_{i=1}^{n}(C-\alpha_i-\mu_i)\xi_i.$$

Usando ahora la igualdad (3.9), que hace que el cuarto término de la parte de la derecha se anule, se obtiene la función de Lagrange que depende únicamente de los multiplicadores de Lagrange α,

$$L(\alpha) = -\frac{1}{2}\left(\sum_{i=1}^{n}\alpha_i y_i \mathbf{x}_i\right)\left(\sum_{j=1}^{n}\alpha_j y_j \mathbf{x}_j\right) + \sum_{i=1}^{n}\alpha_i$$

Obtenemos así la función objetivo del problema dual,

$$L_D(\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T\mathbf{x}_j \tag{3.14}$$

Además, podemos establecer las siguientes restricciones para las variables del problema dual por las igualdades (3.8) y (3.9),

$$\sum_{i=1}^{n}\alpha_i y_i = 0, \tag{3.15}$$

$$C = \alpha_i + \mu_i,\ i = 1,\dots,n. \tag{3.16}$$

Por todo ello, tenemos que el problema de optimización dual a maximizar es

$$\begin{aligned} \text{máx} \quad & L_D(\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T\mathbf{x}_j \\ \text{sujeto a:} \quad & \sum_{i=1}^{n}\alpha_i y_i = 0 \\ & 0 \le \alpha_i \le C \quad i = 1,\dots,n. \end{aligned} \tag{3.17}$$

Se puede observar que a partir de este problema se tiene una clasificación más completa de los vectores soporte, ya que los multiplicadores de Lagrange tienen una cota superior.

Todas estas ecuaciones caracterizan de manera única la solución al problema primal y dual. Para establecer las relaciones entre las variables del problema primal (β,β_0,ξ) con las variables del problema dual, α, usamos la primera igualdad (2.31),

$$\beta^* = \sum_{i=1}^{n}\alpha_i^* y_i \mathbf{x}_i. \tag{3.18}$$

Como sucedía en el capítulo anterior, a partir de la solución del problema dual podemos obtener la solución del problema primal y por tanto expresar el *soft-SVM* en términos de esta solución α^*. Para ello, sustituimos (3.18) en (2.6) y obtenemos la ecuación del *soft-SVM*

$$f(\mathbf{x}) = \sum_{i=1}^{n}\alpha_i^* y_i \mathbf{x}^T\mathbf{x}_i + \beta_0^*. \tag{3.19}$$

Veamos algunos resultados interesantes antes de obtener la expresión del valor de β_0 de una manera similar a como se hizo en el capítulo anterior.
Existen 3 tipos de observaciones de entrenamiento. Supongamos que $\mathbf{x}_i$ una observación de entrenamiento.

Si cumple que su multiplicador de Lagrange es $\alpha_i^* = 0$, entonces es una observación separable. Esto se debe a que si se toma $\alpha_i = 0$, por la ecuación (3.16), se tiene que cumplir $\mu_i = C$, en consecuencia, $\xi_i = 0$ por (3.11), luego esa observación es perfectamente separable. Por la definición vista en el capítulo anterior, este tipo de observaciones no son vectores soporte.

Cuando $\alpha_i^* > 0$, por la restricción (3.10), ha de cumplirse la igualdad en (3.3), es decir,

$$y_i \left(\mathbf{x}_i^T \beta + \beta^* \right) - (1 - \xi_i) = 0,$$

o de manera equivalente

$$y_i f(\mathbf{x}_i) = 1 - \xi_i.$$

En este caso, estas observaciones si que serán vectores soporte, pues siempre van a cumplir la definición ya que $\xi_i \geq 0$, luego van a intervenir en la construcción del hiperplano de separación óptimo. Dentro de estos, hay dos casos.

Consideremos el caso $\alpha_i^* = C$. Se puede decir, que toda observación que cumple esta igualdad va a ser no separable, que como hemos comentado anteriormente, está caracterizado por tener un valor asociado $\xi_i > 0$. Para demostrarlo, suponiendo la hipótesis de que $\alpha_i = C$, por (3.16), se tiene que $\mu_i = 0$ y por (3.11) vemos que $\xi_i > 0$. En este caso, estos vectores soporte, se conocen como *vectores soporte acotados "ligados"*, y puede haber dos posibilidades; que la observación $\mathbf{x}_i$ esté bien clasificada aunque sea no separable, es decir $y_i f(\mathbf{x}_i) \geq 0$, luego $\xi_i = 1 - |f(\mathbf{x}_i)|$, y por tanto $0 < \xi_i \leq 1$, o que la observación no esté bien clasificada, es decir, $y_i f(\mathbf{x}_i) < 0$, luego $\xi_i = 1 + |f(\mathbf{x}_i)|$, y por tanto $\xi_i > 1$.

Por último, consideremos el caso $0 < \alpha_i^* < C$. Por la restricción (3.16), se deduce que $\mu_i \neq 0$ y por la igualdad (3.11) ha de cumplirse $\xi_i = 0$, y en consecuencia, la observación será separable. Estos vectores soporte se conocen como *vectores soporte "normales"*, pues por la restricción (3.11) y como $\xi_i = 0$, se cumple que

$$1 - y_i f(\mathbf{x}_i) = 0,$$

lo que significa que estas observaciones son las que pertenecen a la frontera del margen, y por eso se denominan así, "normales". Estos son aquellos con los que se calcula el valor de β_0, es decir,

$$\beta_0^* = y_i - \mathbf{x}_i^T \beta^*, \quad \text{para todo } i : \ 0 < \alpha_i < C. \tag{3.20}$$

Existe una diferencia con el caso lineal, ya que para el cálculo de β_0^* existe una restricción más fuerte, pues hemos visto que solo intervienen aquellos vectores soporte tales que $0 < \alpha_i < C$. Usaremos un promedio de todas las soluciones de (3.20) para promover la estabilidad numérica, como vimos en el capítulo anterior, es decir

$$\beta_0^* = \frac{1}{N_{\mathcal{SV}'}} \sum_{j \in \mathcal{SV}'} (y_i - \mathbf{x}_j^T \beta^*)$$

donde $\mathcal{SV}' = \{\mathbf{x}_i : \ 0 < \alpha_i < C\}$ y $N_{\mathcal{SV}'}$ es el cardinal de dicho conjunto.

El valor de β_0^* es también posible expresarlo en términos de las variables duales, usando (3.18), de la forma:

$$\beta_0^* = y_i - \sum_{j=1}^{n} \alpha_i y_i \mathbf{x}_i^T \mathbf{x}_j \quad \forall i : \, 0 < \alpha_i < C,$$

donde los coeficientes $\alpha_i^*, \; i = 1, \ldots, n$ son las soluciones correspondientes al problema dual.

Además por las restricciones del problema, se puede reescribir la ecuación (3.19) de forma equivalente, donde solo intervienen los vectores soporte que además están acotados, es decir

$$f^*(\mathbf{x}) = \sum_{i \in \mathcal{SV}'} \alpha_i^* y_i \mathbf{x}^T \mathbf{x}_i + \beta_0^*. \tag{3.21}$$

Una vez que se tiene las soluciones β^* y β_0^* y por tanto la función del hiperplano, f^*, se puede escribir la función de decisión, es decir, *la regla de clasificación Soft-SVM*, como

$$Q(\mathbf{x}) = \text{sign}\left(f^*(\mathbf{x})\right). \tag{3.22}$$

El parámetro de ajuste de este procedimiento, como se ha comentado anteriormente, es el parámetro de costo C. Más adelante en el Capítulo 6 veremos en un ejemplo qué sucede cuando tomamos un valor u otro de este parámetro.

El hecho de que el SVM sea un clasificador basado únicamente en un subconjunto potencialmente pequeño de las observaciones de entrenamiento, en los vectores soporte, significa que es bastante robusto para el comportamiento de las observaciones que están lejos del hiperplano. Esta propiedad no la tienen otros clasificadores y hace que este no modifique mucho el hiperplano de separación cuando existen posibles valores atípicos, *outliers*.

El problema del *Soft-SVM* definido en (3.5) presenta otro planteamiento para poder resolverlo a través de una formulación de minimización de una "pérdida empírica" combinada con una "penalización por complejidad". Este tipo de formulación es típica de muchos problemas de aprendizaje automático.
Para ello vamos a introducir varias definiciones que nos serán útiles para entenderlo.

Dado un dominio $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, sea $\mathcal{H}$ el conjunto de posibles clasificadores (en nuestro caso, clasificadores dependientes del conjunto de posibles hiperplanos), donde $h : \mathcal{Z} \to \mathbb{R}$. Es decir, toma como entrada un objeto del dominio $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ que será un par $(\mathbf{x}, y)$, donde $X = \mathbf{x} \in \mathcal{X}$ será el resultado de un vector aleatorio X representando las variables predictoras (en nuestro caso $\mathcal{X} = \mathbb{R}^p$) e $Y = y \in \mathcal{Y}$ es el resultado de la variable aleatoria Y (en nuestro caso codificará las dos clases a predecir como $\mathcal{Y} = \{-1, 1, \}$).

Sea ℓ una función tal que $\ell : \mathcal{H} \times \mathcal{Z} \to \mathbb{R}_+$. A estas funciones se les conoce como *funciones de pérdida*. Definimos ahora el *riesgo esperado* $R(h)$ como la pérdida esperada de un clasificador $h \in \mathcal{H}$, con respecto a una distribución conjunta del par $Z = (X, Y)$ definida sobre el dominio $\mathcal{Z}$, esto es

$$R(h) = \mathbb{E}_Z[\ell(h, Z)].$$

De manera similar, definimos el *riesgo empírico* como la pérdida empírica calculada sobre el resultado de una muestra aleatoria $Z_1 = z_1 = (\mathbf{x}_1, y_1)$,..., $Z_n = z_n = (\mathbf{x}_n, y_n)$ que

constituye nuestro conjunto de entrenamiento $\mathcal{C} = \{z_1, \ldots, z_n\} \in \mathcal{Z}^n$. Es decir,

$$R_{\mathcal{C}}(h) = \frac{1}{n} \sum_{i=1}^{n} \ell(h, z_i).$$

Podemos considerar ahora el problema de minimizar la función de pérdida empírica sobre $\mathcal{H}$:

$$h^* = \underset{h \in \mathcal{H}}{\operatorname{argmin}} R_{\mathcal{C}}(h). \tag{3.23}$$

En el SVM podemos considerar que las reglas de clasificación $\mathcal{H}$ están asociadas a los distintos hiperplanos separables. Podemos evitar el β_0 e incluirlo también en el vector β que define los hiperplanos (se puede incluir una variable "predictora" tomando un valor constante de 1 al definir $\mathbf{x}$). Así, para un conjunto de entrenamiento $\mathcal{C} = \{z_1, \ldots, z_n\}$ y un determinado β, se define (con cierto abuso de notación)

$$R_{\mathcal{C}}(\beta) = \frac{1}{n} \sum_{i=1}^{n} \ell\left(\beta, z_i\right),$$

para la función de perdida ℓ.

Lema 3.1. *Si ℓ es una función de pérdida convexa y el conjunto de clases de hipótesis $\mathcal{H}$ es convexo, entonces el problema de minimizar la función de pérdida empírica sobre $\mathcal{H}$ es un problema de optimización convexo, esto es, un problema de minimización de una función convexa sobre un conjunto convexo.*

Demostración. La definición de perdida empírica y el Lema 2.10 implica que la función $R_{\mathcal{C}}(h)$ es convexa. □

Como hemos comentado anteriormente, los problemas de optimización convexos pueden resolverse de manera muy eficiente. Sin embargo, en muchas ocasiones, las funciones de pérdidas no son convexas. Como ejemplo de ello, consideremos la clase de subespacios con respecto a la función de pérdida 0-1. Esto función se define de la siguiente forma para $z = (\mathbf{x}, y)$ como

$$\ell^{0-1}(\beta, z) = \ell^{0-1}(\beta, (\mathbf{x}, y)) = \mathbb{1}_{[y \neq \operatorname{sign}(\langle \beta, \mathbf{x} \rangle)]} = \mathbb{1}_{[y\langle \beta, \mathbf{x} \rangle \leq 0]}.$$

Esta función de pérdida no es convexa como podemos ver en la Figura 3.2. Para evitar la dificultad de este planteamiento, un enfoque popular es acotar superiormente estas funciones de pérdida no convexas por funciones de pérdida sustitutas que sean convexas y que acoten superiormente a la original.

Estamos en condiciones de definir la *función de pérdida de Hinge* como sigue

$$\ell^{hinge}(\beta, z) = \ell^{hinge}(\beta, (\mathbf{x}, y)) = [1 - \langle \beta, \mathbf{x} \rangle y]_+ = \text{máx}\{0, 1 - \langle \beta, \mathbf{x} \rangle y\}. \tag{3.24}$$

Para todos los pares (β, z) se cumple la desigualdad

$$\ell^{0-1}(\beta, z) \leq \ell^{hinge}(\beta, z).$$

Además, la convexidad de la función de pérdida de Hinge se tiene por el Lema (2.10). Todo ello se puede observar en la Figura 3.2.

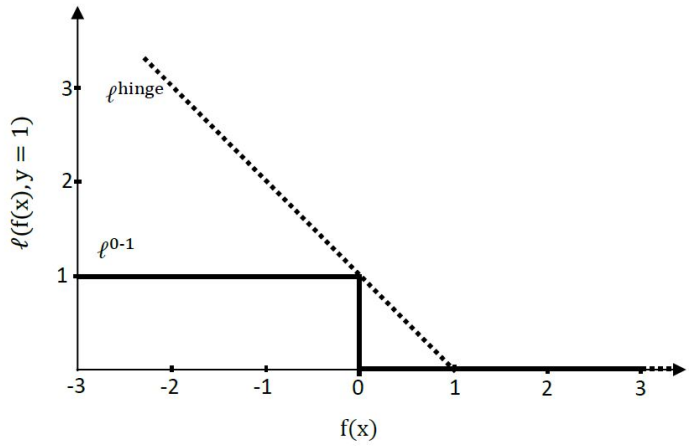

Figura 3.2: Funciones de pérdida de Hinge y 0-1 (Cota superior una de la otra).

Se puede reescribir el problema (3.5) como

$$\min_{\beta,\beta_0} \frac{1}{n}\sum_{i=1}^{n} \ell^{hinge}(\beta,(\mathbf{x}_i,y_i)) + \frac{\lambda}{2}\|\beta\|^2. \tag{3.25}$$

Veamos, para ello, que ambos problemas son equivalentes si volvemos a incluir β_0 en la notación y en nuestras expresiones. Para ello, fijemos β, β_0 y un i cualquiera, y consideremos la minimización de la ecuación (3.5) sobre ξ. Como ξ_i tiene que ser no negativo, la mejor asignación para ξ_i sería 0 si $y_i\left(\mathbf{x}_i^T\beta+\beta_0\right) \geq 1$ y de lo contrario $1 - y_i\left(\mathbf{x}_i^T\beta+\beta_0\right)$. En otras palabras,

$$\xi_i = \ell^{hinge}(\beta,(\mathbf{x}_i,y_i)) = \left[1-\left(\mathbf{x}_i^T\beta+\beta_0\right)y_i\right]_+$$

para todo i. Además el parámetro λ sirve para evitar el sobreajuste actuando como $1/C$, y la igualdad se cumple. Observemos que $\max\{0, 1-(\mathbf{x}_i^T\beta+\beta_0)y_i\}$ es 0 si $\mathbf{x}_i$ está en el lado correcto del margen y se cumple que $\xi_i \geq 0$. Si tenemos un margen grande $\|\beta\|$, es decir menor $1/\|\beta\|$, reducimos la pérdida empírica

$$\frac{1}{n}\sum_{i=1}^{n} \ell^{hinge}(\beta,(\mathbf{x}_i,y_i)),$$

aunque también buscamos un amplio margen de separación que, por otro lado, se consigue con un $\|\beta\|$ pequeño. La constante λ permite equilibrar estos dos objetivos contrapuestos. Por tanto, esta función objetivo (3.25) penaliza tanto por errores empíricos grandes como por normas $\|\beta\|$ elevadas.

Una vez hemos planteado el problema de *Soft*-SVM mediante la optimización de las dos tipos de problemas en (3.5) y (3.25), en los próximos capítulos veremos que la optimización de esta función de pérdida regularizada se puede realizar utilizando métodos de optimización convexa, como el descenso de gradiente estocástico que veremos en detalle en el Capítulo 5.

Capítulo 4

Uso de *Kernels*

Tanto el hiperplano de separación máximo (*Hard*-SVM), como el clasificador de vectores soporte (*Soft*-SVM), explicado en los capítulos anteriores son un enfoque natural para la clasificación en el entorno de dos clases, clasificación binaria, si la frontera entre las dos clases es lineal, o son cuasi-separables linealmente. Para el cálculo de estos hiperplanos, es decir, para la búsqueda de los parámetros, se ha visto que desde el enfoque del problema de optimización dual se puede resolver de forma más sencilla, siendo independiente de la dimensión del problema.

Sin embargo, en la práctica muchos de los conjuntos de datos que se quieren clasificar tienen observaciones que no son linealmente separables. Estos métodos anteriores no sirven para clasificar dichos conjuntos de datos. Como con otros métodos lineales de aprendizaje automático, se puede hacer que el procedimiento sea más flexible ampliando el espacio de características, utilizando transformaciones como polinomios o splines. Generalmente las fronteras lineales en el espacio ampliado logran una mejor separación de la clase de entrenamiento y se traducen en límites no lineales en el espacio original. En este capítulo, trataremos de abordar precisamente eso, como adaptar dichos resultados a este último caso. Para ello, veremos cómo utilizar funciones base no lineales de forma que se puedan definir espacios transformados de manera eficiente con una alta dimensionalidad además de cómo encontrar en ellos el hiperplano de separación óptimo. El éxito de este enfoque puede atribuirse tanto a su flexibilidad para acomodar el conocimiento previo específico del dominio como a tener un conjunto bien desarrollado de algoritmos de implementación eficientes.

La clave para construir este nuevo clasificador, un SVM no lineal, se encuentra en ver que los vectores de variables predictoras se definen en el problema de optimización dual a través de los productos internos $\mathbf{x}_i^T \mathbf{x}_j = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$ con $i, j = 1, \ldots, n$. Para ello, se recurrirá a una técnica que consiste en la transformación mediante una función no lineal del espacio original a un espacio dotado de un producto escalar, que denominaremos función *kernel*. Se conoce como *espacio de entradas* al espacio original de las observaciones, X, y *espacio de características* el espacio transformado, $\mathcal{F}$, que en general, tiene dimensiones grandes y se define a partir de un conjunto de funciones base no lineales.

Sea $h : X \to \mathcal{F}$ una función de transformación que hace corresponder a cada vector de entrada de variables predictoras $\mathbf{x}$ con un punto en el espacio de características $\mathcal{F}$. Sean además, $h_m(\mathbf{x})$, $m = 1, \ldots, M$ las funciones base no lineales que definen este espacio de características $\mathcal{F}$. El procedimiento a seguir es el mismo que hemos explica-

do en los capítulos anteriores. Se ajustará el clasificador de vectores soporte usando las características de entrada $h(\mathbf{x}_i) = (h_1(\mathbf{x}_i), h_2(\mathbf{x}_i), \ldots, h_M(\mathbf{x}_i))$, con $i = 1, \ldots, n$, lo que produce que la función no lineal sea $f(\mathbf{x}) = h(\mathbf{x})^T\beta^* + \beta_0^*$ y el clasificador sea la función $G(\mathbf{x}) = \text{signo}[f(\mathbf{x})]$.

La idea es construir un hiperplano de separación lineal en este nuevo espacio, donde la frontera de decisión lineal obtenida en este espacio de características, se transformará en una frontera de decisión no lineal en el espacio original. En la Figura 4.1 podemos ver un ejemplo gráfico de una función de transformación h.

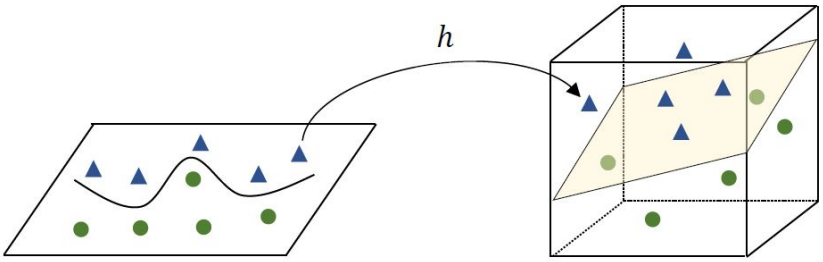

Figura 4.1: Ejemplo gráfico de una función de transformación h

En este contexto, cómo hemos dicho, la función de decisión en el espacio de características viene dada por

$$f(\mathbf{x}) = (\beta_1 h_1(\mathbf{x}) + \ldots + \beta_M h_M(\mathbf{x})) = h(\mathbf{x})^T\beta, \tag{4.1}$$

donde podemos observar que no aparece el término β_0, ya que se puede prescindir de él pues éste puede ser incluido al representarlo en la base de funciones de transformación de la función constante $h_1(\mathbf{x}) = 1$ y aumentando en uno la dimensión del vector β, esto es $\beta \in \mathbb{R}^{p+1}$.

Podemos reescribir la función de decisión en su forma dual transformando de manera conveniente la expresión (4.1) como

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i h(\mathbf{x})^T h(\mathbf{x}_i) = \sum_{i=1}^{n} \alpha_i y_i \langle h(\mathbf{x}), h(\mathbf{x}_i)\rangle. \tag{4.2}$$

Veremos a continuación, que el producto escalar $h(\mathbf{x})^T h(\mathbf{x}_i)$, que se representa también por $\langle h(\mathbf{x}), h(\mathbf{x}_i)\rangle$ se calculará a partir de lo que se denomina función kernel.

Para ello, comencemos viendo que es una función kernel.

Definición 4.1. Una *función kernel* es una función $K : X \times X \to \mathbb{R}$ que a cada par de elementos del espacio original de entrada X, le asigna un valor real que se corresponde con el producto escalar de las imágenes de dichos elementos en el espacio de características $\mathcal{F}$, es decir,

$$K(\mathbf{x}, \mathbf{x}') = \langle h(\mathbf{x}), h(\mathbf{x}')\rangle = (h_1(\mathbf{x})h_1(\mathbf{x}') + \ldots + h_M(\mathbf{x})h_M(\mathbf{x}')) \tag{4.3}$$

donde $h : X \to \mathcal{F}$ envía de X a un (producto interno) espacio de características $\mathcal{F}$.

En el ámbito de los *support vector machines*, un *kernel* se refiere a una función matemática que se utiliza para medir la similitud entre dos observaciones en un espacio de características. La *teoría del kernel* es una herramienta importante en el aprendizaje automático que se utiliza para clasificación, regresión y otras tareas de aprendizaje supervisado y no supervisado. Esta teoría, como ya hemos comentado, se basa en la idea de que la elección de una función kernel adecuada que permite transformar los datos de entrada a un espacio de características de alta dimensión, donde pueden ser más fácilmente separables.

Por otro lado, existe un desarrollo más formal que en este trabajo de fin de máster no consideramos, conocido como teoría del operador integral, que es una rama de las matemáticas que estudia las propiedades de los operadores integrales. Estos operadores son funciones que actúan sobre un espacio vectorial para producir otro espacio vectorial y se pueden expresar como integrales en un cierto espacio de funciones. Los operadores integrales son importantes en muchas áreas de las matemáticas aplicadas, como la teoría de la aproximación, la teoría de las ecuaciones diferenciales, la estadística y el aprendizaje automático.

La teoría del kernel y esta teoría del operador integral están estrechamente relacionadas. La conexión entre ambas es que los kernels son una clase especial de operadores integrales. En particular, el kernel asociado a una función de transformación de características se puede ver como el núcleo de un operador integral que transforma las funciones en el espacio de características. De esta manera, la teoría del kernel utiliza la teoría del operador integral para establecer propiedades teóricas de los métodos de aprendizaje automático basados en kernels, como veremos más adelante.

Una consecuencia importante de la representación dual es que la dimensión del espacio de características no afecta a los cálculos, como este no se representa explícitamente por los vectores de características, la cantidad de operaciones requeridas para calcular el producto interno mediante la evaluación de la función kernel no es necesariamente proporcional a la cantidad de dichas características. El uso de los kernels hace posible enviar los datos implícitamente en un espacio de características y entrenar una función lineal en dicho espacio, potencialmente eludiendo los problemas computacionales relacionados en la evaluación del espacio de características. La única información utilizada sobre las observaciones de entrenamiento en el espacio de características es su *matriz de Gram* o *matriz kernel*, que se define como la matriz $\mathbf{K} = K_{ij}$ de dimensiones $n \times n$, donde $K_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$. Una vez que se define a la matriz $\mathbf{K}$, la función se puede calcular evaluando al menos n veces la matriz $\mathbf{K}$ de la siguiente manera:

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}). \tag{4.4}$$

Utilizando esta función kernel, no hace falta calcular de manera explicita el transformación de la función $h : X \to \mathcal{F}$. Algunas propiedades que estas funciones tienen que cumplir son que han de ser simétricas, es decir,

$$K(\mathbf{x}, \mathbf{x}') = \langle h(\mathbf{x}), h(\mathbf{x}') \rangle = \langle h(\mathbf{x}'), h(\mathbf{x}) \rangle = K(\mathbf{x}', \mathbf{x}),$$

y tienen que satisfacer la siguiente desigualdad que se obtiene a partir de la desigualdad de Cauchy-Schwarz,

$$K(\mathbf{x}, \mathbf{x}') = \langle h(\mathbf{x}), h(\mathbf{x}') \rangle^2 \leq \|h(\mathbf{x})\|^2 \|h(\mathbf{x}')\|^2 = \langle h(\mathbf{x}), h(\mathbf{x}) \rangle \langle h(\mathbf{x}'), h(\mathbf{x}') \rangle = K(\mathbf{x}, \mathbf{x}) K(\mathbf{x}', \mathbf{x}')$$

Sin embargo, estas condiciones no son suficientes para garantizar la existencia del espacio de características. A continuación, introducimos una proposición que nos permite obtener aquellas condiciones que tiene que cumplir una función K para que sea un kernel que induzca un espacio de características.

Proposición 4.2. *Sea X un espacio de entradas finito tal que $K(\mathbf{x}, \mathbf{x}')$ es una función simétrica en X. Entonces $K(\mathbf{x}, \mathbf{x}')$ es una función kernel, si y solo si la matriz de Gram $K_{i,j} = (K(\mathbf{x}_i, \mathbf{x}_j))_{i,j=1}^n$ es semidefinida positiva, es decir, tiene autovalores no negativos.*

Demostración. Es trivial ver que si K es una función *kernel*, por definición la matriz de Gram es semidefinida positiva, pues que sea semidefinida positiva quiere decir que se cumple $\sum_{i=1}^n \sum_{j=1}^n c_i c_j K(\mathbf{x}_i, \mathbf{x}_j) \geq 0$ para cualesquiera conjuntos $\mathbf{x}_1, \ldots, \mathbf{x}_n \in X$ y $c_1, \ldots, c_n \in \mathbb{R}$, con $n > 0$.
Para demostrar la otra implicación, definimos el espacio de funciones sobre X como $\mathbb{R}^X = \{f : X \to \mathbb{R}\}$. Para cada $\mathbf{x} \in X$, sea $h(\mathbf{x})$ la función $\mathbf{x} \mapsto K(\cdot, \mathbf{x})$. Definimos el espacio de vectores cogiendo todas las posibles combinaciones de elementos del tipo $K(\cdot, \mathbf{x})$. Definimos un producto interno en este espacio de vectores como

$$\left\langle \sum_i \alpha_i K(\cdot, \mathbf{x}_i), \sum_j \beta_j K(\cdot, \mathbf{x}'_j) \right\rangle = \sum_{i,j} \alpha_i \beta_j K(\mathbf{x}_i, \mathbf{x}'_j),$$

Este producto interno es válido pues es simétrico, ya que K es simétrica, es linear y además es definido positivo, ya que $K(\mathbf{x}, \mathbf{x}) \geq 0$ cumpliéndose la igualdad cuando $h(\mathbf{x})$ es la función cero. Claramente,

$$\langle h(\mathbf{x}), h(\mathbf{x}') \rangle = \langle K(\cdot, \mathbf{x}), K(\cdot, \mathbf{x}') \rangle = K(\mathbf{x}, \mathbf{x}'),$$

lo que concluye la prueba. □

En los espacios de Hilbert también se pueden definir funciones kernel. Un *espacio de Hilbert* es una generalización del concepto de espacio euclídeo y, formalmente, se define como un espacio de producto interior que es completo con respecto a la norma vectorial definida por el producto interno, donde completo en este contexto significa que cualquier sucesión de Cauchy de elementos del espacio converge a un elemento en el espacio, en el sentido que la norma de las diferencias tiende a cero. La "teoría de los espacios de Hilbert", muestra que las funciones kernel se corresponden con un producto escalar y estas inducen en el espacio transformado un espacio lineal de mayor dimensión que el espacio original, que puede ser infinita. Surge la pregunta de cómo transformar las observaciones de entrada, de dimensión finita, en otro espacio de dimensión posiblemente infinita. El siguiente teorema responde a esta pregunta.

Teorema 4.3 (de Aronszajn)**.** *Para cualquier función $K : X \times X \to \mathbb{R}$ que sea simétrica y semidefinida positiva, existe un espacio de Hilbert y una función $h : X \to \mathcal{F}$ tal que*

$$K(\mathbf{x}, \mathbf{x}') = \langle h(\mathbf{x}), h(\mathbf{x}') \rangle, \text{ para todo } \mathbf{x}, \mathbf{x}' \in X. \tag{4.5}$$

Como consecuencia de este teorema, en 1992 Boser, Guyon y Vapnik [6] demostraron que para construir una función kernel, basta con definir una función que cumpla las dos condiciones del teorema, ha de ser simétrica y semidefinida positiva, no hace falta hacerlo

a partir de un conjunto de funciones base $h(\mathbf{x}) = (h_1(\mathbf{x}), \ldots, h_M(\mathbf{x}))$. Así, la función kernel representa el producto escalar $\langle h(\mathbf{x}), h(\mathbf{x}')\rangle$ que induce a un espacio de dimensión más alta. Esto permite evaluar una función sin conocer dicho conjunto de funciones base, basta con evaluar dicha función kernel sin tener que calcular explícitamente el producto escalar correspondiente. Más adelante lo veremos con unos ejemplos.

Todo esto nos permite inducir cualquier algoritmo lineal en un espacio de Hilbert, donde en el espacio original su versión es no lineal, siendo la transformación utilizada para ello h no lineal. Esto es lo que se conoce como el *truco de los kernels*, en inglés, "*Kernel trick*".

Existe una serie de funciones kernel que son las más utilizadas y entre las que se pueden destacar las siguientes.

Kernel lineal:

$$K_L(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}' \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j}, \tag{4.6}$$

que se corresponde con el núcleo utilizado para el *Hard-SVM*, y se denomina así pues este clasificador es lineal en el espacio de las características. Este kernel esencialmente cuantifica la similitud de un par de observaciones utilizando la correlación de Pearson.

Kernel polinómico de grado-d:

$$K_d(\mathbf{x}, \mathbf{x}') = (\tau + \gamma \langle \mathbf{x}, \mathbf{x}' \rangle)^d = (\tau + \gamma \sum_{j=1}^{p} x_{ij} x_{i'j})^d \quad (\gamma > 0). \tag{4.7}$$

Usando este núcleo con $d > 1$, en lugar de usar el kernel lineal (4.6) en el algoritmo del *Hard-SVM*, se llega a una frontera de decisión mucho más flexible. Esencialmente equivale a ajustar un clasificador de vectores de soporte en un espacio de mayor dimensión que involucra polinomios de grado d, en lugar de en el espacio de características original.

Kernel radial:

$$K_R(\mathbf{x}, \mathbf{x}') = \exp(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2) = \exp(-\gamma \langle (\mathbf{x} - \mathbf{x}'), (\mathbf{x} - \mathbf{x}') \rangle) = \exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2), \tag{4.8}$$

donde γ es una constante positiva. Más adelante veremos una proposición que nos demuestra porqué esta función es también un kernel.
Para ver como funciona el kernel radial (4.8), supongamos que tenemos una observación de *test*, $\tilde{\mathbf{x}} = (\tilde{x}_1, \ldots, \tilde{x}_p)$ que está muy alejada de la observación de entrenamiento $\mathbf{x}_i$ en términos de la distancia Euclidea, por tanto $\sum_{j=1}^{p} (\tilde{x}_j - x_{ij})^2$ es grande, y por tanto $\exp(-\gamma \sum_{j=1}^{p} (\tilde{x}_j - x_{ij})^2)$ será muy pequeño. Esto significa que la observación $\mathbf{x}_i$ no juega demasiado papel en la función de decisión $f(\tilde{\mathbf{x}})$. Recordemos que la etiqueta de clase predicha para el ejemplo de prueba $\tilde{\mathbf{x}}$ se basa en el signo de $f(\tilde{\mathbf{x}})$. En otras palabras, los ejemplos de entrenamiento que están lejos de $\tilde{\mathbf{x}}$ no tienen ningún papel en la etiqueta de clase predicha para $\tilde{\mathbf{x}}$. Esto significa que el núcleo radial tiene un comportamiento muy local, en el sentido de que solo los ejemplos de entrenamiento cercanos tienen un efecto en la etiqueta de clase de una observación del conjunto de test, lo que favorece, en algunas

ocasiones, el "sobreajuste". Veamos cual es la distancia para este kernel radial, es decir, esta es un producto escalar en un espacio transformado de dimensión finita. Si tomamos $\gamma = 1/2\sigma^2$, se tiene que $K(\mathbf{x}, \mathbf{x}') = h(\mathbf{x})^T h(\mathbf{x}') = \exp\left(-\|\mathbf{x} - \mathbf{x}'\|^2/2\sigma^2\right)$, y la distancia por tanto entre $h(\mathbf{x})$ y $h(\mathbf{x}')$ es

$$d(h(\mathbf{x}), h(\mathbf{x}')) = \sqrt{\|h(\mathbf{x}) - h(\mathbf{x}')\|^2} = \sqrt{2\left(1 - \exp\left(-\|\mathbf{x} - \mathbf{x}'\|^2 2\sigma^2\right)\right)} = \sqrt{2(1 - K(\mathbf{x}, \mathbf{x}'))}.$$

Se puede ver que cuando estamos en el caso unidimensional, con $\sigma^2 = 1$ que la distancia varía de manera exponencial. Para el caso de 2D, si tomamos valores pequeños del parámetro σ^2, la distancia estará muy localizada y por tanto todos los puntos de fuera de un radio pequeño están igualmente lejos. Sin embargo si tomamos valores un poco más grandes de σ^2 tendremos una distancia más global, equivalente a un kernel lineal.

Kernel sigmoidal:

$$K_S(\mathbf{x}, \mathbf{x}') = \tanh(\gamma\langle \mathbf{x}, \mathbf{x}'\rangle + \tau) = \tanh\left(\gamma \sum_{j=1}^{p} x_{ij} x_{i'j} + \tau\right) \tag{4.9}$$

A estos parámetros γ, τ y d se les denomina parámetros del kernel.

A continuación, vamos a ver un ejemplo para una función kernel polinomial.

Ejemplo 4.4. Consideremos el espacio de entradas X y además consideremos también dos variables de entrada x_1 y x_2, y tomemos el kernel polinomial de grado 2. Entonces se define el kernel de la siguiente manera

$$\begin{aligned} K(\mathbf{x}, \mathbf{x}') &= (1 + \langle \mathbf{x}, \mathbf{x}'\rangle)^2 = (1 + x_1 x_1' + x_2 x_2')^2 = \\ &= 1 + 2x_1 x_1' + 2x_2 x_2' + (x_1 x_1')^2 + (x_2 x_2')^2 + 2x_1 x_1' x_2 x_2' \end{aligned} \tag{4.10}$$

En este caso $M = 6$, y si tomamos como funciones base $h_1(\mathbf{x}) = 1$, $h_2(\mathbf{x}) = \sqrt{2}x_1$, $h_3(\mathbf{x}) = \sqrt{2}x_2$, $h_4(\mathbf{x}) = x_1^2$, $h_5(\mathbf{x}) = x_2^2$, $h_6(\mathbf{x}) = \sqrt{2}x_1 x_2$, es decir, se tiene que $h(\mathbf{x}) = \{1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1 x_2\}$.

Sean ahora los vectores $\mathbf{u} = (1, 2)$ y $\mathbf{v} = (3, 4)$. Vamos a calcular $h(\mathbf{u}) \cdot h(\mathbf{v})$ de dos maneras, sin utilizar la función kernel y utilizándola.

Se cumple que $h(\mathbf{u}) = \{1, \sqrt{2}, 2\sqrt{2}, 1, 4, 2\sqrt{2}\}$ y $h(\mathbf{v}) = \{1, 3\sqrt{2}, 4\sqrt{2}, 9, 16, 12\sqrt{2}\}$. En consecuencia $h(\mathbf{u}) \cdot h(\mathbf{v}) = 1 + 3 \cdot 2 + 8 \cdot 2 + 9 + 4 \cdot 16 + 24 \cdot 2 = 1 + 6 + 16 + 9 + 64 + 48 = 144$.

Calculamos ahora usando la función kernel $h(\mathbf{u}) \cdot h(\mathbf{v}) = (1 + \langle \mathbf{u}, \mathbf{v}\rangle)^2 = (1 + 1 \cdot 3 + 2 \cdot 4)^2 = (1 + 3 + 8)^2 = 12^2 = 144$.

Este ejemplo lo que nos permite ver, como habíamos comentado ya anteriormente, es que no es necesario conocer todas las funciones base, ni calcular explícitamente todos los términos, simplemente con conocer la función del kernel y evaluar es suficiente.

Una de las ventajas de usar el kernel en lugar de simplemente ampliar el espacio original usando funciones base de las características originales es computacional, pues cuando usamos kernels solo es necesario calcular $K(\mathbf{x}_i, \mathbf{x}_{i'})$ para todos los $\binom{n}{2}$ pares de índices i, i' distintos. Esto se puede hacer sin trabajar explícitamente en el espacio de características ampliado y es importante porque en muchas aplicaciones de SVM, el espacio de características ampliado es tan grande que los cálculos son intratables. Para algunos núcleos,

como el kernel radial de la ecuación (4.8), el espacio de características es implícito y de dimensión infinita, por lo que nunca podríamos hacer los cálculos allí de todos modos.

A parte de estas funciones kernel que se han expuesto anteriormente, se pueden crear otras. Para ello, veamos una caracterización de estas funciones proporcionada por un teorema de análisis funcional cuya demostración queda fuera del alcance de este trabajo.

El uso de una función kernel, como se ha comentado anteriormente, es un atajo computacional atractivo. Si deseamos utilizar este enfoque, parece que es necesario crear primero un espacio de características complicado, luego determinar cuál sería el producto interno en ese espacio y, finalmente, encontrar un método directo para calcular ese valor en términos de las entradas originales. En la práctica, el enfoque adoptado es definir una función del kernel directamente, por lo tanto, definir implícitamente el espacio de características. De esta forma, se evita el espacio de características no solo en el cálculo de productos internos, sino también en el diseño de la propia máquina de aprendizaje. Para ello, primero debemos determinar qué propiedades de una función $K(\mathbf{x}, \mathbf{x}')$ son necesarias para asegurarnos de que es un núcleo para algún espacio de características. En el Teorema 4.3 hemos visto dos propiedades que se tienen que cumplir, que sea simétrica y semidefinida positiva. Además, existe una contribución del análisis funcional, en la que no entraremos en detalle, que proporciona una caracterización a estas funciones kernel, y se conoce como el *Teorema de Mercer*. Este asegura que toda función $K(\mathbf{x}, \mathbf{x}')$ que verifica

$$\int_{X \times X} K(\mathbf{x}, \mathbf{x}') g(\mathbf{x}) g(\mathbf{x}') dx dx' > 0$$

para toda función $g(\cdot)$ de cuadrado integrable, es decir, cuya integral del cuadrado de su módulo definida en el intervalo de definición converge, es una función kernel. Este teorema nos permite caracterizar funciones kernel y por tanto poder crearlas.

También se pueden crear nuevos núcleos mediante transformaciones de alguno que ya se conoce. La siguiente proposición nos permite crear kernels más complicados a partir de otros más sencillos.

Proposición 4.5. *Sean K_1 y K_2 funciones kernel sobre $X \times X$, con $X \subseteq \mathbb{R}^n$, $a \in \mathbb{R}^+$, $f(\cdot)$ una función de valores reales sobre X, $h : X \to \mathbb{R}^m$, con K_3 una función kernel sobre $\mathbb{R}^m \times \mathbb{R}^m$ y B una matriz $n \times n$ semidefinida positiva y simétrica. Las siguientes funciones son kernels:*

1. $K(\mathbf{x}, \mathbf{x}') = K_1(\mathbf{x}, \mathbf{x}') + K_2(\mathbf{x}, \mathbf{x}')$.
2. $K(\mathbf{x}, \mathbf{x}') = aK_2(\mathbf{x}, \mathbf{x}')$.
3. $K(\mathbf{x}, \mathbf{x}') = K_1(\mathbf{x}, \mathbf{x}') \cdot K_2(\mathbf{x}, \mathbf{x}')$.
4. $K(\mathbf{x}, \mathbf{x}') = f(\mathbf{x}) \cdot f(\mathbf{x}')$.
5. $K(\mathbf{x}, \mathbf{x}') = K_3(h(\mathbf{x}), h(\mathbf{x}'))$.
6. $K(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T B \mathbf{x}'$

Demostración. Fijemos un conjunto finito de puntos $x_1, \ldots, x_l$, y sean $\mathbf{K}_1$ y $\mathbf{K}_2$, las correspondientes matrices obtenidas al restringir K_1 y K_2 a estos puntos. Considereremos ahora cualquier vector $\alpha \in \mathbb{R}^l$. Recordemos que una matriz $\mathbf{K}$ es semidefinida positiva si y solo si $\alpha' \mathbf{K} \alpha \geq 0$, para todo α.

1. Tenemos
$$\alpha'(\mathbf{K}_1 + \mathbf{K}_2)\alpha = \alpha'\mathbf{K}_1\alpha + \alpha'\mathbf{K}_2\alpha \geq 0$$
y por tanto $\mathbf{K}_1 + \mathbf{K}_2$ es semidefinida positiva y $K_1 + K_2$ es una función kernel.

2. De una manera similar $\alpha' a\mathbf{K}_1\alpha \geq 0$, probando que aK_1 es un kernel.

3. Sea $\mathbf{K} = \mathbf{K}_1 \bigotimes \mathbf{K}_2$ el producto tensorial de las matrices $\mathbf{K}_1$ y $\mathbf{K}_2$. El producto tensorial de dos matrices semidefinidas positivas es también semidefinido positivo ya que los autovalores del producto son todos pares de productos de los autovalores de los dos componentes. La matriz correspondiente a la función K_1K_2 se conoce como el producto Schur $\mathbf{H}$ de $\mathbf{K}_1$ y $\mathbf{K}_2$ cuyas entradas son los productos de las entradas correspondientes de las dos matrices. La matriz $\mathbf{H}$ es una submatriz principal de $\mathbf{K}$ definida por un conjunto de columnas y el mismo conjunto de filas. Por lo tanto, para cualquier $\alpha \in \mathbb{R}^l$, hay un $\alpha_1 \in \mathbb{R}^{l^2}$ correspondiente, tal que
$$\alpha'\mathbf{H}\alpha = \alpha_1'\mathbf{K}\alpha_1 \geq 0,$$
y por tanto $\mathbf{H}$ es semidefinida positiva como queríamos ver.

4. Podemos reorganizar la forma bilineal como sigue
$$\begin{aligned}\sum_{i=1}^{\ell}\sum_{j=1}^{\ell}\alpha_i\alpha_j K(\mathbf{x}_i, \mathbf{x}_j) &= \sum_{i=1}^{\ell}\sum_{j=1}^{\ell}\alpha_i\alpha_j f(\mathbf{x}_i) f(\mathbf{x}_j) \\ &= \sum_{i=1}^{\ell}\alpha_i f(\mathbf{x}_i)\sum_{j=1}^{\ell}\alpha_j f(\mathbf{x}_j), \\ &= \left(\sum_{i=1}^{\ell}\alpha_i f(\mathbf{x}_i)\right)^2 \geq 0\end{aligned}$$
y en consecuencia hemos probado que es una función kernel.

5. Como K_3 es una función kernel, la matriz obtenida restringiéndola por los puntos $h(\mathbf{x}_1), \ldots, h(\mathbf{x}_l)$ es semidefinida positiva como queríamos.

6. Consideremos la diagonalización de la matriz $B = V^T\Lambda V$, siendo V una matriz ortogonal y Λ la matriz diagonal que contiene los autovalores no negativos. Sea $\sqrt{\Lambda}$ la matriz diagonal con las raíces cuadradas de los autovalores, y sea $A = \sqrt{\Lambda}V$. Tenemos por tanto que se cumple
$$K(\mathbf{x}, \mathbf{z}) = \mathrm{x}'B\mathbf{z} = \mathbf{x}^T V^T\Lambda V\mathrm{z} = \mathbf{x}^T V^T\sqrt{\Lambda}\sqrt{\Lambda}V\mathbf{z} = \mathbf{x}^T A^T A\mathbf{z} = \langle A\mathbf{x} \cdot A\mathbf{z}\rangle$$
utilizando en el producto interno la transformación de características de matriz A.

□

Otro corolario que nos permite obtener funciones *kernel* a partir de una ya conocida además de demostrarnos porqué la función radial lo es también, es el siguiente

Corolario 4.6. *Sea $K_1(\mathbf{x}, \mathbf{z})$ una función kernel sobre $X \times X$, $\mathbf{x}, \mathbf{z} \in X$, y $P(\mathbf{x})$ un polinomio con coeficientes positivos. Entonces las siguientes funciones también son kernel:*

1. $K(\mathbf{x}, \mathbf{z}) = P(K_1(\mathbf{x}, \mathbf{z}))$,

2. $K(\mathbf{x}, \mathbf{z}) = \exp(K_1(\mathbf{x}, \mathbf{z}))$,

3. $K(\mathbf{x}, \mathbf{z}) = \exp(-\|\mathbf{x} - \mathbf{z}\|^2/\sigma^2)$.

Demostración. Veamos cada una de los puntos

1. Para un polinomio el resultado es inmediato al combinar las partes de la Proposición 4.7. Tenga en cuenta que la constante está cubierta por el punto 4 de dicha proposición.

2. La función exponencial se puede aproximar arbitrariamente mediante polinomios con coeficientes positivos y por lo tanto es un límite de núcleos. Dado que los núcleos están claramente cerrados tomando límites puntuales, se demuestra el resultado.

3. Podemos descomponer la función como sigue:

$$\exp\left(-\|\mathbf{x} - \mathbf{z}\|^2/\sigma^2\right) = \exp\left(-\|\mathbf{x}\|^2/\sigma^2\right) \exp\left(-\|\mathbf{z}\|^2/\sigma^2\right) \exp\left(2(\mathbf{x} \cdot \mathbf{z}\rangle/\sigma^2\right).$$

   Los primeros dos factores forman un kernel por el apartado 4 de la proposición anterior, mientras que el tercer factor lo es también por el segundo punto de este corolario.

□

Volviendo a la formulación del problema en el caso no separable linealmente, el problema primal asociado a este caso, es muy similar al que hemos descrito previamente para el caso lineal y casi lineal, con la diferencia de que el hiperplano de separación que buscamos se determina en el espacio de características, es decir

$$\begin{aligned} \text{mín} \quad & \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{n} \xi_i \\ \text{sujeto a:} \quad & y_i h(\mathbf{x}_i)^T \beta \geq 1 - \xi_i \quad i = 1, \ldots, n \\ & \xi_i \geq 0 \quad i = 1, \ldots, n. \end{aligned} \tag{4.11}$$

La complejidad de este problema primal (4.11) depende de la dimensión del espacio de características, que en general es alta, llegando incluso a infinita como se ha comentado anteriormente. Por ello de nuevo, se considera el problema dual asociado, cuya complejidad depende del número de observaciones. Dado el conjunto de funciones base $h = \{h_1(\mathbf{x}), \ldots, h_M(\mathbf{x})\}$, el problema a resolver es encontrar el valor de los parámetros α_i^*, $i = 1, \ldots, n$, que optimiza el siguiente problema dual

$$\begin{aligned} \text{máx} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j h(\mathbf{x}_i)^T h(\mathbf{x}_j) \\ \text{sujeto a:} \quad & \sum_{i=1}^{n} \alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C \quad i = 1, \ldots, n. \end{aligned} \tag{4.12}$$

Se puede observar que no es necesario conocer las componentes h_i de la función, sino sólo los productos escalares de $h(\mathbf{x}_i), h(\mathbf{x}_j)$ y por tanto se puede reescribir el problema de optimización dual con la función kernel, por la Definición 4.1, como

$$\begin{aligned} \text{máx} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \\ \text{sujeto a:} \quad & \sum_{i=1}^{n} \alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C \quad i = 1, \ldots, n. \end{aligned} \tag{4.13}$$

Además por la Definición 4.1, se puede sustituir el producto escalar en (4.2) por la función kernel, es decir, podemos construir la función de decisión sin usar explícitamente la función de transformación h como

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i^* y_i K(\mathbf{x}, \mathbf{x}_i), \tag{4.14}$$

donde los parámetros α_i^*, $i = 1, \ldots, n$ son las soluciones del problema de optimización cuadrático dado por (4.13), conocidos el conjunto de observaciones de entrenamiento $\mathcal{C} = \{(\mathbf{x}_i, y_i),\, i = 1, \ldots, n\}$, la función kernel K y el parámetro de regularización C, para el que se utilizan técnicas de validación cruzada para encontrar la mejor solución de ese parámetro.

Tengamos en cuenta que en (4.14), para evaluar la función $f(\mathbf{x})$, necesitamos calcular el producto interno entre el nuevo punto $\mathbf{x}$ y cada uno de los puntos de entrenamiento $\mathbf{x}_i$. Sin embargo, resulta que α_i es distinto de cero solo para los vectores de soporte en la solución; es decir, si una observación de entrenamiento no es un vector de soporte, entonces su α_i es igual a cero, como habíamos visto para el *soft*-SVM. Entonces,dado el conjunto $\mathcal{SV}$ de índices de vectores soporte, podemos reescribir la función (4.14) como

$$f(\mathbf{x}) = \sum_{i \in \mathcal{SV}} \alpha_i^* y_i K(\mathbf{x}, \mathbf{x}_i), \tag{4.15}$$

que típicamente involucra muchos menos términos que (4.14).

Resumiendo todo ello, encontramos la siguiente proposición

Proposición 4.7. *Consideremos el conjunto de observaciones de entrenamiento $\mathcal{C}$, definido por $\mathcal{C} = \{(\mathbf{x}_i, y_i),\, i = 1, \ldots, n\}$. Sea el espacio de características implícitamente definido por la función kernel $K(\mathbf{x}, \mathbf{x}')$, y supongamos que el parámetro α^* resuelve el problema de optimización cuadrática:*

$$\begin{aligned} \text{máx} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j h(\mathbf{x}_i)^T h(\mathbf{x}_j) \\ \textit{sujeto a:} \quad & \sum_{i=1}^{n} \alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C \quad i = 1, \ldots, n. \end{aligned} \tag{4.16}$$

Sea $f(\mathbf{x}) = \sum_{i \in sv} \alpha_i^* y_i K(\mathbf{x}, \mathbf{x}_i)$, donde β_0^* se elige de modo que $y_i f(\mathbf{x}_i) = 1$ para cualquier i con $C > \alpha^* > 0$. Entonces, la regla de decisión dada por el signo de $f(\mathbf{x})$ es equivalente al

hiperplano en el espacio de características implícitamente definido por el kernel $K(\mathbf{x}, \mathbf{x}')$ *que resuelve el problema de optimización* (4.11)*, donde las variables de holgura se definen en relación con el margen*

$$M = \left(\sum_{i \in \mathcal{SV}} \alpha_i^* - \frac{1}{C} \|\alpha^*\|^2 \right)^{-1/2}. \tag{4.17}$$

Demostración. El valor de β_0^* se elige usando las condiciones de complementariedad de Karush-Kuhn-Tucker que implican que si $C > \alpha^* > 0$ se tiene que cumplir tanto $\xi^* = 0$ como

$$y_i(\mathbf{x}_i^T \beta^* + b^*) - 1 + \xi_i^* = 0.$$

La norma de β^* viene dada por la siguiente expresión.

$$\begin{aligned} \|\beta^*\|^2 &= \sum_{i,j=1}^{n} y_i y_j \alpha_i^* \alpha_j^* K(\mathbf{x}_i, \mathbf{x}_j) \\ &= \sum_{j \in \mathcal{SV}} \sum_{i \in \mathcal{SV}} y_i y_j \alpha_i^* \alpha_j^* K(\mathbf{x}_i, \mathbf{x}_j). \end{aligned} \tag{4.18}$$

□

En resumen, para resolver el problema dual (4.13) no se necesita conocer el conjunto de funciones base para la transformación y tampoco es necesario conocer las coordenadas de las observaciones transformadas en el espacio de características. Únicamente se necesita conocer la forma funcional del kernel correspondiente $K : X \times X \to \mathbb{R}$, incluso cuando el conjunto de funciones base fuera infinito.

Entre las distintas propiedades que podemos encontrar de los SVM aplicando las funciones kernel, tenemos que se puede representar cualquier función booleana y cualquier función suficientemente "suave", es decir, infinitamente diferenciable, con precisión arbitraria siempre con una elección adecuada de la función kernel. Además, la función objetivo no tiene óptimos locales, solo uno global y es independiente de la dimensión del espacio de características. Por último, también permite crear fronteras diferentes en función de la función kernel utilizada, y de sus parámetros además del valor de coste C.

Capítulo 5

Algoritmo del Gradiente Estocástico

En este capítulo describiremos y analizaremos un enfoque de aprendizaje especializado, que se llama descenso de gradiente estocástico.

Antes de tratar el método de descenso de gradiente estocástico, trataremos el *descenso de gradiente estándar*, que es una técnica de optimización ampliamente utilizada en aprendizaje automático y en la resolución de problemas de optimización en general. El descenso de gradiente es un procedimiento de optimización iterativo en el que en cada paso mejoramos la solución dando un paso a lo largo del negativo del gradiente de la función a minimizar. Así, para aplicar descenso por gradiente, se debe calcular el gradiente de la función de objetivo asociada a los parámetros a optimizar. En cada paso, se actualizan los valores de los parámetros restando el producto del gradiente y la tasa de aprendizaje. Este proceso se repite hasta que se alcanza un mínimo local de la función de objetivo o se alcanza un número predeterminado de iteraciones. Es importante tener en cuenta que el descenso de gradiente puede ser sensible a la elección de la tasa de aprendizaje y la inicialización de los parámetros del modelo por la posible existencia de múltiples mínimos locales. Es necesario ajustar cuidadosamente estos parámetros para garantizar una convergencia rápida y estable del algoritmo.

En este capitulo nos centraremos sobre la familia de problemas de aprendizaje convexos y muy especialmente en el SVM. Siguiendo la notación de los capítulos anteriores, nos referiremos a β como los vectores que definen los clasificadores (basados en hiperplanos) en $\mathcal{H}$.

En el algoritmo de *descenso de gradiente estocástico*, que vamos posteriormente a presentar, se tratará de minimizar la función de riesgo esperada

$$R(\beta) = E_Z[\ell(\beta, Z)]$$

usando directamente un procedimiento de descenso del gradiente sobre esa misma función de riesgo esperado $R(\beta)$. Como no conocemos la distribución conjunta Z, tampoco conoceremos el gradiente de dicha función de riesgo esperado $R(\beta)$, pero el algoritmo del descenso del gradiente estocástico evita este problema al permitir que el procedimiento de optimización dé un paso a lo largo de una dirección aleatoria, siempre que el valor esperado de la dirección sea el negativo del gradiente. Y, como veremos, encontrar una dirección aleatoria cuyo valor esperado corresponda a ese negativo del gradiente es bastante simple aunque no conozcamos la distribución subyacente Z. La ventaja de este algoritmo en el contexto de problemas de aprendizaje convexos, sobre otros, es que es un algoritmo eficiente que se puede implementar en unas pocas líneas de código.

En este capítulo trataremos primero el algoritmo de descenso de gradiente básico y analizaremos su tasa de convergencia para funciones convexas de Lipschitz. A continuación, presentaremos la noción de subgradiente y mostraremos que el descenso de gradiente también se puede aplicar a funciones no diferenciables. El tema principal de este capítulo al que queremos llegar es la descripción del algoritmo de descenso de gradiente estocástico, junto con alguna de sus variantes.

Descenso de gradiente

Como hemos comentado, antes de describir el método de descenso de gradiente estocástico, trataremos el enfoque de descenso de gradiente estándar para minimizar una función convexa diferenciable $f(\beta)$.

El *gradiente* de una función diferenciable $f : \mathbb{R}^d \to \mathbb{R}$ en β, se denota por $\nabla f(\beta)$ y es el vector de derivadas parciales de f, es decir,

$$\nabla f(\beta) = \left(\frac{\partial f(\beta)}{\partial \beta 1}, \cdots, \frac{\partial f(\beta)}{\partial \beta d} \right).$$

El *descenso de gradiente* es un algoritmo iterativo. Empezamos con un valor inicial de β, por ejemplo, $\beta^{(1)} = 0$. Entonces, en cada iteración, damos un paso en la dirección del negativo del gradiente en el punto actual. Es decir, el paso de actualización es

$$\beta^{(t+1)} = \beta^{(t)} - \eta \nabla f(\beta^{(t)}), \tag{5.1}$$

donde $\eta > 0$ es un parámetro que se discutirá más adelante. Intuitivamente, dado que el gradiente apunta en la dirección de la mayor tasa de aumento de f alrededor de $\beta^{(t)}$, el algoritmo da un pequeño paso en la dirección opuesta, disminuyendo así el valor de la función. Finalmente, después de T iteraciones, el algoritmo genera el vector promediado, $\tilde{\beta} = \frac{1}{T} \sum_{t=1}^{T} \beta^{(t)}$. La salida también podría ser el último vector, $\beta^{(T)}$, o el vector de "mejor" rendimiento $\text{argmin}_{t=1,\ldots,T} f(\beta^{(t)})$, pero tomar el promedio resulta bastante útil, especialmente cuando generalizamos el descenso de gradiente a funciones no derivables y al caso estocástico.

Otra forma de motivar el descenso de gradiente es la aproximación de Taylor. El gradiente de la función f en β produce la aproximación de Taylor de primer orden de f alrededor de β por $f(\mathbf{u}) \approx f(\beta) + \langle \mathbf{u} - \beta, \nabla f(\beta) \rangle$. Cuando f es una función convexa, esta aproximación es una cota inferior de la función f, esto es,

$$f(\mathbf{u}) \geq f(\beta) + \langle \mathbf{u} - \beta, \nabla f(\beta) \rangle.$$

Por lo tanto, para cualquier β cercano a $\beta^{(t)}$ tenemos que $f(\beta) \approx f(\beta^{(t)}) + \langle \beta - \beta^{(t)}, \nabla f(\beta^{(t)}) \rangle$. En consecuencia, podemos minimizar la aproximación de $f(\beta)$. Sin embargo, la aproximación podría perderse para algún β que esté muy lejos de $\beta^{(t)}$. Por lo tanto, nos gustaría minimizar conjuntamente la distancia entre β y $\beta^{(t)}$ y la aproximación de f alrededor de $\beta^{(t)}$. Si el parámetro η controla la compensación entre los dos términos, obtenemos la regla de actualización

$$\beta^{(t+1)} = \underset{\beta}{\text{argmin}} \frac{1}{2} \|\beta - \beta^{(t)}\|^2 + \eta (f(\beta^{(t)}) + \langle \beta - \beta^{(t)}, \nabla f(\beta^{(t)}) \rangle).$$

Resolviendo lo anterior tomando la derivada con respecto a β y comparándola con cero se obtiene la misma regla de actualización que en la ecuación (5.1). Primero, notemos que

$$\frac{\partial}{\partial\beta}\left[\frac{1}{2}\|\beta-\beta^{(t)}\|^2\right]=\beta-\beta^{(t)},$$
$$\frac{\partial}{\partial\beta}\langle\beta-\beta^{(t)},\nabla f(\beta^{(t)})\rangle=\nabla f(\beta^{(t)}).$$

Entonces, tomando la derivada de la ecuación con respecto a β de la ecuación anterior, obtenemos

$$\frac{\partial}{\partial\beta}\left[\frac{1}{2}\|\beta-\beta^{(t)}\|^2+\eta(f(\beta^{(t)})+\langle\beta-\beta^{(t)},\nabla f(\beta^{(t)})\rangle)\right]=(\beta-\beta^{(t)})+\eta\nabla f(\beta^{(t)}).$$

Igualando esto a 0, tenemos que

$$\beta=\beta^{(t)}-\eta\nabla f(\beta^{(t)}).$$

Por lo tanto, el valor de β que minimiza la función dada es $\beta^{(t)}-\eta\nabla f(\beta^{(t)})$ como queríamos probar.

Para analizar la tasa de convergencia del algoritmo de descenso de gradiente, nos limitamos al caso de las funciones de Lipschitz convexas, pues muchos problemas se prestan fácilmente a esta configuración. Se dice que una función $f:A\to\mathbb{R}$ es *ρ-Lipschitz* si para todo $\mathbf{u},\mathbf{v}\in A$ se cumple

$$|f(\mathbf{u})-f(\mathbf{v})|\leq\rho\|\mathbf{u}-\mathbf{v}\|.$$

Sea β^* cualquier vector y sea B una cota de su norma $\|\beta^*\|\leq B$. Es conveniente pensar en β^* como el mínimo de $f(\beta)$, pero el análisis que vamos a hacer a continuación se cumple para cada β^*.

Nos gustaría obtener una cota superior del error al usar $f(\bar{\beta})$ para aproximar $f(\beta^*)$ donde

$$\bar{\beta}=\frac{1}{T}\sum_{t=1}^{T}\beta^{(t)}.$$

La idea es que esta cota superior nos da una idea de cómo de cerca estamos de la solución óptima.

De la definición de $\bar{\beta}$ y usando la desigualdad de Jensen, tenemos que

$$\begin{aligned}f(\bar{\beta})-f(\beta^\star)&=f\left(\frac{1}{T}\sum_{t=1}^{T}\beta^{(t)}\right)-f(\beta^\star)\\&\leq\frac{1}{T}\sum_{t=1}^{T}\left(f(\beta^{(t)})\right)-f(\beta^\star)\\&=\frac{1}{T}\sum_{t=1}^{T}\left(f(\beta^{(t)})-f(\beta^\star)\right).\end{aligned}\tag{5.2}$$

Para cualquier t, por la convexidad de la función f, se tiene que

$$f(\beta^{(t)})-f(\beta^\star)\leq\langle\beta^{(t)}-\beta^\star,\nabla f(\beta^{(t)})\rangle\tag{5.3}$$

y combinando las ecuaciones (5.2) y (5.3), se obtiene

$$f(\bar{\beta}) - f(\beta^\star) \leq \frac{1}{T}\sum_{t=1}^{T}\langle \beta^{(t)} - \beta^\star, \nabla f(\beta^{(t)})\rangle.$$

Para acotar el lado derecho nos basamos en el siguiente lema:

Lema 5.1. *Sean $\nu_1, \ldots, \nu_T$ un conjunto de vectores arbitrarios. Cualquier algoritmo con inicialización $\beta^{(1)} = 0$ y una regla de actualización de la forma*

$$\beta^{(t+1)} = \beta^{(t)} - \eta\nu_t \tag{5.4}$$

cumple que

$$\sum_{t=1}^{T}\langle \beta^{(t)} - \beta^\star, \nu_t\rangle \leq \frac{\|\beta^\star\|^2}{2\eta} + \frac{\eta}{2}\sum_{t=1}^{T}\|\nu_t\|^2. \tag{5.5}$$

En particular, para cualquier $B, \rho > 0$, si para todo t se tiene que $\|\nu_t\| \leq \rho$ y si establecemos $\eta = \sqrt{\frac{B^2}{\rho^2 T}}$, entonces para todo $\beta^\star$ con $\|\beta^\star\| \leq B$, se cumple que

$$\frac{1}{T}\sum_{t=1}^{T}\langle \beta^{(t)} - \beta^\star, \nu_t\rangle \leq \frac{B\rho}{\sqrt{T}}.$$

Demostración. Usando manipulaciones algebraicas, es decir, completando cuadrados, obtenemos

$$\begin{aligned}
\langle \beta^{(t)} - \beta^\star, \nu_t\rangle &= \frac{1}{\eta}\langle \beta^{(t)} - \beta^\star, \eta\nu_t\rangle \\
&= \frac{1}{2\eta}\left(-\|\beta^{(t)} - \beta^\star - \eta\nu_t\|^2 + \|\beta^{(t)} - \beta^\star\|^2 + \eta^2\|\nu_t\|^2\right) \\
&= \frac{1}{2\eta}\left(-\|\beta^{(t+1)} - \beta^\star\|^2 + \|\beta^{(t)} - \beta^\star\|^2\right) + \frac{\eta}{2}\|\nu_t\|^2,
\end{aligned} \tag{5.6}$$

donde la última igualdad se deriva de la definición de la regla de actualización (5.4). Sumando la igualdad sobre t, tenemos

$$\sum_{t=1}^{T}\langle \beta^{(t)} - \beta^\star, \nu_t\rangle = \frac{1}{2\eta}\sum_{t=1}^{T}\left(-\|\beta^{(t+1)} - \beta^*\|^2 + \|\beta^{(t)} - \beta^*\|^2\right) + \frac{\eta}{2}\sum_{t=1}^{T}\|\nu_t\|^2. \tag{5.7}$$

La primera suma en el lado derecho es una serie telescópica, es decir, es una serie cuyas sumas parciales poseen un número fijo de términos tras su cancelación, en este caso el resultado es

$$\|\beta^{(1)} - \beta^\star\|^2 - \|\beta^{(T+1)} - \beta^\star\|^2.$$

Sustituyendo esta igualdad en la ecuación (5.7), se tiene

$$\begin{aligned}
\sum_{t=1}^{T}\langle \beta^{(0)} - \beta^\star, \nu_t\rangle &= \frac{1}{2\eta}\left(\|\beta^{(1)} - \beta^*\|^2 - \|\beta^{(T+1)} - \beta^\star\|^2\right) + \frac{\eta}{2}\sum_{t=1}^{T}\|\nu_t\|^2 \\
&\leq \frac{1}{2\eta}\|\beta^{(1)} - \beta^\star\|^2 + \frac{\eta}{2}\sum_{t=1}^{T}\|\nu_t\|^2 \\
&= \frac{1}{2\eta}\|\beta^*\|^2 + \frac{\eta}{2}\sum_{t=1}^{T}\|\nu_t\|^2
\end{aligned}$$

donde la última igualdad se debe a la definición $\beta^{(1)} = 0$. Esto prueba (5.5), y la segunda parte del lema se tiene por el límite superior de $\|\beta^\star\|$ por B, $\|\nu_t\|$ por ρ, dividiendo entre T y reemplazando el valor de η . □

Si el Lema 5.1 se aplica al algoritmo de descenso de gradiente con $\nu_t = \nabla f(\beta^{(t)})$, por satisfacerse las condiciones de dicho lema, se llega al siguiente corolario:

Corolario 5.2. *Sea f una función ρ-Lipschitz convexa, y sea*

$$\beta^\star \in \text{argmin}_{\{\beta:\|\beta\|\leq B\}} f(\beta).$$

Si ejecutamos el algoritmo de descenso de gradiente en f para T pasos con $\eta = \sqrt{\frac{B^2}{\rho^2 T}}$, entonces el vector de salida $\bar{\beta}$ satisface la siguiente desigualdad

$$f(\bar{\beta}) - f\left(\beta^\star\right) \leq \frac{B\rho}{\sqrt{T}}.$$

Además, para todo $\varepsilon > 0$, para conseguir $f(\bar{\beta}) - f\left(\beta^\star\right) \leq \varepsilon$, es suficiente ejecutar el algoritmo de descenso de gradiente un número de iteraciones T, tal que

$$T \geq \frac{B^2\rho^2}{\varepsilon^2}.$$

El algoritmo del descenso del gradiente requiere que la función f sea diferenciable. Ahora vamos a generalizarlo más allá de las funciones diferenciables. Mostraremos que este algoritmo se puede aplicar a funciones no diferenciables usando en lugar del gradiente, el llamado subgradiente de $f(\beta)$ en $\beta^{(t)}$.

Para motivar la definición de los subgradientes, tenemos que para una función convexa f, el gradiente en β define la pendiente de una tangente que se encuentra debajo de f.. Es decir, para todo $\mathbf{u}$,

$$f(\mathbf{u}) \geq f(\beta) + \langle \mathbf{u} - \beta, \nabla f(\beta) \rangle. \tag{5.8}$$

Esto lo podemos ver en la parte izquierda de la Figura 5.1.

La existencia de una tangente que se encuentra debajo de f es una propiedad importante de las funciones convexas, que de hecho es una caracterización alternativa de la convexidad.

Lema 5.3. *Sea $\mathcal{S}$ un conjunto abierto y convexo. Una función $f : \mathcal{S} \to \mathbb{R}$ es convexa si y solo si para todo $\beta \in \mathcal{S}$ existe un $\mathbf{v}$ tal que para todo $\mathbf{u} \in \mathcal{S}$*

$$f(\mathbf{u}) \geq f(\beta) + \langle \mathbf{u} - \beta, \mathbf{v} \rangle. \tag{5.9}$$

Esta desigualdad nos permite definir lo que es un subgradiente.

Definición 5.4. Un vector $\mathbf{v}$ que satisface la ecuación (5.9) se llama *subgradiente* de f en β. El conjunto de subgradientes de f en β se denomina *conjunto diferencial* y se denota como $\partial f(\beta)$.

En el lado derecho de la Figura 5.1 se muestra una ilustración de los subgradientes. Para funciones escalares, un subgradiente de una función convexa f en β es una pendiente de una línea que toca f en β y no está por encima de f en ningún otro lugar.

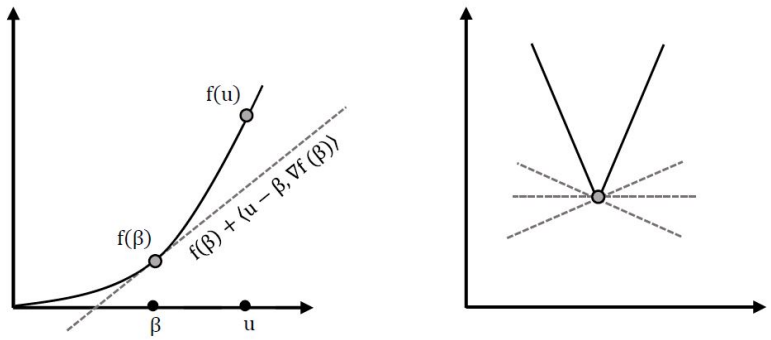

Figura 5.1: En la imagen de la izquierda podemos ver que el lado derecho de la ecuación (5.8) es la tangente de f en β. Para una función convexa, los límites inferiores de la tangente f. En la imagen de la derecha podemos ver varios subgradientes de una función convexa no diferenciable.

Para construir el subgradiente de una función convexa dada, se tiene que si una función es derivable en un punto β, entonces el conjunto diferencial es trivial, pues $\partial f(\beta)$ contiene un solo elemento: el gradiente de f en β, $\nabla f(\beta)$.

Veamos a continuación un ejemplo de cálculo del conjunto diferencial de una función no diferenciable.

Ejemplo 5.5. Si consideramos la función $f(\mathbf{x}) = \|\mathbf{x}\|$, usando todo lo anterior, podemos construir fácilmente el conjunto diferencial para las partes diferenciables de f, y el único punto que requiere especial atención es $\mathbf{x}_0 = 0$. En ese punto, es fácil ver que el subdiferencial es cualquier valor en el intervalo -1 y 1. Por lo tanto podemos definirlo de manera completa como

$$\partial f(\mathbf{x}) = \begin{cases} \{1\} & \text{if } \mathbf{x} > 0 \\ \{-1\} & \text{if } \mathbf{x} < 0 \\ [-1, 1] & \text{if } \mathbf{x} = 0 \end{cases}$$

Para muchos usos prácticos, no necesitamos calcular todo el conjunto de subgradientes en un punto dado, ya que bastaría con un miembro de este conjunto. La siguiente afirmación muestra cómo construir un subgradiente para funciones máximas en un punto.

Proposición 5.6. *Sea* $g(\beta) = \max_{i=1,\dots,r} g_i(\beta)$ *para* r *funciones diferenciables convexas* $g_1, \dots, g_r$. *Para* $j \in \operatorname{argmax}_{i=1,\dots,r} g_i(\beta)$ *se tiene*

$$\nabla g_j(\beta) \in \partial g(\beta).$$

Demostración. Como g_j es convexa, tenemos por el Lema 5.3 que para todo $\mathbf{u}$

$$g_j(\mathbf{u}) \geq g_j(\beta) + \langle \mathbf{u} - \beta, \nabla g_j(\beta) \rangle .$$

Como $g(\beta) = g_j(\beta)$ y $g(\mathbf{u}) \geq g_j(\mathbf{u})$ obtenemos que

$$g(\mathbf{u}) \geq g(\beta) + \langle \mathbf{u} - \beta, \nabla g_j(\beta) \rangle ,$$

lo que concluye la prueba. □

Para verlo con un ejemplo, vamos a calcular el subgradiente de la función de pérdida de hinge.

Ejemplo 5.7. Recordemos para ello que la función de pérdida de hinge, vista en capítulos anteriores, se definía como $f(\beta) = \max\{0, 1 - y\langle\beta, \mathbf{x}\rangle\}$ para un vector $\mathbf{x}$ y un escalar y. Para calcular un subgradiente de la función de pérdida de hinge en algún β, nos basamos en la proposición anterior y obtenemos que el vector definido a continuación es un subgradiente de esta función de pérdida en β:

$$\partial f(\beta) = \begin{cases} 0 & \text{si } 1 - y\langle\beta, \mathbf{x}\rangle \leq 0 \\ -y\mathbf{x} & \text{si } 1 - y\langle\beta, \mathbf{x}\rangle > 0 \end{cases}$$

El siguiente lema nos da una definición equivalente usando las normas de los subgradientes:

Lema 5.8. *Sea A un conjunto abierto y convexo y sea $f : A \to \mathbb{R}$ una función convexa. Entonces, f es ρ-Lipschitz sobre A si y solo si para todo $\beta \in A$ y $\mathbf{v} \in \partial f(\beta)$ se cumple que $\|\mathbf{v}\| \leq \rho$.*

Demostración. Veamos la implicación de derecha a izquierda. Para ello supongamos que para todo $\mathbf{v} \in \partial f(\beta)$ se cumple que $\|\mathbf{v}\| \leq \rho$. Como $\mathbf{v} \in \partial f(\beta)$ se tiene que

$$f(\beta) - f(\mathbf{u}) \leq \langle\mathbf{v}, \beta - \mathbf{u}\rangle.$$

Acotando la parte de la derecha utilizando la desigualdad de Cauchy-Schwartz obtenemos

$$f(\beta) - f(\mathbf{u}) \leq \langle\mathbf{v}, \beta - \mathbf{u}\rangle \leq \|\mathbf{v}\|\|\beta - \mathbf{u}\| \leq \rho\|\beta - \mathbf{u}\|.$$

Utilizamos el mismo argumento para demostrar que $f(\mathbf{u}) - f(\beta) \leq \rho\|\beta - \mathbf{u}\|$. En consecuencia f es ρ-Lipschitz.

Veamos ahora la otra implicación. Supongamos que f es una función ρ-Lipschitz. Tomamos algún $\beta \in A, \mathbf{v} \in \partial f(\beta)$. Como A es un conjunto abierto, tenemos que existe $\varepsilon > 0$ tal que $\mathbf{u} = \beta + \varepsilon\mathbf{v}/\|\mathbf{v}\|$ pertenece a A. Por lo tanto, $\langle\mathbf{u} - \beta, \mathbf{v}\rangle = \varepsilon\|\mathbf{v}\|$ y $\|\mathbf{u} - \beta\| = \varepsilon$. Por la definición de subgradiente,

$$f(\mathbf{u}) - f(\beta) \geq (\mathbf{v}, \mathbf{u} - \beta\rangle = \varepsilon\|\mathbf{v}\|.$$

Por otro lado, por ser f una función Lipschitziana, tenemos

$$\rho\varepsilon = \rho\|\mathbf{u} - \beta\| \geq f(\mathbf{u}) - f(\beta).$$

Combinando las dos desigualdades, se tiene que $\|\mathbf{v}\| \leq \rho$, lo que concluye la prueba. □

Como observación de este Lema tenemos que si f es ρ-Lipschitz, entonces se cumple que $\|\nabla f(\beta^{(t)})\| \leq \rho$.

El algoritmo de descenso del gradiente se puede generalizar a funciones no diferenciables mediante el uso de un subgradiente de $f(\beta)$ en $\beta^{(t)}$, en lugar del gradiente. El análisis de la tasa de convergencia permanece sin cambios, simplemente hay que tener en cuenta que la ecuación (5.3) también es válida para los subgradientes.

Descenso de gradiente estocástico

A continuación, vamos a explicar el método del descenso de gradiente estocástico. En este método no se exige que la dirección de actualización se base exactamente en el gradiente, en cambio, se permite que la dirección sea un vector aleatorio y solo hace falta que su valor esperado en cada iteración sea igual a la dirección del gradiente, o de manera más general, se requiere que el valor esperado del vector aleatorio sea un subgradiente de la función en el vector actual.

El procedimiento de *Descenso de gradiente estocástico para minimizar una función* $f(\beta)$ puede ser esquematizado como sigue:

- Para un $\eta > 0$ y $T \in \mathbb{N}$:
- Inicializar con $\beta^{(1)} = 0$
- Para $t = 1, ..., T-1$:
  - $\diamond$ V_t vector aleatorio con $\mathbb{E}[V_t|\beta^{(t)}] \in \partial f(\beta^{(t)})$
  - $\diamond$ Actualizar $\beta^{(t+1)} = \beta^{(t)} - \eta V_t$
- Devolver $\bar{\beta} = \frac{\sum_{t=1}^{T} \beta^{(t)}}{T}$

Nótese que los $\beta^{(t)}$ son vectores aleatorios que se van actualizando en el proceso iterativo.

Para las funciones Lipschitzianas convexas ya hemos visto que existía una cota superior para el algoritmo del descenso del gradiente en el Corolario 5.2. Para el caso estocástico, en el que la esperanza condicional de V_t está en $\partial f(\beta^{(t)})$, no podemos aplicar directamente la ecuación (5.3). Sin embargo, podemos obtener una cota superior similar en el vector final esperado del descenso del gradiente estocástico. Esto lo formalizamos en el siguiente teorema.

Teorema 5.9. *Sean $B, \rho > 0$, f una función convexa y $\beta^\star$ con $\beta^\star \in \operatorname{argmin}_{\{\beta:\|\beta\|\leq B\}} f(\beta)$. Ejecutamos el algoritmo de descenso del gradiente estocástico para f en T iteraciones con $\eta = \sqrt{\frac{B^2}{\rho^2 T}}$. Suponemos además que para todo t, $\mathbb{P}[\|V_t\| \leq \rho] = 1$. Entonces el vector de salida $\bar{\beta}$ satisface la siguiente desigualdad*

$$\mathbb{E}[f(\bar{\beta})] - f(\beta^\star) \leq \frac{B\rho}{\sqrt{T}}.$$

Además, para todo $\varepsilon > 0$, para conseguir $\mathbb{E}[f(\bar{\beta})] - f(\beta^\star) \leq \varepsilon$, es suficiente ejecutar el algoritmo de descenso de gradiente estocástico para un número de iteraciones que satisfaga

$$T \geq \frac{B^2\rho^2}{\varepsilon^2}.$$

Demostración. Sea $V_1, \ldots, V_T$ una secuencia de vectores aleatorios y $\mathbb{E}_{V_1,\ldots,V_T}$ la esperanza en función de la distribución conjunta de la secuencia de vectores aleatorios $V_1, \ldots, V_T$.

Usando la desigualdad (5.2) y tomado esperanzas, se tiene

$$\mathbb{E}_{V_1,\ldots,V_T}\left[f(\bar{\beta}) - f\left(\beta^\star\right)\right] \leq \mathbb{E}_{V_1,\ldots,V_T}\left[\frac{1}{T}\sum_{t=1}^{T}\left(f\left(\beta^{(t)}\right) - f\left(\beta^\star\right)\right)\right].$$

Dado que el Lema 5.1 se cumple para cualquier sucesión $V_1, V_2, \ldots, V_T$, también se podrá aplicar al algoritmo de descenso de gradiente estocástico. Tomando esperanzas de la cota en dicho lema tendríamos que

$$\mathbb{E}_{V_1,\ldots,V_T}\left[\frac{1}{T}\sum_{t=1}^{T}\langle\beta^{(t)} - \beta^\star, V_t\rangle\right] \leq \frac{B\rho}{\sqrt{T}}.$$

Solo nos falta por probar el hecho de que se satisface

$$\mathbb{E}_{V_1,\ldots,V_T}\left[\frac{1}{T}\sum_{t=1}^{T}\left(f\left(\beta^{(t)}\right) - f\left(\beta^\star\right)\right)\right] \leq \mathbb{E}_{V_1,\ldots,V_T}\left[\frac{1}{T}\sum_{t=1}^{T}\left\langle\beta^{(t)} - \beta^\star, V_t\right\rangle\right], \tag{5.10}$$

y eso concluiría con la prueba.

Con este propósito, usando la linealidad de la esperanza, tenemos

$$\mathbb{E}_{V_1,\ldots,V_T}\left[\frac{1}{T}\sum_{t=1}^{T}\left\langle\beta^{(t)} - \beta^\star, V_t\right\rangle\right] = \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}_{V_1,\ldots,V_T}\left[\langle\beta^{(t)} - \beta^\star, V_t\rangle\right].$$

Nos fijamos primero en que

$$\mathbb{E}_{V_1,\ldots,V_T}\left[\langle\beta^{(t)} - \beta^\star, V_t\rangle\right] = \mathbb{E}_{V_1,\ldots,V_t}\left[\langle\beta^{(t)} - \beta^\star, V_t\rangle\right],$$

por como están definidos los $\beta^{(t)}$. A continuación, usamos que para dos variables aleatorias generales X e Y (no tienen que ver con el vector aleatorio X y la variable Y que usamos para modelar las variables de entrada a salida de nuestros clasificadores) se tiene que $\mathbb{E}_X[\phi(X)] = \mathbb{E}_Y\mathbb{E}_X[\phi(X)|Y]$. Fijando $X = (V_1, \ldots, V_t)$ e $Y = (V_1, \ldots, V_{t-1})$ obtenemos:

$$\begin{aligned}\mathbb{E}_{V_1,\ldots,V_T}\left[\langle\beta^{(t)} - \beta^\star, V_t\rangle\right] &= \mathbb{E}_{V_1,\ldots,V_t}\left[\langle\beta^{(t)} - \beta^\star, V_t\rangle\right] \\ &= \mathbb{E}_{V_1,\ldots,V_{t-1}}\mathbb{E}_{V_1,\ldots,V_t}\left[\left\langle\beta^{(t)} - \beta^\star, V_t\right\rangle \mid V_1, \ldots, V_{t-1}\right].\end{aligned}$$

Condicionalmente a los vectores en la secuencia $V_1, \ldots, V_{t-1}$, el valor de $\beta^{(t)}$ ya no es aleatorio y, por tanto,

$$\mathbb{E}_{V_1,\ldots,V_{t-1}}\mathbb{E}_{V_1,\ldots,V_t}\left[\left\langle\beta^{(t)} - \beta^\star, V_t\right\rangle \mid V_1, \ldots, V_{t-1}\right] = \mathbb{E}_{V_1,\ldots,V_{t-1}}\langle\beta^{(t)} - \beta^\star, \mathbb{E}_{V_t}\left[V_t|V_1, \ldots, V_{t-1}\right]\rangle.$$

Dado que $\beta^{(t)}$ solo depende de $V_1, \ldots, V_{t-1}$, y que el algoritmo de descenso del gradiente estocástico cumple

$$\mathbb{E}_{V_t}[V_t \mid \beta^{(t)}] \in \partial f(\beta^{(t)}),$$

se obtiene que

$$\mathbb{E}_{V_t}[V_t \mid V_1, ..., V_{t-1}] \in \partial f(\beta^{(t)}).$$

De este modo,

$$\mathbb{E}_{V_1,\ldots,V_{t-1}}\langle \beta^{(t)} - \beta^\star, \mathbb{E}_{V_t}\left[V_t \mid V_1,\ldots,V_{t-1}\right]\rangle \geq \mathbb{E}_{V_1,\ldots,V_{t-1}}[f(\beta^{(t)}) - f\left(\beta^\star\right)]$$

Con lo que se ha demostrado que

$$\begin{aligned}\mathbb{E}_{V_1,\ldots,V_T}[\langle \beta^{(t)} - \beta^\star, V_t\rangle] &\geq \mathbb{E}_{V_1,\ldots,V_{t-1}}[f(\beta^{(t)}) - f(\beta^\star)] \\ &= \mathbb{E}_{V_1,\ldots,V_T}[f(\beta^{(t)}) - f(\beta^\star)].\end{aligned}$$

Sumando sobre t, dividiendo por T y usando la linealidad de la esperanza, obtenemos que se cumple la ecuación (5.10), lo que concluye nuestra demostración. □

Una vez que hemos introducido y analizado el algoritmo de descenso de gradiente estocástico para funciones generales convexas. Ahora veamos su uso en las tareas de aprendizaje automático. Recordemos que en el aprendizaje automático (caso de separación por hiperplanos) nos enfrentamos al problema de minimizar el riesgo esperado

$$R(\beta) = \mathbb{E}_Z[\ell(\beta, Z)].$$

Hemos visto en el Capítulo 3 método basados en técnicas de optimización para minimización del riesgo empírico (o versiones penalizadas), donde minimizamos el riesgo empírico $R_{\mathcal{C}}(\beta)$ para el conjunto de observaciones $\mathcal{C} = \{z_1,\ldots,z_n\}$ como una "aproximación" a minimizar $R(\beta)$. El algoritmo del descenso de gradiente estocástico nos permite adoptar un enfoque diferente y buscar minimizar $R(\beta)$ directamente. Como no conocemos la distribución conjunta de $Z = (X, Y)$, no podemos simplemente calcular $\nabla R(\beta)$ y minimizarlo con el método del descenso del gradiente. Sin embargo, con el algoritmo del descenso del gradiente estocástico, todo lo que necesitamos, en cada paso, es generar un vector aleatorio que cumpla que $\mathbb{E}[V_t|\beta^{(t)}] \in \partial R(\beta^{(t)})$. Ahora veremos cómo se puede obtener fácilmente tal vector aleatorio.

Para simplificar, consideremos primero el caso de las funciones de pérdida ℓ diferenciables y, por tanto, la función de riesgo $R(\beta)$ también será diferenciable. Si tomamos el vector aleatorio $Z = (X, Y)$ y V_t igual al gradiente de $\ell(\beta^{(t)}, Z)$, por la linealidad del gradiente, tenemos

$$\mathbb{E}\left[V_t|\beta^{(t)}\right] = \mathbb{E}_Z\left[\nabla\ell(\beta^{(t)}, Z)\right] = \nabla\mathbb{E}_Z\left[\ell(\beta^{(t)}, Z)\right] = \nabla R(\beta^{(t)}). \tag{5.11}$$

Por tanto, la realización de V_t se obtiene muestreando una sola observación nueva desde la distribución conjunta de $Z = (X, Y)$, por ejemplo $Z = z$, y calculando el gradiente de $\ell(\beta, z)$ evaluado en $\beta = \beta^{(t)}$.

El mismo argumento vale para las funciones de pérdida no diferenciables. Simplemente dejamos que V_t sea un subgradiente de $\ell(\beta, Z)$ en $\beta^{(t)}$. Entonces, para cada vector $\mathbf{u}$ y realización $Z = z$ se cumple la siguiente desigualdad

$$\ell(\mathbf{u}, z) - \ell(\beta^{(t)}, z) \geq \langle \mathbf{u} - \beta^{(t)}, V_t\rangle.$$

Tomando esperanzas en ambos lados con respecto a Z y condicionadas al valor de $\beta^{(t)}$ obtenemos

$$\begin{aligned}R(\mathbf{u}) - R(\beta^{(t)}) &= \mathbb{E}[\ell(\mathbf{u}, z) - \ell(\beta^{(t)}, z)|\beta^{(t)}] \\ &\geq \mathbb{E}[\langle \mathbf{u} - \beta^{(t)}, V_t\rangle \mid \beta^{(t)}] \\ &= \langle \mathbf{u} - \beta^{(t)}, \mathbb{E}[V_t \mid \beta^{(t)}]\rangle.\end{aligned}$$

De todo esto podemos deducir que $\mathbb{E}[V_t \mid \beta^{(t)}]$ es un subgradiente de $R(\beta)$ en $\beta^{(t)}$, ya que cumple con la Definición 5.4.

Ahora usaremos nuestro análisis del algoritmo del descenso de gradiente estocástico para obtener un análisis de complejidad muestral para aprender problemas acotados Lipschitz convexos. La aplicación del Teorema 5.9 proporciona el siguiente corolario:

Corolario 5.10. *Consideremos un problema de aprendizaje acotado convexo y de Lipschitz con los parámetros ρ y B. Entonces, para cualquier $\varepsilon > 0$, si repetimos el método de descenso de gradiente estocástico para minimizar $R(\beta)$ un número de iteraciones*

$$T \geq \frac{B^2\rho^2}{\varepsilon^2}$$

con $\eta = \sqrt{\frac{B^2}{\rho^2 T}}$, entonces la salida del algoritmo de descenso de gradiente estocástico satisface

$$R(\bar{\beta}) \leq \min_{\beta} R(\beta) + \varepsilon.$$

Dado que estamos tratando con problemas de aprendizaje convexos en los que la función de pérdida es convexa, vamos a presentar un problema que es también convexo y se puede resolver usando el descenso del gradiente estocástico, como veremos a continuación. El método trata ahora de realizar la minimización en β mediante

$$\min_{\beta} \left(\frac{\lambda}{2}\|\beta\|^2 + R_{\mathcal{C}}(\beta) \right),$$

que ahora sí implica la perdida empírica $R_{\mathcal{C}}(\beta)$ y se adopta la habitual penalización en la norma al cuadrado de β

Para ello veamos primero la definición de una función λ-fuertemente convexa.

Definición 5.11. Una función f es λ-fuertemente convexa si para todos los vectores $\mathbf{u}$, β y constantes $\alpha \in (0, 1)$ se cumple

$$f(\alpha\beta + (1-\alpha)\mathbf{u}) \leq \alpha f(\beta) + (1-\alpha)f(\mathbf{u}) - \frac{\lambda}{2}\alpha(1-\alpha)\|\beta - \mathbf{u}\|^2.$$

Claramente, toda función convexa es 0-fuertemente convexa. A continuación mostramos un lema que será utilizado posteriormente.

Lema 5.12. *Se cumplen las siguientes implicaciones*

1. *La función $f(\beta) = \lambda\|\beta\|^2$ es 2λ-fuertemente convexa.*
2. *Si f es λ-fuertemente convexa y g es convexa, entonces $f + g$ es λ-fuertemente convexa.*
3. *Si f es λ-fuertemente convexa y $\mathbf{u}$ es un mínimo de f, entonces para cualquier β,*

$$f(\beta) - f(\mathbf{u}) \geq \frac{\lambda}{2}\|\mathbf{2} - \mathbf{u}\|^2.$$

Demostración. Los dos primeros puntos se obtienen directamente de la definición. Para probar el tercero, dividimos la definición de convexidad fuerte por α y reorganizamos los términos para obtener

$$\frac{f(\mathbf{u}+\alpha(\beta-\mathbf{u}))-f(\mathbf{u})}{\alpha} \leq f(\beta)-f(\mathbf{u})-\frac{\lambda}{2}(1-\alpha)\|\beta-\mathbf{u}\|^2.$$

Tomando el límite $\alpha \to 0$, obtenemos que el lado derecho de la ecuanción converge a $f(\beta)-f(\mathbf{u})-\frac{\lambda}{2}\|\beta-\mathbf{u}\|^2$. Por otro lado, el lado izquierdo se convierte en la derivada de la función $g(\alpha)=f(\mathbf{u}+\alpha(\beta-\mathbf{u}))$ en $\alpha=0$. Dado que $\mathbf{u}$ es un mínimo de f , se deduce que $\alpha=0$ es un mínimo de g, y por tanto el lado izquierdo de lo anterior tiende a cero cuando el límite $\alpha \to 0$, con lo que concluye nuestra demostración. □

A partir de la Definición 5.11, vamos a ver una variante del descenso del gradiente estocástico que tiene una tasa de convergencia más rápida para problemas en los que la función objetivo es fuertemente convexa. En este caso, el método de descenso de gradiente estocástico se puede modificar a funciones λ-fuertemente convexas:

- Para un $\eta > 0$ y $T \in \mathbb{N}$:
- Inicializar con $\beta^{(1)} = 0$
- Para $t = 1, ..., T-1$:
  - V_t vector aleatorio con $\mathbb{E}[V_t|\beta^{(t)}] \in \partial f(\beta^{(t)})$
  - $\eta_t = \frac{1}{\lambda \cdot t}$
  - $\beta_0^{(t+1)} = \beta^{(t)} - \eta_t V_t$
  - Actualizar $\beta^{(t+1)} = \arg\min_\beta \|\beta - \beta_0^{(t+1)}\|^2$ (paso de "proyección")
- Devolver $\bar{\beta} = \frac{\sum_{t=1}^{T} \beta^{(t)}}{T}$

Nos basamos en la siguiente proposición, que generaliza el Lema 5.12, con una demostración muy similar y que no será detallada.

Proposición 5.13. *Si f es una función λ-fuertemente convexa entonces para cualquier vector $\beta, \mathbf{u}$ y $\mathbf{v} \in \partial f(\beta)$ se cumple*

$$\langle \beta - \mathbf{u}, \mathbf{v} \rangle \geq f(\beta) - f(\mathbf{u}) + \frac{\lambda}{2}\|\beta - \mathbf{u}\|^2$$

La proposición anterior es usada para probar el siguiente resultado que extiende el Teorema 5.9 a este tipo de funciones λ-fuertemente convexas:

Teorema 5.14. *Supongamos que f es λ-fuertemente convexa y $\mathbb{E}[\|V_t\|^2] \leq \rho^2$. Sea $\beta^\star \in \arg\min_{\beta \in \mathcal{H}} f(\beta)$ una solución óptima. Entonces,*

$$\mathbb{E}[f(\bar{\beta})] - f(\beta^*) \leq \frac{\rho^2}{2\lambda T}(1 + \log(T))$$

Demostración. Por hipótesis, f es una función fuertemente convexa. Como $\mathbb{E}[V_t \mid \beta^{(t)}]$ está en el subgradiente de f en $\beta^{(t)}$, tenemos que

$$\langle \beta^{(t)} - \beta^\star, \mathbb{E}[V_t \mid \beta^{(t)}] \rangle \geq f(\beta^{(t)}) - f(\beta^\star) + \frac{\lambda}{2}\|\beta^{(t)} - \beta^\star\|^2. \tag{5.12}$$

Ahora veremos que

$$\langle \beta^{(t)} - \beta^*, \mathbb{E}[V_t \mid \beta^{(t)}] \rangle \leq \frac{\mathbb{E}[\|\beta^{(t)} - \beta^\star\|^2 - \|\beta^{(t+1)} - \beta^\star\|^2]}{2\eta_t} + \frac{\eta_t}{2}\rho^2 \tag{5.13}$$

Por la definición de $\beta_0^{(t)}$ tenemos que $\|\beta^{(t)} - \beta^\star\|^2 > \|\beta^{(t+1)} - \beta^\star\|^2$. Por lo tanto

$$\begin{aligned} \|\beta^{(t)} - \beta^\star\|^2 - \|\beta^{(t+1)} - \beta^\star\|^2 &\geq \|\beta^{(t)} - \beta^\star\|^2 - \|\beta_0^{(t)} - \beta^\star\|^2 \\ &= 2\eta_t \langle \beta^{(t)} - \beta^\star, V_t \rangle - \eta_t^2 \|V_t\|^2. \end{aligned} \tag{5.14}$$

Tomando la esperanza ambos lados, reordenando y usando la hipótesis $\mathbb{E}[\|V_t\|^2] \leq \rho^2$ se obtiene la ecuación (5.13). Comparando las ecuaciones (5.12) y (5.13) y sumando sobre t obtenemos

$$\begin{aligned} &\sum_{t=1}^{T} (\mathbb{E}[f(\beta^{(t)})] - f(\beta^\star)) \\ &\quad \leq \mathbb{E}\left[\sum_{t=1}^{T}\left(\frac{\|\beta^{(t)} - \beta^\star\|^2 - \|\beta^{(t+1)} - \beta^\star\|^2}{2\eta_t} - \frac{\lambda}{2}\|\beta^{(t)} - \beta^*\|^2\right)\right] + \frac{p^2}{2}\sum_{t=1}^{T}\eta_t. \end{aligned}$$

A continuación, usamos la definición $\eta_t = 1/(\lambda t)$ y observamos que la primera suma del lado derecho de la ecuación se reduce a $-\lambda T\|\beta^{(T+1)} - \beta^\star\|^2 \leq 0$. Por lo tanto,

$$\sum_{t=1}^{T} (\mathbb{E}[f(\beta^{(t)})] - f(\beta^*)) \leq \frac{\rho^2}{2\lambda}\sum_{t=1}^{T}\frac{1}{t} \leq \frac{\rho^2}{2\lambda}(1 + \log(T)).$$

El teorema se deriva de la desigualdad anterior, dividiendo por T y usando la desigualdad de Jensen. □

La función $f(\beta) = \frac{\lambda}{2}\|\beta\|^2 + R_{\mathcal{C}}(\beta)$ es una función λ-fuertemente convexa. Podemos pues, aplicar la variante del algoritmo del descenso de gradiente explicada anteriormente con $\beta \in \mathbb{R}^d$. Para aplicar este algoritmo, solo necesitamos encontrar una forma de construir una estimación insesgada de un subgradiente de f en $\beta^{(t)}$. Esto se hace tomando una observación $\mathbf{z}$ al azar en $\mathcal{C} = \{z_1, ..., z_n\}$ y elegir V_t tal que $V_t \in \partial\ell(\beta^{(t)}, \mathbf{z})$ lo que hace que se tenga que el valor esperado de $\lambda\beta^{(t)} + V_t$ sea un subgradiente de f en $\beta^{(t)}$.

El procedimiento de actualización sería el siguiente (el paso de "proyección no es necesario porque estamos trabajando con $\beta \in \mathbb{R}^d$):

$$\begin{aligned} \beta^{(t+1)} &= \beta^{(t)} - \frac{1}{\lambda t}\left(\lambda\beta^{(t)} + V_t\right) = \frac{t-1}{t}\beta^{(t)} - \frac{1}{\lambda t}V_t \\ &= \frac{t-1}{t}\left(\frac{t-2}{t-1}\beta^{(t-1)} - \frac{1}{\lambda(t-1)}V_{t-1}\right) - \frac{1}{\lambda t}V_t \\ &= -\frac{1}{\lambda t}\sum_{i=1}^{t} V_i. \end{aligned} \tag{5.15}$$

Una vez que hemos introducido esta filosofía de algoritmos de descenso de gradiente estocástico, estamos en condiciones de presentar el procedimiento de gradiente estocástico para ser aplicado en el caso del *Soft*-SVM:

- Para un $\eta > 0$ y $T \in \mathbb{N}$:
- Inicializar con $\theta^{(1)} = 0$
- Para $t = 1, ..., T-1$:
  - ◇ Se actualiza $\beta^{(t)} = \theta^{(t)}/(\lambda t)$
  - ◇ Se elije al azar una observación $(\mathbf{x}_i, y_i)$ del conjunto de entrenamiento $\mathcal{C}$
  - ◇ Si $y_i\langle\beta^{(t)}, \mathbf{x}_i\rangle < 1$ entonces $\theta^{(t+1)} = \theta^{(t)} + y_i\mathbf{x}_i$ y si $y_i\langle\beta^{(t)}, \mathbf{x}_i\rangle \geq 1$ entonces $\theta^{(t+1)} = \theta^{(t)}$
- Devolver $\bar{\beta} = \frac{\sum_{t=1}^{T}\beta^{(t)}}{T}$

Recordemos que queremos resolver el problema de minimización de la pérdida regularizada:

$$\min_{\beta}\left(\frac{\lambda}{2}\|\beta\|^2 + \frac{1}{n}\sum_{i=1}^{n}\max\{0, 1 - y_i\langle\beta, \mathbf{x}_i\rangle\}\right). \tag{5.16}$$

y que, a partir de (5.15), podemos reescribir la regla de actualización del algoritmo del descenso del gradiente estocástico como

$$\beta^{(t+1)} = -\frac{1}{\lambda t}\sum_{j=1}^{t} V_j,$$

donde V_j es un subgradiente de la función de pérdida cuando $\beta = \beta^{(j)}$ para la observación aleatoria $z = (\mathbf{x}, y)$ elegida de forma aleatoria del conjunto de entrenamiento $\mathcal{C}$ en la iteración j, es decir, $V_j \in \partial\ell(\beta^{(j)}, \mathbf{z})$. Para la función de pérdida ℓ tipo hinge considerada en (5.16), si $z = (\mathbf{x}, y)$ es la observación elegida al azar del conjunto de entrenamiento entonces podemos tomar V_j que sea igual a 0 si $y\langle\beta^{(j)}, \mathbf{x}\rangle \geq 1$ y $V_j = y\mathbf{x}$ si $y\langle\beta^{(j)}, \mathbf{x}\rangle < 1$.

Capítulo 6

Implementación práctica y ejemplos

En la actualidad existe una gran diversidad de repositorios de datos abiertos donde se pueden obtener conjuntos de datos tanto de clasificación binaria y multiclase como de regresión. Entre ellos los más destacados son el UCI Machine Learning Repository [8], Kaggle [9], y también el paquete de R **mlbench**, que contiene numerosos conjuntos de datos para probar las distintas técnicas de Aprendizaje Automático.

La mayor parte del software que existe para aplicar SVM está escrito en C o C++, como la librería **libsvm** [7], que proporciona una implementación rápida y robusta de los SVM y da resultados muy competentes en la mayoría de problemas de clasificación y regresión. También destacan las librerías **SVMlight, SVMTorch, Royal Holloway Support Vector Machines, mySVM** y **M-SVM**. También se pueden usar *toolboxs* en MATLAB como **The MathWorks, SVM and Kernel Methods Matlab Toolbox o MATLAB Support Vector Machine Toolbox** y la **SVM toolbox for Matlab**.

Para comprobar el funcionamiento práctico de los algoritmos tratados a lo largo del trabajo, vamos a considerar un par de conjuntos de datos de clasificación binaria obtenidos de estos repositorios, y utilizaremos el programa R, en el cual podremos realizar tanto la clasificación, como un estudio descriptivo de los datos.

R tiene diferentes librerías con las funciones necesarias para implementar diversas formulaciones del SVM con la posibilidad de usar diferentes tipos de kernels y técnicas como la validación cruzada que sirven para conseguir la mejor elección de los parámetros del kernel. A continuación vamos a describir por encima las distintas funcionalidades de cada una de ellas [10, 11]:

1. **e1071**, es una de las librerías más populares para SVM en R. Ofrece una amplia gama de opciones de kernel, como lineal, polinómico, radial y sigmoidal. Además, permite la selección de hiperparámetros mediante validación cruzada. Sin embargo, puede ser lenta en conjuntos de datos grandes.

2. **kernlab** es otra librería popular de SVM en R. También ofrece una variedad de opciones de kernel, incluyendo kernels personalizados. Una ventaja de esta librería es que es eficiente en grandes conjuntos de datos, ya que utiliza técnicas de reducción de dimensionalidad. Sin embargo, la selección de hiperparámetros puede ser un poco más difícil en comparación con otras librerías.

3. **LiblineaR** aplica el SVM lineal para clasificación binaria y cuenta con un enfoque eficiente y rápido para grandes conjuntos de datos. Es fácil de usar y ofrece una

opción de selección de parámetros basada en la validación cruzada. Sin embargo, no tiene la misma flexibilidad que otras librerías en cuanto a opciones de kernel y parámetros.

4. **LibSVM**, es una librería muy popular y ampliamente utilizada que ofrece un conjunto completo de funciones para SVM. Es conocida por ser rápida y eficiente, especialmente en conjuntos de datos grandes. Además, permite la selección de hiperparámetros mediante validación cruzada y cuenta con una gran cantidad de opciones de kernel. La principal desventaja es que puede ser difícil de usar para aquellos que no tienen experiencia en SVM.

5. `mlr3learners` es parte del paquete **mlr3**, que es una plataforma de aprendizaje automático completa en R. `mlr3learners` ofrece una amplia gama de modelos de aprendizaje automático, incluyendo SVM. Una ventaja de esta librería es que se integra bien con otras herramientas de **mlr3**, como la selección automática de modelos y la validación cruzada. La desventaja es que puede ser más compleja para usuarios principiantes en R.

6. **quanteda.dictionaries** es una librería que se enfoca principalmente en el procesamiento de lenguaje natural, pero también incluye funciones para SVM. Es una buena opción para trabajar en problemas de clasificación de texto o lenguaje natural. La principal desventaja es que es menos flexible en cuanto a opciones de kernel y parámetros.

La elección de una librería u otra depende de las necesidades específicas y del conjunto de datos con el que se esté trabajando. El uso de conjuntos de datos grandes hace que algunas librerías no sean muy interesantes por ser más lentas, aunque esas librerías pueden proporcionar alternativamente más flexibilidad o facilitan la selección de parámetros.

Veamos algunas aplicaciones prácticas del SVM. Para ello vamos a comenzar utilizando un conjunto de datos del campo de la informática que daremos contexto a continuación obtenido de [8] y basado en detectar *spam* en correo electrónico.

El correo electrónico es la forma más rápida de intercambiar mensajes de un lugar a otro en todo el mundo, el mayor uso del correo electrónico llevó a aumentar los mensajes recibidos en el buzón, donde el destinatario recibe muchos, incluidos aquellos que causan problemas importantes y diferentes, como el robo de la identidad del destinatario, la pérdida de información esencial que causa pérdidas a las empresas además del daño a la red. Estos mensajes son tan peligrosos que el usuario no puede evitarlos, especialmente porque toman diferentes formas, como por ejemplo anuncios. Estos mensajes se conocen como mensajes no deseados o de "spam". Para eliminar estos mensajes de spam y evitar que se acceda a ellos, se utiliza un filtrado a partir de técnicas de aprendizaje automático.

Nuestro objetivo es a partir de este conjunto de datos construir un SVM que indique con la mayor certeza posible si un e-mail es spam. Para ello, utilizaremos la librería **e1071** disponible en R. Comenzaremos realizando una partición del conjunto de datos inicial, en dos subconjuntos. Uno será el conjunto de entrenamiento, con el que ajustaremos el SVM y que contendrá entorno al 70 % de los datos. Este 70 % de los datos para entrenar el SVM los vamos a seleccionar al azar. Por otro lado tenemos el conjunto de test con el que probaremos la calidad del SVM ajusta con el 30 % de datos restantes.

El conjunto de datos corresponde a una base de datos de correo electrónico donde se clasifican dichos correos como no deseados o no deseados. Consta de 4601 registros de los cuales 1813 corresponden a spam y los otros 2788 no son spam. Se consideran $p = 57$ variables que, además del tipo de clase de correo electrónico, representan la frecuencia de aparición en el correo de ciertas "palabras" o "caracteres" frecuentes en un correo electrónico. A continuación listamos de manera agrupada las características que aparecen en el conjunto de datos:

- Las 48 primeras características son números reales continuos. El rango es de $[0 - 100]$ donde "word_freq_Word" es el porcentaje de palabras que aparecen en el correo electrónico y corresponden a la palabra "Word" del total del correo electrónico. Una "palabra" en este caso es cualquier cadena de caracteres alfanuméricos limitada por caracteres no alfanuméricos o un signo ortográfico de puntuación como final de cadena.

- Las siguientes 6 características son números reales continuos cuyo rango es de $[0 - 100]$ y son del tipo "char_freq_Char" que son el porcentaje de caracteres que aparecen en el correo electrónico y corresponden a ese signo ortográfico de puntuación "Char".

- La característica 55 es de tipo real continuo "capital_run_length_average" y proporciona la longitud promedio de secuencias ininterrumpidas de letras mayúsculas.

- La característica 56 es de tipo entero continuo "capital_run_length_longest" y define la longitud máxima de la secuencia ininterrumpida de letras mayúsculas.

- La característica 57 es de tipo entero continuo "capital_run_length_total" y define la suma de las longitudes de las secuencias ininterrumpidas de letras mayúsculas, es decir, el número total de letras mayúsculas del correo electrónico.

- La última columna en realidad no es una característica. Es la clase a la que corresponde, donde se indica si el correo electrónico es considerado spam 1 o no 0.

Las funciones que utilizamos en R para leer el conjunto de datos y dividirlo en dos, "train" y "test", son las siguientes, donde hemos fijado una semilla para garantizar la reproducibilidad de los resultados obtenidos:

```
load("C:/Users/Usuario/Downloads/Spambase.RData")
dataset<-spambase
View(dataset)
index <- 1:nrow(dataset)
set.seed(1997)
trainindex <- sample(index, trunc(length(index)*70/100))
trainset <- dataset[trainindex,]
testset <- dataset[-trainindex,]
```

Para encontrar los parámetros óptimos, tanto del kernel como el parámetro de coste C, vamos a utilizar las técnicas de validación cruzada, en nuestro caso con $m = 10$ particiones (m-fold). La función `tune.svm` crea un rejilla de combinaciones de parámetros

del SVM. Para cada una de las combinaciones de parámetros en dicha rejilla, se divide el conjunto de datos "trainset" en m subgrupos de observaciones. Se ajustan m modelos de SVM considerando como conjuntos de entrenamiento incluyendo $m-1$ subgrupos de observaciones y se prueba la efectividad de dicho SVM con el m-ésimo subconjunto de datos que habíamos dejado fuera. Este proceso se repite m veces dejando cada vez fuera uno de esos m subconjuntos de datos en los que habíamos participado el total de datos. El error de la validación cruzada m-fold para esa combinación de parámetros en la rejilla se halla realizando la media aritmética de los errores de los m ajustes realizados. La salida que proporciona `tune.svm` es la combinación de parámetros, kernel y coste, que proporcionan el mínimo error promediado.

En este caso, vamos a estudiar la calidad del clasificador sobre tres tipos de kernels distintos vistos en el Capítulo 4, el kernel polinómico, el kernel radial y el kernel sigmoidal.

Aplicamos `tune.svm` sobre "trainset" para obtener los parámetros óptimos para cada uno de ellos y dejamos sin utilizar el "testset" que nos servirá, después, como conjunto para probar el funcionamiento de SVM ajustados con esos parámetros óptimos. Las funciones de R que aplicamos en este caso son:

```
#descargamos el paquete e1071 y lo instalamos
install.packages('e1071', dependencies = TRUE)
library(e1071)

#Kernel radial
set.seed(1997)
tRadial <- tune.svm(is_spam~., data = trainset, gamma = 10^(-4:2),
                    cost =10^(-3:2), kernel="radial")
summary(tRadial)

#Kernel sigmoidal
set.seed(1997)
tSigmo <- tune.svm(is_spam~., data = trainset, gamma = 10^(-3:1),
                   cost =10^(-1:1), coef0=(-2:1), kernel="sigmoid")
summary(tSigmo)

#Kernel polinomico
set.seed(1997)
tPoli <- tune.svm(is_spam~., data = trainset, gamma = 10^(-2:0),
                  cost =10^(0:1), coef0=(-1:1),
                  degree=(0:3),kernel="polynomial")
summary(tPoli)
```

Mediante este código, obtenemos la siguiente tabla con los parámetros óptimos para cada tipo de kernel utilizado. En R se denotan los parámetros de la siguiente forma γ como `gamma`, C como `cost`, τ como `coef0` y d como `degree`.

Kernel	**Error óptimo**	γ	C	τ	d
Radial	0.07453	0.01	1	-	-
Sigmoidal	0.08602	0.01	10	-2	-
Polinómico	0.06894	0.01	10	1	2

Tabla 6.1: Resumen parámetros óptimos de los Kernels utilizados

A continuación, usamos la función `svm` para ajustar la regla óptima para cada kernel:

```
modelKRad <- svm(is_spam~., data = trainset, kernel = "radial",
                  gamma = 0.01, cost = 1)

modelKSig <- svm(is_spam~., data = trainset, kernel = "sigmoid",
                  tau = -2, cost = 10)

modelKPol <- svm(is_spam~., data = trainset, kernel = "polynomial",
                  gamma = 0.01 , cost = 10, coef0=1, degree=2)
```

Para poder comparar la calidad de cada modelo generado no usamos que tal funciona el ajuste en los datos "trainset", lo que daría lugar a solo una estimación del error *aparente*, que suele ser optimista y que suele premiar el sobreajuste de la regla de clasificación. En su lugar, usamos las funciones `predict` y `table` sobre el conjunto de datos "testset", que habíamos dejado fuera, y de esta forma obtenemos estimaciones del error de generalización más correctas que las obtenidas por los errores aparentes.

```
predKRad <- predict(modelKRad, testset[,-58],probability = TRUE)
tabKRad <- table(pred = predKRad, true = testset[,58])

predKSig <- predict(modelKSig, testset[,-58],probability = TRUE)
tabKSig <- table(pred = predKSig, true = testset[,58])

predKPol <- predict(modelKPol, testset[,-58],probability = TRUE)
tabKPol <- table(pred = predKPol, true = testset[,58])
```

Esto crea las matrices de confusión que van a ser usadas para ver el funcionamiento de los métodos aplicados. Nótese que quitamos la columna `58` que es la que contiene la información "real" sobre ser spam o no

Las medidas de evaluación se definen a partir de esas matrices de confusión de la siguiente forma:

- *Verdaderos positivos* (TP): el correo electrónico es catalogado como spam y en la clase verdadera también es spam.
- *Verdaderos negativos* (TN): el correo electrónico es catalogado como no es spam, y en la clase verdadera no lo es tampoco.
- *Falsos positivos* (FP): correo electrónico que no es spam en la clase real pero que es catalogado como spam.

- *Falsos negativos* (FN): correo electrónico que es spam en clase verdadera pero que no es catalogado como spam.

	Clase Verdadera	
Clase en que se cataloga	Spam	No Spam
Spam	TP	FP
No Spam	FN	TN

Tabla 6.2: Matriz de Confusión

A partir de esta matriz de confusión mostrada en la Figura 6.2, se pueden definir las siguientes métricas que nos sirven para conocer cómo funcionan nuestros clasificadores según nuestros objetivos:

- La *sensibilidad*, también conocida como "Tasa de Verdaderos Positivos" o "recall" es la proporción de observaciones de spam que el modelo ha identificado correctamente como spam en relación con todas las instancias de spam reales:

$$\text{Sensibilidad} = \frac{\text{TP}}{\text{TP+FN}}$$

- La *especificidad* representa la proporción de observaciones que no son spam que el modelo ha identificado correctamente como no spam en relación con todas las observaciones que no son spam reales:

$$\text{Especificidad} = \frac{\text{TN}}{\text{TN+FP}}$$

- La *precisión* es la proporción de observaciones clasificadas correctamente como spam en relación con todas las observaciones clasificadas como spam por el modelo:

$$\text{Precisión} = \frac{\text{TP}}{\text{TP+FP}}$$

La sensibilidad se centra en la capacidad del modelo para detectar positivos (spam), mientras que la especificidad se enfoca en la capacidad del modelo para detectar negativos (no spam). La precisión se refiere a la exactitud de las clasificaciones positivas realizadas por el modelo. En este ejemplo, donde seguramente no deseamos que un correo importante sea determinado como spam, a precio de que nos deje sin filtrar algún correo de spam, seguro que nos interesa una especificidad alta.

Para nuestro conjunto de datos, podemos observar en la Tabla 6.3 un resumen de las principales métricas que se han obtenido a partir de la matriz de confusión para cada uno de los tres modelos.

```
show(tabKRad)
    true
pred   0   1
   0 773  58
   1  31 519

show(tabKSig)
    true
pred   0   1
   0 773  67
   1  31 510

show(tabKPol)
    true
pred   0   1
   0 767  48
   1  37 529
```

Modelo	**Sensibilidad**	**Especificidad**	**Precisión**
Kernel Radial	0.89948	0.96144	0.94363
Kernel Sigmoidal	0.88388	0.96144	0.94269
Kernel Polinómico	0.91681	0.95398	0.93462

Tabla 6.3: Métricas de rendimiento de los clasificadores

Para escoger el mejor clasificador para nuestro conjunto de datos, como hemos comentado anteriormente, nos basamos en aquel cuya especificidad sea mayor. Como para el kernel radial y el sigmoidal tienen la misma, vemos que el radial tiene mayor sensibilidad y precisión y en consecuencia es mejor clasificador.

A continuación mostraremos un segundo ejemplo, usando también la librería **e1071** en R, en un problema de mayor dimensionalidad $p = 256$. Con este fin, usaremos las imagenes correspondientes a los dígitos "5" y "8" escritos a mano incluidos en el conjunto "USPS" del repositorio UCI [8]. Este conjunto de datos se creó mediante el escaneo automático de sobres por parte del Servicio Postal de EE. UU. y muestra una amplia gama de estilos de fuente. Así, contaremos con un conjunto de datos `X58` que contiene 556 dígitos que son etiquetados como "5" y 542 dígitos etiquetados como "8". Estas imágenes contienen 16×16 píxeles, lo que nos lleva a $p = 16 \times 16 = 256$ variables x_1, x_2, ..., x_{256} que miden la intensidad de la imagen en ese píxel. La Figura 6.1 muestra 4 de estas imágenes, dos correspondientes al dígito "5" y dos al correspondientes al dígito "8". La variable `cls` de este conjunto `X58` de datos nos proporciona la etiqueta (dígito) al que estaba asignada esa imagen.

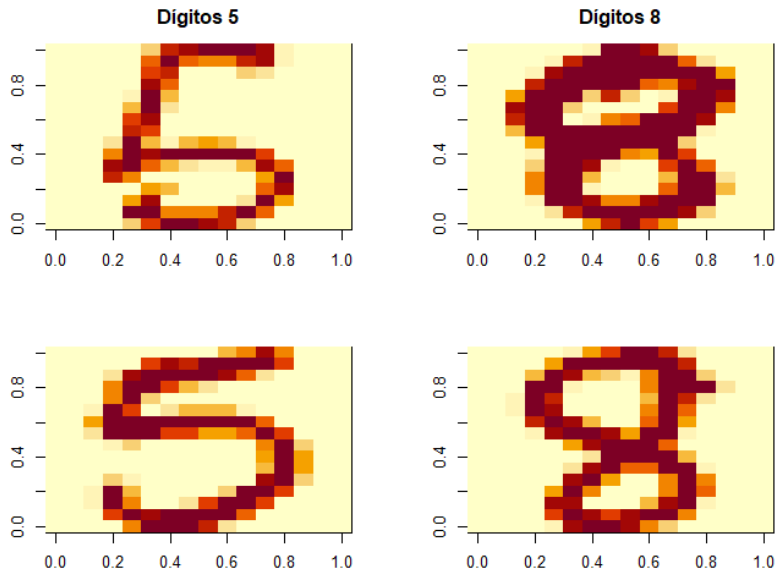

Figura 6.1: Ejemplos de las imágenes consideradas en el conjunto de datos "USPS".

En este ejemplo, nos centraremos en usar kernels tipo "bases radiales" y, primeramente, eligiéremos los parámetros óptimos mediante la función `tune.svm`. Tras unas pruebas iniciales, acotamos nuestra búsqueda de parámetros en una rejilla de valores que se obtienen al cruzar valores de γ entre 10^{-5}, 10^{-4}, ..., y 10^{-1} y valores del coste C entre 10^{-1}, 10^{0}, 10^{1} y10^{2}. El código utilizado es el siguiente:

```
tobj <- tune.svm(cls~.,data=X58,kernel='radial',
                      gamma=10^(-5:-1), cost=10^(-1:2))
summary(tobj)
```

Obtenemos la siguientes salida numérica:

```
Parameter tuning of 'svm':

- sampling method: 10-fold cross validation

- best parameters:
 gamma cost
 0.001   10

- best performance: 0.01642202

- Detailed performance results:
   gamma  cost      error dispersion
1  1e-05   0.1 0.50816514 0.04439007
2  1e-04   0.1 0.10479566 0.05208055
3  1e-03   0.1 0.03185154 0.01972968
4  1e-02   0.1 0.11384487 0.03045004
```

```
5  1e-01   0.1 0.50816514 0.04439007
6  1e-05   1.0 0.08928274 0.03861044
7  1e-04   1.0 0.03004170 0.01604535
8  1e-03   1.0 0.01823186 0.01289617
9  1e-02   1.0 0.04917431 0.02579808
10 1e-01   1.0 0.47804837 0.07198567
11 1e-05  10.0 0.03185988 0.01878258
12 1e-04  10.0 0.01914095 0.01574600
13 1e-03  10.0 0.01642202 0.01349394
14 1e-02  10.0 0.04552961 0.02461888
15 1e-01  10.0 0.46167640 0.07105138
16 1e-05 100.0 0.02277731 0.01501645
17 1e-04 100.0 0.02188490 0.01376606
18 1e-03 100.0 0.01914095 0.01516277
19 1e-02 100.0 0.04552961 0.02461888
20 1e-01 100.0 0.46167640 0.07105138
```

Usando elección de parámetros por 10-fold (la segunda columna de la tabla anterior muestra los estimadores del error de generalización obtenidos al promediar los 10 errores en cada una de los 10 subconjuntos de datos que considera 10-fold y la última columna un estimador de la desviación típica de esos estimadores) vemos que unos parámetros bastante razonables para aplicar SVM con bases radiales son $\gamma = 10^{-3} = 0.001$ y $C = 10$. Para estos valores de γ y C se alcanza un error de generalización estimado de 0.0164. Es decir, esperaríamos equivocarnos, de forma automatizada, sobre un 1.64 % de las imágenes, lo que no está nada mal puesto que algunas imágenes son ciertamente complejas de ser clasificadas correctamente por un humano, por usar escrituras de los dígitos bastante extrañas.

Este resultado del `tune.svm` se puede representar gráficamente mediante:

```
plot(tobj,transform.x = log10, xlab = expression(log[10](gamma)),
       transform.y = log10, ylab = expression(log[10](C)),
       color.palette = topo.colors)
```

El gráfico obtenido puede verse en la Figura 6.2, donde se aprecia bastante bien como una correcta elección de parámetros γ y C afecta notablemente al funcionamiento del SVM, pasando de tasas de error próximas al 1 % (como la óptima anteriormente comentada) a tasas de error próximas al 50 % (casi equivalentes a asignar dígitos al azar en este ejemplo).

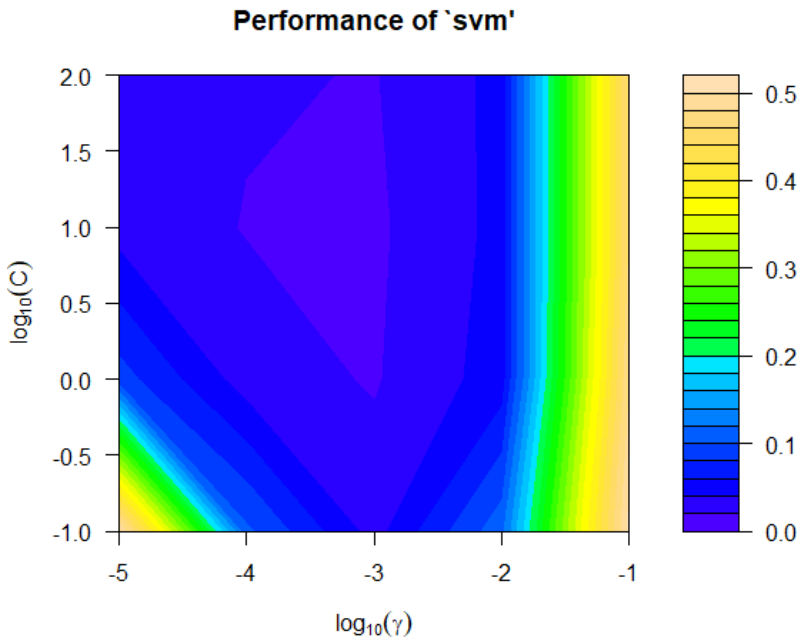

Figura 6.2: Resultado del `tune.svm` mostrando los errores 10-fold para distintas combinaciones de parámetros al usar kernels con "bases radiales" y justificando la elección $\gamma = 10^{-3} = 0.001$ y $C = 10$.

Capítulo 7

Algunas extensiones

Los SVM, como hemos visto en los capítulos anteriores, son un algoritmo muy utilizado de aprendizaje supervisado para resolver problemas de clasificación binaria. Sin embargo, a lo largo de los años, se han desarrollado varias extensiones del SVM para abordar diferentes problemas de aprendizaje automático.

Una de las extensiones más comunes es la resolución del problema de clasificación multiclase. En lugar de clasificar entre dos clases, se trata de clasificar entre tres o más clases. Hay varias formas de abordar este problema, entre las que se incluyen la técnica "one-versus-all" y la técnica "one-versus-one".

Otra aplicación interesante de estos es el problema de "caracterización" de texto, que consiste en clasificar objetos que se representan mediante cadenas de caracteres, como secuencias de ADN, palabras o frases. En este caso, los SVM se utilizan en combinación con funciones kernel especiales para trabajar directamente con las secuencias de caracteres.

Otra extensión tratada en muchos libros es la resolución del problema de regresión, donde el objetivo es predecir una variable continua en lugar de una variable categórica. Para ello, se utiliza una variante llamada SVM de regresión.

En este capítulo comentaremos un poco por encima todas estas variantes junto con algunos métodos de optimización para el cálculo de los SVM como alternativas al algoritmo del gradiente estocástico visto en el Capítulo 5.

Problema SVM Multiclase

Hasta ahora, los problemas que hemos presentado a lo largo de este trabajo son problemas binarios, es decir, tenemos únicamente dos clases en las que se clasifican. Hay ocasiones en las que los datos pueden clasificarse en más de dos clases. Este es el caso de los problemas multiclase, donde $\mathbf{x}_i \in \mathbb{R}^p$ es el vector de variables predictoras e $y_i \in \{1, 2, \ldots, K\}$ la etiqueta de clase, siendo K el número de posibles clases. Ya que los clasificadores SVM expuestos en los capítulos anteriores están formulados solo para dos clases, necesitamos saber cómo la metodología SVM puede extenderse para distinguir entre $K > 2$ clases. Para ello, se ha intentado definir el SVM multiclase de varias formas siempre tratando de reducir este problema a una serie de problemas binarios. Trataremos por encima dos de esos enfoques.

El primero de ellos es el *One-Versus-All*, "uno contra el resto", que divide el problema de clase K en K subproblemas del tipo "clase k-ésima" frente a "clase no k-ésima", con

$k = 1, 2, \ldots, K$. Tomando como referencia el subproblema k-ésimo, se construye la función f_k en el que la clase k-ésima se codifica como positiva y la unión de las demás clases se codifica como negativa en la clasificación. Una nueva observación $\mathbf{x} \in \mathbb{R}^p$ se asignará a la clase con el mayor valor de $f_k(\mathbf{x})$, con $k = 1, 2, \ldots, K$, siendo $f_k(\mathbf{x})$ la solución SVM óptima para el problema binario de la k-ésima clase frente al resto.

El otro enfoque es el que se conoce como *One-Versus-One*, "uno contra uno", que divide el problema de clase K en $\binom{K}{2}$ problemas, que son todas las posibles comparaciones de los pares de clases. En este caso se construye una función clasificadora f_{jk} codificando la "clase j-ésima" como positiva y la "clase k-ésima" como negativa, con $j, k = 1, 2, \ldots, K$ y $j \neq k$. Para una nueva observación $\mathbf{x} \in \mathbb{R}^p$ se le asignará la clase que tenga el mayor número de votos evaluando en todas las funciones clasificadoras f_{jk} y sumando dichos votos.

Aunque este tipo de enfoques se utilizan ampliamente en la práctica para resolver problemas de SVM multiclase, se debe tener cuidado con su uso.

El problema de "uno contra el resto" es popular para llevar a cabo tareas de categorización de texto, donde cada documento puede pertenecer a más de una clase. Esto es un tipo de problema que comentaremos brevemente a continuación. Este tipo de enfoque permite optimizar el método del SVM para cada subproblema binario, pero puede generar una clasificación diferente a otros clasificadores utilizados para el caso de multiclase. El éxito de la clasificación del enfoque de "uno contra el resto" depende del grado de desequilibrio del tamaño de clase de cada subproblema y de si una clase domina a todas las demás clases al determinar la clase más probable para cada nueva observación $\mathbf{x}$.

El enfoque de "uno contra uno", que usa solo aquellas observaciones que pertenecen a las clases involucradas en cada comparación por pares, tiene el problema de tener que usar muestras más pequeñas para entrenar a cada clasificador, lo que, a su vez, puede aumentar la variabilidad de la solución.

Problema de Categorización de Texto con SVM

La categorización de texto es la asignación de documentos de texto (o hipertexto) en lenguaje natural a un número determinado de categorías predefinidas en función del contenido de dichos documentos. Aunque la categorización manual de documentos de texto es actualmente el sentido común, por ejemplo, usar carpetas para guardar archivos, mensajes de correo electrónico, direcciones URL, etc, algunas categorizaciones/clasificación de texto están automatizadas, como es el caso, como hemos visto en el Capítulo 6, de filtros de spam o correo no deseado para ayudar a los usuarios a lidiar con el gran volumen de mensajes de correo electrónico diarios. Para reducir los costos de las tareas de categorización de texto, se espera un mayor grado de automatización en el futuro. Para estos problemas de categorización de texto, se han propuesto durante estos años lo que se conoce como *kernels de cadenas*, que vamos a ver a continuación sin entrar mucho en detalle. Para ello, comencemos definiendo los siguientes conceptos.

Sea $\mathcal{A}$ un alfabeto finito, diremos que una *cadena*

$$s = s_1 s_2 \cdots s_{|s|},$$

es una secuencia finita de elementos de $\mathcal{A}$, incluyendo la secuencia vacía y donde $|s|$ denota la longitud de s. Llamamos a u una subsecuencia de s, que la denotamos como

$u = s(i)$, si hay índices $i = (i_1, i_2, \ldots, i_{|u|})$, con $1 \leq i_1 < \cdots < i_{|u|} \leq |s|$, tal que $u_j = s_{i_j}, j = 1, 2, \ldots, |u|$. Si los índices i son contiguos, decimos que u es una *subcadena* de s. La longitud de u en s es

$$\ell(\mathrm{i}) = i_{|u|} - i_1 + 1,$$

que es el número de elementos de s superpuestos por la subsecuencia u.

Por ejemplo, sea s la cadena "SVM" ($s_1 = \mathrm{S}$, $s_2 = \mathrm{V}$, $s_3 = \mathrm{M}$, $|s| = 3$), y considerense todas las posibles secuencias de 2 símbolos, "SV", "SM" y "VM", derivadas de s. Para la cadena $u = \mathrm{SV}$, tenemos que $u_1 = \mathrm{S} = s_1, u_2 = \mathrm{V} = s_2$, de donde, $u = s(i)$, donde $i = (i_1, i_2) = (1, 2)$. Así, $\ell(i) = 2$. De manera similar, para la subsecuencia $u = \mathrm{SM}$, se tiene que $u_1 = \mathrm{S} = s_1, u_2 = \mathrm{M} = s_3$, de donde, $i = (i_1, i_2) = (1, 3)$, y $\ell(i) = 3$. Además, la subsecuencia $u = \mathrm{MV}$ tiene $u_1 = \mathrm{M} = s_2, u_2 = \mathrm{V} = s_3$, de donde, $i = (2, 3)$, y $\ell(i) = 2$.

Si $D = \mathcal{A}^m$ es el conjunto de todas las cadenas finitas de longitud como máximo m del lenguaje definido $\mathcal{A}$, entonces, el espacio de características para un kernel de cadena es $\mathbb{R}^D$. La transformación de las características Φ_u, que opera en una cadena $s \in \mathcal{A}^m$, se caracteriza en términos de un cadena dada $u \in \mathcal{A}^m$. Para tratar con subsecuencias no contiguas, definimos $\lambda \in (0, 1)$ como la *tasa de disminución*, o factor de reducción, y lo usamos para ponderar los huecos interiores en las subsecuencias. El grado de importancia que le damos a una subsecuencia contigua se refleja en cuanto de pequeño tomamos el valor de λ. El valor $\Phi_u(s)$ se calcula de la siguiente manera: hay que identificar todas las subsecuencias (indexadas por i) de s que son idénticas a u; para cada una de esas subsecuencias, se eleva λ a la potencia $\ell(i)$; y luego se suman los resultados sobre todas las subsecuencias. Debido a que $\lambda < 1$, los valores más grandes de $\ell(i)$ tienen menos peso que los valores más pequeños de $\ell(i)$. Esto se escribe como

$$\Phi_u(s) = \sum_{\mathrm{i}:u=s(i)} \lambda^{\ell(\mathbf{i})}, \quad u \in \mathcal{A}^m. \tag{7.1}$$

Considerando el ejemplo anterior, tenemos que $\Phi_{\mathrm{SV}}(\mathrm{SVM}) = \lambda^2$, $\Phi_{\mathrm{SM}}(\mathrm{SVM}) = \lambda^3$ y $\Phi_{\mathrm{VM}}(\mathrm{SVM}) = \lambda^2$.

Dos documentos se consideran "similares" si tienen muchas subsecuencias en común, es decir, cuantas más subsecuencias tengan en común, más similares se considerarán. Hay que tener en cuenta que el grado de contigüidad presente en una subsecuencia determina el peso de esa subcadena en la comparación; cuanto más cerca esté la subsecuencia de una subcadena contigua, más debería contribuir a la comparación.

Sean s y t dos cadenas. El kernel asociado con la transformación de características correspondientes a s y t viene dado por la suma de los productos internos para todas las subcadenas comunes de longitud m,

$$\begin{aligned} K_m(s, t) &= \sum_{u \in \mathrm{D}} \langle \Phi_u(s), \Phi_u(t) \rangle \\ &= \sum_{u \in \mathrm{D}} \sum_{i:u=s(i)} \sum_{j:u=s(j)} \lambda^{\ell(i)+\ell(j)}. \end{aligned} \tag{7.2}$$

El kernel (7.2) se denomina *kernel de cadena*, o *kernel de subsecuencias ponderadas por intervalos*. Para el ejemplo que hemos visto, sea t la cadena "SVA" ($t_1 = \mathrm{S}, t_2 = \mathrm{V}, t_3 = \mathrm{A}, |t| = 3$). Tengase en cuenta que las cadenas "SVM" y "SVA" son subcadenas de la cadena "Support Vector MAchines".

Las tres subcadenas de 2 elementos de t son "SVA","SA","VA". Para estas subcadenas, tenemos que $\Phi_{\mathrm{SV}}(\mathrm{SVA}) = \lambda^2$, $\Phi_{\mathrm{SA}}(\mathrm{SVA}) = \lambda^3$ y $\Phi_{\mathrm{VA}}(\mathrm{SVA}) = \lambda^2$. El producto interior (7.1) de las dos cadenas viene dado por $K_2(SVM, SVA) = \langle \Phi_{\mathrm{SV}}(\mathrm{SVM}), \Phi_{\mathrm{SV}}(\mathrm{SVA}) \rangle = \lambda^4$.

El mapeo de características en el espacio de características generalmente se normalizan para eliminar cualquier sesgo introducido por la longitud del documento. Esto es equivalente a normalizar el kernel (7.2),

$$K_m^*(s,t) = \frac{K_m(s,t)}{\sqrt{K_m(s,s)K_m(t,t)}}.$$

Para nuestro ejemplo, por la definición de (7.2) tenemos que $K_2(\mathrm{SVM}, \mathrm{SVM}) = \langle \Phi_{\mathrm{SV}}(\mathrm{SVM}), \Phi_{\mathrm{SV}}(\mathrm{SVM}) \rangle + \langle \Phi_{\mathrm{SM}}(\mathrm{SVM}), \Phi_{\mathrm{SM}}(\mathrm{SVM}) \rangle + \langle \Phi_{\mathrm{VM}}(\mathrm{SVM}), \Phi_{\mathrm{VM}}(\mathrm{SVM}) \rangle = \lambda^6 + 2\lambda^4$, y, de manera similar, $K_2(\mathrm{SVA}, \mathrm{SVA}) = \lambda^6 + 2\lambda^4$. Por tanto se tiene $K_2^*(\mathrm{SVM}, \mathrm{SVA}) = \lambda^4/(\lambda^6 + 2\lambda^4) = 1/(\lambda^2 + 2)$.

Problema SVM de Regresión

Dada la eficacia demostrada de los support vector machine en el ámbito de la clasificación, surge la pregunta de por qué no aprovechar esta teoría y conocimientos en otros contextos como la regresión. La técnica de los vectores de soporte se presenta como una herramienta versátil para resolver problemas de estimación de funciones en problemas de alta dimensionalidad. Debido a los buenos resultados alcanzados para la clasificación, se plantea la posibilidad de aplicar este enfoque basado en conceptos análogos a los vistos en los capítulos anteriores para abordar problemas de regresión.

Para ello, tenemos que buscar el hiperplano regresor que mejor se ajuste al conjunto de datos de entrenamiento $\mathcal{C}$ y no existe un conjunto de clases en las que queremos clasificar. En la clasificación SVM, el margen se utiliza para determinar la cantidad de separación entre dos clases de puntos que no se superponen: cuanto mayor sea el margen, más confianza tenemos en que el hiperplano de separación óptimo es un clasificador superior. En la regresión, no estamos interesados en separar puntos, sino en proporcionar una función de los vectores de entrada que sigan más de cerca los datos observados. Por lo tanto, en la regresión, la idea se basa en considerar una distancia margen ε, de tal forma que todas las observaciones han de encontrarse dentro de una *banda* o *tubo* de nuestro hiperplano, es decir, que se encuentren a una distancia menor de ε del hiperplano. Para definir este hiperplano sólo se tiene en consideración aquellas observaciones que disten más de ε del hiperplano, que se describirán a través de variables de holgura. Estos será los considerados como vectores soporte.

Para formular estas ideas, hay que definir una función de pérdida apropiada. La función de perdida que se define ignora los errores asociados con los puntos que se encuentran dentro de una cierta distancia $\varepsilon > 0$ de la verdadera función de regresión lineal,

$$f(\mathbf{x}) = \beta_0 + \mathbf{x}^T\beta. \tag{7.3}$$

Esto significa que si la observación $(\mathbf{x}, y)$ es tal que $|y - f(\mathbf{x})| \leq \varepsilon$, entonces la pérdida se toma como 0 y si, por el contrario, la observación $(\mathbf{x}, y)$ cumple que $|y - f(\mathbf{x})| > \varepsilon$, entonces la pérdida es $|y - f(\mathbf{x})| - \varepsilon$. Con esta estrategia en mente, podemos definir los siguientes dos tipos de función de pérdida:

$$L_1^{\varepsilon}(y, f(\mathbf{x})) = \mathrm{máx}\{0, |y - f(\mathbf{x})| - \varepsilon\}. \tag{7.4}$$

$$L_2^{\varepsilon}(y, f(\mathbf{x})) = \text{máx}\left\{0, (y - f(\mathbf{x}))^2 - \varepsilon\right\}. \tag{7.5}$$

Dichas funciones son denominadas funciones de pérdida ε-insensible lineal y cuadrática respectivamente. Veamos la optimización lineal del problema con estas funciones de pérdida.

Definimos las variables de holgura ξ_i y ξ_j' de la siguiente manera. Si la observación $(\mathbf{x}_i, y_i)$ está en la parte superior del ε-tubo, entonces $\xi_i = y_i - f(\mathbf{x}_i) - \varepsilon \geq 0$, mientras que si la observación $(\mathbf{x}_j, y_j)$ se encuentra en la parte inferior del ε-tubo, entonces $\xi_j = f(\mathbf{x}_j) - \varepsilon - y_j \geq 0$. Para las observaciones que quedan fuera del tubo, los valores de las variables de holgura dependen de la forma de la función de pérdida y para aquellas que están dentro del tubo, las variables de holgura tienen valor cero.

Para la función de pérdida ε-insensible lineal, el problema de optimización primal es encontrar $\beta_0, \beta, \xi' = (\xi_1', \cdots, \xi_n')^T$, y $\xi = (\xi_1, \cdots, \xi_n)^T$ tal que

$$\begin{aligned} \text{mín} \quad & \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i') \\ \text{sujeto a:} \quad & y_i - (\beta_0 + \mathbf{x}_i^T\beta) \leq \varepsilon + \xi_i', \\ & (\beta_0 + \mathbf{x}_i^T, \beta) - y_i \leq \varepsilon + \xi_i, \\ & \xi_i' \geq 0, \xi_i \geq 0, \quad i = 1, 2, \ldots, n. \end{aligned} \tag{7.6}$$

La constante $C > 0$ se incluye en la formulación del problema de una manera análoga a como también sucedía para el problema de clasificación, para equilibrar la uniformidad de la función f frente a nuestra tolerancia de desviaciones mayores que ε. Ya que ε solo se encuentra en las restricciones, la solución a este problema de optimización tiene que incorporar una banda alrededor de la función de regresión. En consecuencia, la función de Lagrange primal para el problema (7.6) es la siguiente.

$$\begin{aligned} L_P(\beta, \beta_0, \xi, \xi') \ = & \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i') - \sum_i a_i\left\{y_i - \left(\beta_0 + \mathbf{x}_i^T\beta\right) - \varepsilon - \xi_i'\right\} \\ & - \sum_i b_i\left\{\left(\beta_0 + \mathbf{x}_i^T\beta\right) - \beta_i - \varepsilon - \xi_i\right\} \\ & - \sum_i c_i\xi_i' - \sum_i d_i\xi_i, \end{aligned} \tag{7.7}$$

donde a_i, b_i, c_i y $d_i, i = 1, 2, \ldots, n$, son los multiplicadores de Lagrange, lo que implica que hay una restricción mas que es que a_i, b_i, c_i y d_i, con $i = 1, 2, \ldots, n$, son todas no negativas. Las derivadas parciales de la función de Lagrange primal son

$$\begin{aligned} \frac{\partial L_P}{\partial \beta_0} &= \sum_i a_i - \sum_i b_i, \\ \frac{\partial L_P}{\partial \beta} &= \beta + \sum_i a_i\mathbf{x}_i - \sum_i b_i\mathbf{x}_i, \\ \frac{\partial L_P}{\partial \xi_i} &= C + b_i - d_i, \\ \frac{\partial L_P}{\partial \xi_i'} &= C + a_i - c_i. \end{aligned} \tag{7.8}$$

Igualando estas derivadas a cero para la solución se obtienen las siguientes igualdades.

$$\beta^* = \sum_i (b_i - a_i)\,\mathbf{x}_i, \tag{7.9}$$

$$\sum_i (b_i - a_i) = 0, \tag{7.10}$$

$$C + b_i - d_i = 0, \quad C + a_i - c_i = 0, \quad i = 1, 2, \ldots, n.$$

La igualdad (7.9) se conoce como *expansión del support vector* porque β^* se puede escribir como una combinación lineal de los vectores de variables predictoras $\mathbf{x}_i$. Si se establece que $\beta = \beta^*$ en la ecuación de regresión (7.3) se tiene que

$$f^*(\mathbf{x}) = \beta_0 + \sum_{i=1}^{n} (b_i - a_i)(\mathbf{x}^T \mathbf{x}_i) \tag{7.11}$$

Sustituyendo β^* en la función de Lagrange primal y usando las igualdades (7.9) y (7.10) tenemos su correspondiente problema dual

$$\begin{aligned} \max_{\mathbf{a},\mathbf{b}} \quad & L_D(\mathbf{a}, \mathbf{b}) = (\mathbf{b} - \mathbf{a})^T \mathrm{y} - \varepsilon(\mathbf{b} + \mathbf{a})^T \mathbf{1}_n - \frac{1}{2}(\mathbf{b} - \mathbf{a})^T \mathrm{K}(\mathbf{b} - \mathbf{a}) \\ \text{sujeto a:} \quad & 0 \leq \mathbf{a}, \\ & \mathbf{b} \leq C 1_n, \\ & (\mathbf{b} - \mathbf{a})^T \mathbf{1}_n = 0. \end{aligned} \tag{7.12}$$

donde $\mathbf{a} = (a_1, \ldots, a_n)^T, \mathbf{b} = (b_1, \ldots, b_n)^T$, $\mathbf{1}_n$ es la matriz unitaria de dimensión $n \times n$ y $\mathbf{K} = \{\langle \mathbf{x}_i, \mathbf{x}_j \rangle\}_{i,j=1,\ldots,n}$ es la matriz de los productos internos de los vectores de las variables predictoras para el SVM lineal. Las condiciones de complementariedad de Karush-Kuhn-Tucker establecen que los productos de las variables duales y las restricciones son todas cero,

$$\begin{aligned} a_i(\beta_0 + \mathbf{x}_i^\top \beta - y_i - \varepsilon - \xi_i) = 0, \quad & i = 1, 2, \ldots, n, \\ b_i(y_i - \beta_0 - \mathbf{x}_i^T \beta - \varepsilon - \xi_i') = 0, \quad & i = 1, 2, \ldots, n, \\ \xi_i \xi_i' = 0,\ a_i b_i = 0, \quad & i = 1, 2, \ldots, n, \\ (a_i - C)\xi_i = 0,\ (b_i - C)\xi_i' = 0, \quad & i = 1, 2, \ldots, n. \end{aligned} \tag{7.13}$$

En la práctica, el valor de ε se suele tomar en torno a 0.1. La solución a este problema de optimización produce una función lineal de $\mathbf{x}$ acompañada de una banda o tubo de $\pm\varepsilon$ alrededor de la función. Como hemos comentado antes, los puntos que no caen dentro del tubo son los vectores soporte.

Como para el problema de clasificación también lo podemos extender para el caso no lineal en el que se usan kernels para considerar transformaciones no lineales, pero en este trabajo no se aborda tal cuestión.

Algoritmos de optimización para el cálculo del SVM

Cuando se tiene el problema primal transformado a su dual, existen distintos métodos para su resolución. En particular, se utilizan algunas técnicas de programación cuadrática.

En este trabajo hemos tratado en el Capítulo 5 el método del descenso del gradiente que es uno de los algoritmos más utilizados en la actualidad.

Otras técnicas numéricas para grandes conjuntos de datos de aprendizaje están ahora disponibles en muchos paquetes de software de SVM. Entre los ejemplos de estas técnicas avanzadas se incluyen la "fragmentación", la "descomposición" y la "optimización mínima secuencial". Cada método se basa en ciertos elementos comunes, como son, elegir un subconjunto dentro del conjunto de aprendizaje $\mathcal{C}$, se siguen de cerca las condiciones de optimización de KKT para descubrir qué puntos que no están en el subconjunto violan las condiciones y aplican una estrategia de optimización adecuada.

El método de "chunking" es una técnica utilizada para resolver problemas de optimización cuadrática utilizando un subconjunto más pequeño del conjunto de entrenamiento $\mathcal{C}$, generalmente entre 100 y 500 observaciones. En el contexto de los SVM, para resolver el problema de optimización solo se tienen en cuenta aquellas observaciones que son vectores soporte. Una vez se ha obtenido la solución óptima utilizando el subconjunto de observaciones seleccionadas, se puede aplicar el clasificador resultante de esta primera etapa a todas las observaciones del conjunto de entrenamiento original $\mathcal{C}$. Durante esta segunda fase, algunas observaciones pueden ser clasificadas incorrectamente, lo que significa que el clasificador las asigna a una clase diferente a la esperada. Estas observaciones erróneamente clasificadas son de interés ya que indican posibles áreas donde el clasificador puede mejorar su rendimiento. Para evaluar la importancia de estas observaciones erróneamente clasificadas, se utiliza el concepto de condiciones KKT, que como hemos visto en capítulos anteriores son un conjunto de criterios matemáticos que deben cumplirse para que una solución sea óptima en el problema de optimización cuadrática. Para ello, las observaciones se ordenan en función de cómo violan las condiciones KKT, es decir, se mide cuánto se alejan estas observaciones de cumplir las condiciones óptimas según las restricciones del problema. Esta medida de violación de las condiciones KKT se utiliza para determinar qué observaciones tienen un mayor impacto en la solución y merecen una atención especial en la iteración siguiente del método de "chunking". Este proceso se repite iterativamente hasta que todas las observaciones cumplan con las condiciones KKT, lo que conduce a la solución óptima del problema cuadrático. Sin embargo, debido a la naturaleza de este clasificador y al proceso de selección de puntos, es un algoritmo que tiende a ser lento e ineficiente.

El método de "descomposición", es similar al método chunking, excepto que en cada iteración, el tamaño del subconjunto es siempre el mismo. Si se agregan nuevos puntos al subconjunto significa que se deben eliminar un número igual de puntos antiguos.

El método de "optimización mínima secuencial", se basa en el método de descomposición a diferencia de que el subconjunto consta de solo dos observaciones en cada iteración. Con este algoritmo, se actualizan los valores de los α_i correspondientes a las observaciones para las que se ha resuelto el problema, es decir se actualizan para que la restricción $\sum_{i=1}^{n} \alpha_i y_i$ se satisfaga y la solución se encuentre dentro de la región factible. De estos tres últimos métodos, el de "optimización mínima secuencial" es el más potente como optimizador cuadrático y que más se suele utilizar en las librerías que tienen disponibles funciones para estimar los SVMs.

En cuanto a las direcciones futuras y posibles líneas de trabajo, hay varias áreas en las que se están investigando actualmente las extensiones de los SVMs, como la incorporación de información estructurada en los modelos de SVM, la adaptación de los SVMs a datos

incluso de más alta dimensionalidad, y la combinación de estos con otras técnicas de aprendizaje automático, como redes neuronales y métodos de ensamble.

Apéndice A

Algoritmo del Perceptrón

El primer algoritmo iterativo para el Aprendizaje Automático mediante combinaciones lineales, con una filosofia precursora del SVM, fue es el procedimiento propuesto por Frank Rosenblatt en 1958 en el perceptrón [12]. Este método despertó un gran interés cuando fue introducido, por ser un primer ejemplo de como las máquinas (computadores) pueden ser capaces de "aprender" de forma "autónoma" (algo, hasta entonces, aparentemente "reservado" a los seres vivos). Este método está, pues, bastante ligado a los fundamentos de la Inteligencia Artificial.

Si una respuesta $y_i = 1$ está mal clasificada, entonces $\mathbf{x}_i^T\beta+\beta_0 < 0$, y lo contrario para una respuesta mal clasificada con $y_i = -1$. El perceptrón es un algoritmo iterativo que construye una secuencia de vectores $\beta^{(0)}, \beta^{(1)}, \ldots$. Inicialmente, $\beta^{(0)}$ está configurado para ser el vector con todos sus elementos iguales a 0. En la iteración t, el perceptrón puede encontrase una observación i que está mal etiquetada por $\beta^{(t)}$, es decir, una observación para la cual signo$(\langle\beta^{(t)}, \mathbf{x}_i\rangle) \neq y_i$. Entonces, el perceptrón actualiza $\beta^{(t)}$ añadiéndole la observación $\mathbf{x}_i$ escalada por la etiqueta y_i. Es decir, $\beta^{(t+1)} = \beta^{(t)} + y_i\mathbf{x}_i$. Recordemos que nuestro objetivo es tener $y_i\langle\beta, \mathbf{x}_i\rangle > 0$ para todo i y, además, hay que tener en cuenta que

$$y_i\langle\beta^{(t+1)}, \mathbf{x}_i\rangle = y_i\langle\beta^{(t)} + y_i\mathbf{x}_i, \mathbf{x}_i\rangle = y_i\langle\beta^{(t)}, \mathbf{x}_i\rangle + \|\mathbf{x}_i\|^2.$$

Por lo tanto, la actualización de perceptron guía la solución para que sea "más correcta" la clasificación en la i-ésima observación.

El objetivo final es minimizar

$$D(\beta, \beta_0) = -\sum_{i\in\mathcal{M}} y_i(\mathbf{x}_i^T\beta + \beta_0)$$

donde $\mathcal{M}$ es el conjunto de puntos mal clasificados. La cantidad es no negativa y proporcional a la distancia de los puntos mal clasificados a la frontera (hiperplano) definido por $\mathbf{x}_i^T\beta + \beta_0 = 0$. El gradiente (suponiendo que $\mathcal{M}$ es fijo) está dado por

$$\partial\frac{D(\beta, \beta_0)}{\partial\beta} = -\sum_{i\in\mathcal{M}} y_i\mathbf{x}_i$$
$$\partial\frac{D(\beta, \beta_0)}{\partial\beta_0} = -\sum_{i\in\mathcal{M}} y_i$$

De hecho, el algoritmo utiliza el descenso de gradiente estocástico para minimizar este criterio. Esto significa que en lugar de calcular la suma de las contribuciones del gradiente

de cada observación seguida de un paso en la dirección del gradiente negativo, da un paso después de visitar cada observación. Por lo tanto, las observaciones mal clasificadas se visitan en alguna secuencia, y los parámetros β se actualizan a través de la forma

$$\begin{pmatrix} \beta \\ \beta_0 \end{pmatrix} \leftarrow \begin{pmatrix} \beta \\ \beta_0 \end{pmatrix} + \eta \begin{pmatrix} y_i \mathbf{x}_i \\ y_i \end{pmatrix}$$

Aquí η es la tasa de aprendizaje que, en este caso, puede tomarse como 1 sin pérdida de generalidad. Si las clases son linealmente separables, se puede demostrar que el algoritmo converge a un hiperplano de separación en un número finito de pasos. El Teorema A.1, [3], garantiza que en el caso linealmente separable, el algoritmo se detiene con todos los datos del conjunto de entrenamiento correctamente clasificados y , por tanto, converge a un hiperplano de separación en un número finito de pasos.

Teorema A.1. *Sea* $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ *un conjunto de observaciones que suponemos linealmente separables,* $\alpha = \max_i \|\mathbf{x}_i\|$ *y* γ *es la menor norma* $\|\beta\|$ *de entre todos los* β *tales que* $y_i\langle\beta, \mathbf{x}_i\rangle \geq 1$ *para todo* i, $i = 1, ..., n$. *El algoritmo del perceptrón se detiene después de* $(\alpha\gamma)^2$ *iteraciones como máximo y al detenerse se verifica que* $y_i\langle\beta, \mathbf{x}_i\rangle > 0$ *para todo* i, $i = 1, ..., n$.

Demostración. Por definición de la condición de parada, cuando se detiene, el perceptrón debe haber separado todas las observaciones. Mostraremos que si el perceptrón se ejecuta durante T iteraciones, entonces debemos tener $T \leq (\alpha\gamma)^2$.

Sea $\beta^\star$ un vector que alcanza el mínimo en la definición de γ, esto es $\|\beta^\star\| = \gamma$. En otras palabras, $y_i\langle\beta^\star, \mathbf{x}_i\rangle \geq 1$ para todo i, y entre todos los vectores que satisfacen estas restricciones, $\beta^\star$ es de norma mínima. La idea de la prueba es mostrar que después de realizar T iteraciones, el coseno del ángulo entre $\beta^\star$ y $\beta^{(T+1)}$ es al menos $\frac{\sqrt{T}}{\alpha\gamma}$. Es decir, deseamos probar que

$$\frac{\langle\beta^\star, \beta^{(T+1)}\rangle}{\|\beta^\star\|\|\beta^{(T+1)}\|} \geq \frac{\sqrt{T}}{\alpha\gamma}. \tag{A.1}$$

Por la desigualdad de Cauchy-Schwartz, el lado izquierdo de la ecuación (A.1) es como máximo 1. El cumplimiento de (A.1) necesitaría que

$$1 \geq \frac{\sqrt{T}}{\alpha\gamma} \Rightarrow T \leq (\alpha\gamma)^2.$$

Para demostrar que la (A.1) se cumple, primero demostramos que $\langle\beta, \beta^{(T+1)}\rangle \geq T$. De hecho, en la primera iteración, $\beta^{(0)} = (0, ..., 0)$ y por lo tanto $\langle\beta, \beta^{(0)}\rangle = 0 \geq 0$, mientras que en la iteración t, si actualizamos usando la observación $(\mathbf{x}_i, y_i)$ tendríamos

$$\begin{aligned} \langle\beta^\star, \beta^{(t+1)}\rangle - \langle\beta^\star, \beta^{(t)}\rangle &= \langle\beta^\star, \beta^{(t+1)} - \beta^{(t)}\rangle \\ &= \langle\beta^\star, y_i\beta_i\rangle = y_i\langle\beta^\star, \mathbf{x}_i\rangle \geq 1. \end{aligned}$$

Por lo tanto, después de realizar T iteraciones, obtenemos

$$\langle\beta^\star, \beta^{(T+1)}\rangle = \sum_{t=1}^{T}(\langle\beta^*, \beta^{(t+1)}\rangle - \langle\beta^\star, \beta^{(t)}\rangle) \geq T \tag{A.2}$$

como queríamos.

A continuación, veamos una cota superior de $\|\beta^{(T+1)}\|^2$. Para cada iteración t tenemos que

$$\begin{aligned}\|\beta^{(t+1)}\|^2 &= \|\beta^{(t)} + y_i \mathbf{x}_i\|^2 \\ &= \|\beta^{(t)}\|^2 + 2y_i\langle\beta^{(t)}, \mathbf{x}_i\rangle + y_i^2\|\mathbf{x}_i\|^2, \\ &\leq \|\beta^{(t)}\|^2 + \alpha^2\end{aligned} \tag{A.3}$$

donde la última desigualdad se debe a que la observación i verifica necesariamente que $y_i\langle\beta^{(t)}, \mathbf{x}_i\rangle \leq 0$, y la norma de $\mathbf{x}_i$ está acotada por α. Ahora bien, como $\|\beta^{(0)}\|^2 = 0$, si usamos la desigualdad (A.3) recursivamente para T iteraciones, obtenemos que

$$\|\beta^{(T+1)}\|^2 \leq T\alpha^2 \Rightarrow \|\beta^{(T+1)}\| \leq \sqrt{T}\alpha. \tag{A.4}$$

Combinando las desigualdades (A.2) con (A.4) y que $\|\beta^\star\| = \gamma$ obtenemos que

$$\frac{\langle\beta^{(T+1)}, \beta^\star\rangle}{\|\beta^\star\|\|\beta^{(T+1)}\|} \geq \frac{T}{\gamma\sqrt{T}\alpha} = \frac{\sqrt{T}}{\alpha\gamma}.$$

Así, hemos demostrado que se cumple la desigualdad (A.1), y esto concluye la demostración. □

El perceptron es simple de implementar y está garantizado para converger. Sin embargo, su tasa de convergencia depende del parámetro γ, que en algunas situaciones puede ser muy grande. En tales casos, sería mejor implementar otro tipo de algoritmos. No obstante, para muchos conjuntos de datos, el tamaño de γ está garantizado que no puede ser demasiado grande y el perceptrón, en esos casos, converge bastante rápido.

Hay una serie de problemas con este algoritmo:

- Cuando los datos son separables, hay muchas soluciones, y la que se encuentra depende de la inicialización considerada.
- El número "finito" de pasos que precisa puede ser muy grande Si se tiene una separación muy estrecha entre las clases, es decir, el margen es muy pequeño, se requieren mayores tiempos computacionales para llegar a la solución óptima.
- Cuando los datos no son separables, el algoritmo no converge y se desarrollan "ciclos". Los ciclos pueden ser muy largos y, por tanto, difíciles de detectar.

Bibliografía

[1] Trevor Hastie, Robert Tibshirani & Jerome Friedman, *The Elements of Statistical Learning (Data Mining, Inference, and Prediction)*, Springer 2 Edition (2008).

[2] Nello Critianini & John Shawe-Taylor, *An Introduction to Support Vector Machines and other kernel-based learning methods*, Cambridge University Press.

[3] Shai Shalev Shwartz & Shai Ben David, *Understanding Machine Learning*, Cambridge University Press (2014).

[4] Alan J. Izenman, *Modern Multivariate Statistical Technicques*, Springer Texts in Statistics (2006).

[5] Corinna Cortes & Vladimir Vapnik, *Support-vector networks. Machine learning*, SpringerLink (1995).

[6] Bernhard E. Boser, Isabelle M. Guyon & Vladimir N. Vapnik, *A Training Algorithm for Optimal Margin Classifiers*, Proceedings of the 5th Annual Workshop on Computational Learning Theory (COLT'92).

[7] Alexandros Karatzoglou, David Meyer & Kurt Hornik *Journal of Statistical Software, Support Vector Machines in R.* April 2006, Volume 15, Issue 9.

[8] *UCI machine learning repository.* `https://archive.ics.uci.edu/ml/index.php`

[9] *Kaggle.* `https://www.kaggle.com/datasets`

[10] *The Comprehensive R Archive Network.* `https://cran.r-project.org/manuals.html`.

[11] Gareth James, Daniela Witten, Trevor Hastie & Robert Tibshirani, *An Introduction to Statistical Learning with Applications in R*, Springer Texts in Statistics (2013).

[12] Frank Rosenblatt, *The perceptron: a probabilistic model for information storage and organization in the brain*, Psychological Review 65, 386-408 (1958).