Máquinas de Vectores Soporte (SVM)

Índice

Referencia principal

Cap.5 de "Hands On Machine Learning with Scikit Learn and TensorFlow"

Al terminar este tema comprenderemos el fundamento teórico de los SVM y como se utilizan en Python

1. Intuición

Supongamos un conjunto de entrenamiento con m vectores de características bidimensionales, $\mathbf{x} = (x_1, x_2)^T$ que sea linealmente separable en dos clases, como el mostrado en la Figura 1. Intuitivamente, podríamos trazar la superficie de decisión de varios clasificadores lineales ya que el 'espacio' entre ambas clases es bastante amplio. Por ejemplo, a la derecha se han trazado 3 clasificadores, C_1, C_2 y C_3; cada uno de los cuales separa sin ningun error el conjunto de entranmiento. Pero ¿qué pasará cuando ejecutemos cualquiera de ellos sobre nuevos ejemplos?

Cuando la superficie de separación está demasiado cerca de los ejemplos de entrenamiento es muy probable que los ejemplos nuevos se clasifiquen mal, o sea que 'caigan en el lado equivocado'. ¿No sería mejor buscar un clasificador que maximizara el espacio entre clases, denominado **margen**? Maximizar el margen es equivalente a minimizar el riesgo de que nuevos ejemplos sean mal clasificados. Como procede de la construcción del clasificador se le llama **riesgo estructural**. .

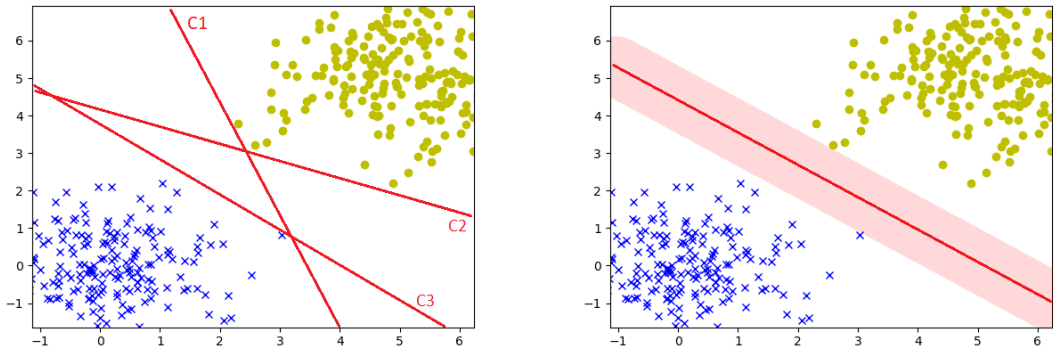

Figura 1: (Der.) Tres clasificadores que no cometen ningún error en el conjunto de entrenamiento pero seguramente funcionen mal con datos nuevos. (Izq.) Este clasificador que maximiza el margen entre ambas clases. [Fuente: Original de A. Cuesta]

En ocasiones un clasificador que maximice el margen puede ser demasiado restrictivo. Si permitimos que algunos ejemplos puedan estar dentro del margen, entonces el clasificador resultante puede tener mayor poder generalizador. Esta técnica se denomina *soft margin*, y en la Figura 2 se puede ver un ejemplo de como cambia tanto el margen como el clasificador.

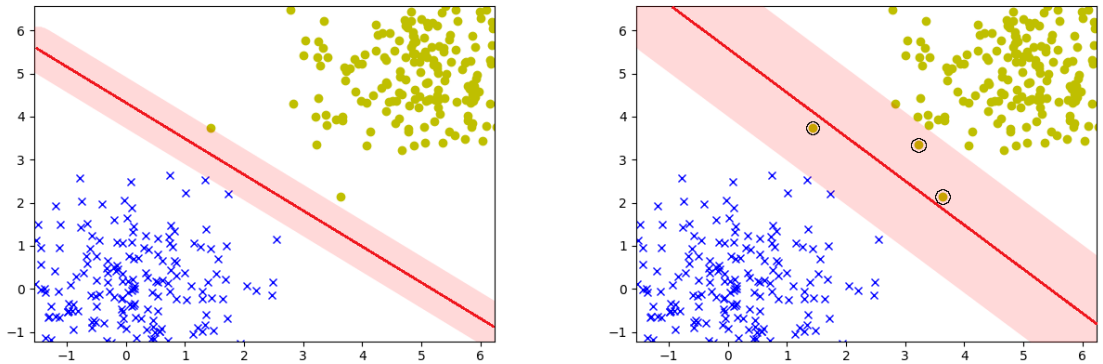

Figura 2: (Der.) Márgenes *hard*. (Izq.) márgenes *soft* [Fuente: Original de A. Cuesta]

¿Y si no son linealmente separables?. En la Figura 3 podemos ver un conjunto de datos cuya superficie de separación no es lineal. Como vimos la semana pasada, una manera de resolver el

problema es generar características nuevas de tal manera que, en el nuevo espacio, si lo sean. Lo más fascinante de los SVM es que seremos capaces de encontrar el clasificador no lineal sin hacer la transformación de las características mediante el **truco del kernel**.

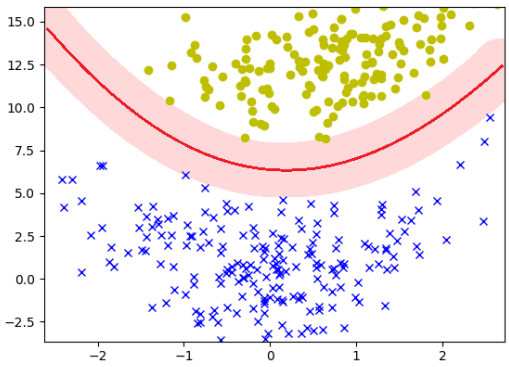

Figura 3: Clasificación no lineal. [Fuente: Original de A. Cuesta]

2. Clasificación lineal

En las semanas pasadas utilizabamos $\hat{y} = \mathbf{w}^T\mathbf{x}$ para indicar el resultado de la suma ponderada del ejemplo por los pesos, mientras que $\hat{t}$ indicaba la estimación de la etiqueta resultante de $\hat{y}$ y del umbral θ. El motivo era que explicabamos la clasificación a partir de la regresión. Los SVM se desarrollaron directamente para clasificación. Por tanto en este tema utilizaremos una notación con dos diferencias importantes:

- Ahora los vectores $\mathbf{w}$ y $\mathbf{x}$ no estaran 'expandidos', es decir:

$$\mathbf{w} = (w_1, w_2, \ldots, w_m)^T \quad , \quad \mathbf{x} = (x_1, x_2, \ldots, w_m)^T \ .$$

- Dado que hemos sacado el término independiente del vector de pesos, la expresión del discriminante es ahora:

$$\hat{t} = \begin{cases} +1 & \text{si} \quad \mathbf{w}^T\mathbf{x} + b > 0 \\ -1 & \text{si} \quad \mathbf{w}^T\mathbf{x} + b < 0 \end{cases} \tag{1}$$

  donde b es el término independiente y, al igual que en la semana pasada, las etiquetas son $+1$ y -1 porque serán 'matemáticamente convenientes'.

2.1. Márgenes *Hard*

El clasificador con márgenes *hard* es decir que no permitan ejemplos en su interior, es el primer paso para construir el problema de optimización de los SVM

Para hacer más sencilla la explicación vamos a reducir el número de caracaterísticas a dos, de modo que $\mathbf{w} = (w_1, w_2)^T$ y $\mathbf{x} = (x_1, x_2)^T$. En este caso el hiperplano separador será una recta con ecuación implícita $w_1x_1 + w_2x_2 + b = 0$, que escrita de forma compacta queda $\mathbf{w}^T\mathbf{x} + b = 0$. Toda recta se puede definir por un vector director $\mathbf{v} = (v_1, v_2)^T$ y un punto, que en este caso será b. Y recordando geometría sabemos que todo hiperplano (y la recta lo es) viene definido por su vector característico o normal, porque es perpendicular a la superficie, y un punto. Los componentes del vector normal son los coeficientes de las variables en la ecuación implícita, por tanto no es otro que $\mathbf{w}$, y el punto es, de nuevo, b.

Con esta visión geométrica del problema, podemos calcular la distancia a la que se encuentra el hiperplano separador del origen siguiendo el razonamiento de la Figura 4, donde se ha utilizado las siguiente propiedad:

> Puesto que $\mathbf{v}$ es perpendicular a $\mathbf{w}$, su producto escalar $\mathbf{v}^T\mathbf{w} = 0$. Una manera conveniente es $\mathbf{w} = (v_2/v_1, -1)^T$. Así efectivamente $\mathbf{v}^T\mathbf{w} = (v_1 v_2/v_1) - v_2 = 0$.

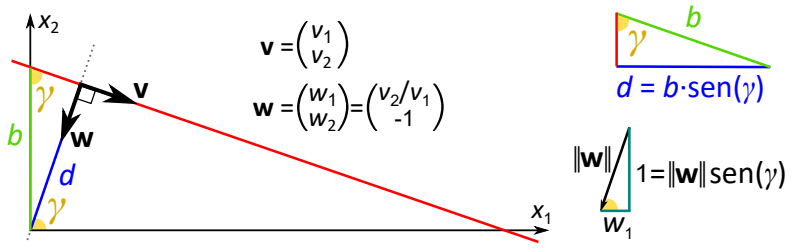

Figura 4: El clasificador lineal viene dado por un vector director $\mathbf{v}$ y un punto de corte b. El vector $\mathbf{w}$ es perpendicular a $\mathbf{v}$ y sirve para calcular la distancia de la recta al origen. Combinando todo podemos representar d en función de $\mathbf{w}$ y b. [Fuente: Original de A. Cuesta]

En definitiva, la distancia del hiperplano al origen se puede escribir como

$$d = \frac{b}{\|\mathbf{w}\|}$$

Para crear el márgen 'duro' (*hard*) debemos considerar dos hiperplanos paralelos tal y como se muestra en la Figura 5, de manera que la distancia entre ellos sea lo más grande posible pero no se extienda más allá de los ejemplos.

Por otro lado la eq. (1) dice que la etiqueta será positiva si el resultado de $\mathbf{w}^T\mathbf{x} + b$ es positivo, y que será negativa si el resultado es negativo, pero en ninguno de los dos casos especifica cuanto mayor o menor que cero debe ser. Si imponemos que sea de 1 unidad entonces la norma $\|\mathbf{w}\|$ se hará mayor o menor para acomodarse a esta restricción, pero su vector $\mathbf{w}$ seguirá teniendo la misma dirección. Al imponer esta nueva restricción la región de los ejemplos positivos es aquella que $\hat{t} = +1 \Longrightarrow \mathbf{w}^T\mathbf{x} + b \geq +1$. Del mismo modo, la región de ejemplos negativos es aquella que $\hat{t} = -1 \Longrightarrow \mathbf{w}^T\mathbf{x} + b < -1$. Y entonces:

$$2\varepsilon = (d + \varepsilon) - (d - \varepsilon) = \frac{b + 1 - (b - 1)}{\|\mathbf{w}\|} = \frac{2}{\|\mathbf{w}\|}, \quad \text{luego} \quad \varepsilon = \frac{1}{\|\mathbf{w}\|}$$

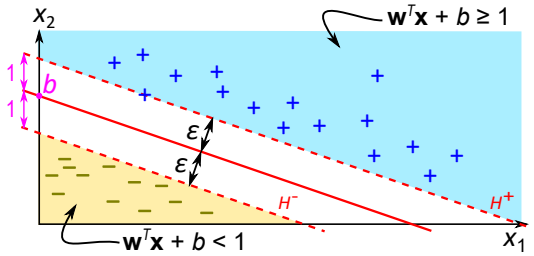

Figura 5: Para crear el margen se añaden dos planos paralelos entre las clases, sin que ningún ejemplo caiga entre ellos. [Fuente: Original de A. Cuesta]

Con lo cual llegamos a dos conclusiones importantes:

- Cuanto más pequeña sea la norma del vector de pesos $\|\mathbf{w}\|$, mayor será el márgen ε ; es decir buscamos $\mathbf{w}^* = \arg\min_{\mathbf{w}} \|\mathbf{w}\|$

- Las dos restricciones impuestas se pueden escribir en una sola:

$$\left.\begin{array}{lllll} \text{Si} & \hat{t}^{(i)} = +1 & \text{entonces} & \mathbf{w}^{(i)T}\mathbf{x} + b \geq +1 \\ \text{Si} & \hat{t}^{(i)} = -1 & \text{entonces} & \mathbf{w}^{(i)T}\mathbf{x} + b < -1 \end{array}\right\} = \left(\hat{t}^{(i)}\right)\left(\mathbf{w}^{(i)T}\mathbf{x} + b\right) \geq 1.$$

Con lo que finalmente podemos plantar el siguiente problema de optimización con restricciones:

$$\text{mín } \tfrac{1}{2}\mathbf{w}^T\mathbf{w}, \quad \text{sujeto a} \quad \left(\hat{t}^{(i)}\right)\left(\mathbf{w}^{(i)T}\mathbf{x} + b\right) \geq 1, \quad \text{para} \quad i = 1, 2, \ldots, m.$$

donde la función objetivo ha cambiado ligeramente para que sea más fácil de resolver. La razón es que $\mathbf{w}^T\mathbf{w} = \|\mathbf{w}\|^2$, mientras que el $\frac{1}{2}$ se va con el exponente al derivar la función de coste.

2.2. Márgenes *Soft*

El modo de permitir que cada ejemplo i se situe sobre el margen, o incluso lo sobrepase, es penalizarlo en la función de coste con un peso $\zeta^{(i)}$ que mide en cuanto se puede violar la restricción de no sobrepasar el margen. Estos pesos se denominan *slack variables*(*slack* significa que 'aprieta poco'), y se asignan del modo que muestra la Figura 6.

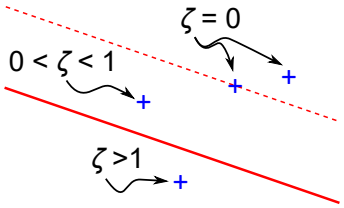

Figura 6: *Slack variables* para ejemplos que violan la restricción. [Fuente: Original de A. Cuesta]

Por tanto la función de coste tendrá dos objetivos contrapuestos. Por un lado se pretende hacer el margen lo más grande posible y por otro se quiere minimizar el número de excepciones. Para gestionar dicho compromiso se añade el hiperparámetro C, de manera que la función de coste queda:

$$\min_{\mathbf{w},b,\zeta} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{m}\zeta^{(i)}$$

Por otro lado también se modifican las restricciones ya que si antes un ejemplo debía cumplir que $\left(\hat{t}^{(i)}\right)\left(\mathbf{w}^{(i)T}\mathbf{x} + b\right) \geq 1$, ahora que se le permite estar en el margen, aunque en la medida de lo posible NO estar mal clasificado, su peso será $0 \leq \zeta < 1$, por lo que la restricción pasa a ser $\left(\hat{t}^{(i)}\right)\left(\mathbf{w}^{(i)T}\mathbf{x} + b\right) \geq 1 - \zeta^{(i)}$.

En definitiva, el problema de optimización que se plantea para clasificadores SVM lineales es:

$$\begin{array}{ll} \min\limits_{\mathbf{w},b,\zeta} & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{m}\zeta^{(i)}, \\ \text{sujeto a} & \begin{cases} \left(\hat{t}^{(i)}\right)\left(\mathbf{w}^{(i)T}\mathbf{x} + b\right) \geq 1 - \zeta^{(i)} \\ \zeta^{(i)} \geq 0 \\ i = 1, 2, \ldots, m. \end{cases} \end{array} \tag{2}$$

2.3. El problema dual

La ecuación (2) es un problema de optimización cuadrática con restricciones cuya solución se puede seguir en los libros. Por el tiempo tan reducido de este curso, no vamos a ver su desarrollo

completo. Lo más importante es que el problema se resuelve mediante el método de los multiplicadores de Lagrange, pasando el problema a un problema alternativo pero equivalente, llamado 'dual', con la siguiente formulación:

$$\min_{\alpha} \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha^{(i)}\alpha^{(j)}t^{(i)}t^{(j)}\mathbf{x}^{(i)T}\mathbf{x}^{(j)} - \sum_{i=1}^{m}\alpha^{(i)}, \text{ sujeto a } \alpha^{(i)} > 0 \text{ para } i = 1, 2, \ldots, m. \tag{3}$$

Aunque no lo vayamos a resolver, conocer el planteamiento del problema es importante porque presenta una serie de características que hacen de los SVM una técnica diferente de otras.

- Cada ejemplo $\mathbf{x}^{(i)}$ tiene un multiplicador de Lagrange $\alpha^{(i)}$.
- El problema se puede resolver mediante métodos computacionales.
- Una vez resuelto, algunos (muchos) multiplicadores se anulan.
  Los ejemplos asociados a los multiplicadores distintos de cero son los **vectores soporte**.
- Una vez obtenidos los multiplicadores distintos de cero, el vector de pesos óptimo es:

$$\mathbf{w}^* = \sum_{i=1}^{m}\alpha^{*(i)}t^{(i)}\mathbf{x}^{(i)} \quad ; \quad b^* = \frac{1}{n_s}\sum_{i:\alpha^{*(i)}>0}\left((1-t^{(i)})(\mathbf{w}^{*T}\mathbf{x}^{(i)})\right), \tag{4}$$

  donde n_s es el número de vectores soporte, es decir el número de $\alpha^{*(i)} > 0$.
- La función de coste dual depende del producto escalar de cada ejemplo i con cada ejemplo j, es decir de $\mathbf{x}^{(i)T}\mathbf{x}^{(j)}$.

3. Clasificación no lineal

Como ya vimos, un modo de hacer clasificación no lineal es transformar los vectores de características y despúes hacer clasificación lineal sobre los ejemplos transformados. Es decir que aplicamos una función $\phi(\mathbf{x})$ a cada vector.

3.1. El truco del Kernel

Si hacemos esto en SVM entonces el problema dual pasaría a depender del producto escalar de cada ejemplo transformado i con cada ejemplo transformado j.

El producto escalar de dos vectores está relacionado con la distancia que hay entre ellos. Si tuvieramos una función que devolviera la 'distancia' entre dos vectores dados, que no tiene por qué ser la distancia en línea recta, podríamos usar esa función en vez del producto escalar.

Dicha función $K(\mathbf{x}, \mathbf{x}')$ se denomina *Kernel* entre dos vectores $\mathbf{x}$ y $\mathbf{x}'$. Si lo aplicamos a la función de coste dual con vectores transformados pasaríamos de:

$$\min_{\alpha} \tfrac{1}{2}\textstyle\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha^{(i)}\alpha^{(j)}t^{(i)}t^{(j)}\phi(\mathbf{x}^{(i)})^T\phi(\mathbf{x}^{(j)}) - \sum_{i=1}^{m}\alpha^{(i)}$$

$$\Downarrow$$

$$\min_{\alpha} \tfrac{1}{2}\textstyle\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha^{(i)}\alpha^{(j)}t^{(i)}t^{(j)}K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) - \sum_{i=1}^{m}\alpha^{(i)}$$

Esto se conoce como el **truco del kernel** o *kernel trick*.

La solución del problema no lineal sería entonces:

$$\mathbf{w}^* = \sum_{i=1}^{m} \alpha^{*(i)} t^{(i)} \phi(\mathbf{x}^{(i)}) \quad ; \quad b^* = \frac{1}{n_s} \sum_{i:\alpha^{*(i)}>0} \left((1 - t^{(i)})(\mathbf{w}^{(*)T} \phi(\mathbf{x}^{(i)})) \right),$$

pero, si no conocemos la transformación ϕ, entonces ¿¿cómo vamos a calcular los pesos óptimos?? Los pesos sirven para predecir la etiqueta de nuevos ejemplos. Con el *kernel trick* podemos hacer la predicción directamente, sin tener que calcular dichos pesos. La etiqueta predicha para un nuevo ejemplo $\mathbf{z}$ sería:

$$\hat{t} = \mathbf{w}^{*T} \phi(\mathbf{z}) + b^* = \left(\sum_{i=1}^{m} \alpha^{*(i)} t^{(i)} \phi(\mathbf{x}^{(i)}) \right)^T \phi(\mathbf{z}) + b^*$$

Como $\alpha^{(i)}$ y $t^{(i)}$ son escalares, el único vector es $\phi(\mathbf{x}^{(i)})$, por tanto:

$$\hat{t} = \sum_{i:\alpha^{*(i)}>0} \alpha^{*(i)} t^{(i)} \left(\phi(\mathbf{x}^{(i)})^T \phi(\mathbf{z}) \right) + b^* = \sum_{i:\alpha^{*(i)}>0} \alpha^{*(i)} t^{(i)} K(\mathbf{x}^{(i)}, \mathbf{z}) + b^*$$

En la expresión anterior se han eliminado todos los multiplicadores de Lagrange $\alpha^{*(i)} = 0$. Podemos ver que la predicción se hace mediante la suma del ejemplo dado $\mathbf{z}$ 'kernelizado' con cada uno de los vectores soporte y ponderada por la etiqueta y el multiplicador de Lagrange de cada uno de los vectores soporte. Sin embargo en la expresión aún queda por desarrollar el término b^*, que manipulando igual que antes quedaría:

$$b^* = \frac{1}{n_s} \sum_{i:\alpha^{*(i)}>0} \left(\left(1 - t^{(i)}\right) \sum_{j:\alpha^{*(j)}>0} \left(\alpha^{*(j)} t^{(j)} K(\mathbf{x}^{(j)}, \mathbf{z}) \right) \right)$$

En definitiva, la predicción depende del Kernel y de los vectores soporte completamente.

3.2. Kernels

Diferentes Kernels dan lugar a clasificadores con diferentes expresividades. Pero no toda función sirve como Kernel. Sin entrar en detalles, para que una función sea Kernel debe satisfacer el teorema de Mercer. En la tabla siguiente se muestran 4 Kernels muy populares

Nombre	$K(\mathbf{a}, \mathbf{b}) =$	Hiperparámetros
Lineal	$\mathbf{a}^T\mathbf{b}$	— — —
Polinomial	$\left(\gamma \mathbf{a}^T\mathbf{b} + r\right)^d$	γ, r, d
Sigmoide	$\tanh\left(\gamma \mathbf{a}^T\mathbf{b} + r\right)$	γ, r
RBF Gaussiano	$\exp\left(-\gamma \|\mathbf{a} - \mathbf{b}\|^2\right)$	γ

RBF = Radial Basis Function

Índice alfabético