Tarea 4 - Pregunta 2

Uso de Kernels en SVM

Sofía Chávez Bastidas

Recordemos que las formulaciones Primal y Dual de la SVM son:

Formulación Primal

En esta formulación se busca el par de $\mathbf{w}$ y b que minimizan una función objetivo sujeta a las restricciones de clasificación, esta función es equivalente a maximizar el ancho de margen. La formulación primal con hard margin es:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\ \textbf{s.a.} \quad & y_i(w^T x_i + b) \geq 1, \forall i \in \{1, .., N\} \end{aligned} \tag{1}$$

Mientras que para soft margin es:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + c\sum_{i=1}^{N} \xi_i \\ \textbf{s.a.} \quad & y_i(w \cdot x_i + b) \geq 1 - \xi_i, \xi_i \geq 0, \forall i \in \{1, .., N\} \end{aligned} \tag{2}$$

Formulación Dual

Por contraparte, en la formulación dual el problema se reescribe en función de los multiplicadores de Lagrange α_i. La formulación dual con hard margin es:

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle \\ \text{s. a.} \quad & \sum_{i=1}^{N} \alpha_i y_i = 0, \quad 0 \leq \alpha_i, \forall i \end{aligned} \tag{3}$$

A su vez, con soft margin es:

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle \\ \text{s. a.} \quad & \sum_{i=1}^{N} \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq c, \forall i \end{aligned} \tag{4}$$

Con ello, la formulación dual nos permite clasificar un nuevo punto mediante:

$$\hat{y}(x_*) = \mathbf{sgn}\left(\sum_{i=1}^{N} \alpha_i y_i \langle x_i, x_* \rangle + b \right) \tag{5}$$

1. ¿Cuáles son las principales ventajas de usar la formulación dual en vez de la formulación primal en SVM?

Las principales ventajas de utilizar la formulación dual sobre la primal en SVM son:

- La dual permite el uso del truco del kernel, que es útil para trabajar en espacios de alta dimensionalidad sin calcular explícitamente las transformaciones Phi, esto facilita la implementación de SVM para datos no linealmente separables.
- La dual también es mejor en términos de complejidad computacional, puesto que el problema de optimización depende del número de muestras de entrenamiento (N) y no del número de características d (i. e. la dimensión de los datos). Esto puede ser más eficiente computacionalmente cuando $N < d$.

- Además de lo anterior, la complejidad computacional también es manejada de mejor forma al momento de clasificar un nuevo punto que no estuvo en el set de entrenamiento. Al ajustar el modelo, sólo algunos α_i serán no nulos, aquellos que correspondan a support vectors, como los demás $\alpha_j = 0$, para calcular la clase de un nuevo punto basta iterar sobre los support vectors, que normalmente son muy pocos respecto al dataset completo.

2. ¿Por qué es útil utilizar el truco del kernel en este tipo de clasificadores?

El truco del kernel es útil en las svm por varios motivos, en primer lugar, permite transformar los datos a un espacio de mayor dimensionalidad, mapeando implícitamente el dato a un espacio con más features, sin necesidad de calcular explícitamente la transformación $\phi(x)$. Esto facilita la clasificación de datos no linealmente separables en el espacio original.

Una consecuencia de esto y un segundo motivo de la utilidad del truco del kernel es su flexibilidad en esta trasformación, permitiendo usar diferentes tipos de kernels (lineal, polinómico, RBF, etc.) que adapten el modelo a diferentes tipos de problemas y datasets.

Una tercera característica de la utilidad del kernel es su baja complejidad respecto a sus transformaciones, al respecto cabe mencionar el kernel RBF, que implícitamente mapea los datos a un espacio con dimensiones infinitas, sin embargo, conserva un costo computacional finito. Además entrega propiedades como una frontera de decisión infinitamente diferenciable.

3. Soft margin vs kernel trick

Cuando los datos de entrenamiento no son linealmente separables, el problema de optimización asociado a SVM no tiene solución. Para evitar este problema de infactibilidad es posible utilizar la técnica de *soft margin* o bien, el truco del kernel. Explique en qué se diferencian ambas técnicas y cómo cambian la solución del problema.

Soft margin es una extensión de la SVM básica de hard margin. Mientras que hard margin requiere que todos los puntos estén correctamente clasificados, soft margin permite que algunas muestras se clasifiquen incorrectamente. Para ello introduce de variables de holgura (ξ_i) en la optimización, estas variables relajan la condición de separación estricta, generando un trade off entre la maximización del margen y la precisión de la clasificación.

El balance entre ancho de margen y precisión de la clasificación viene dado por c en las ecuaciones 2 y 4, a menor c, el modelo es más flexible respecto a las muestras mal clasificadas y permite más errores para maximizar el ancho de margen. Por contraparte, a mayor c, se penaliza con más rigor las muestras más clasificadas (el límite $c \to \infty$ se recupera el hard margin)

El truco del kernel se utiliza para mapear los datos originales a un espacio de alta dimensión donde los datos puedan ser linealmente separables. Esto se logra usando una función kernel $K(x_i, x_j)$, que calcula el producto interno en el espacio de características sin necesidad de realizar o calcular explícitamente la transformación $\phi(\mathbf{x})$.

Con ello, el problema dual se convierte en:

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \\ \textbf{s.a.} \quad & 0 \leq \alpha_i \leq c, \quad \sum_{i=1}^{N} \alpha_i y_i = 0 \end{aligned} \tag{6}$$

Por lo tanto, la diferencia entre ambos radica en que soft margin modifica la formulación primal y dual de la SVM para permitir errores de clasificación, resultando en una optimización más flexible que puede manejar datos no separables en el espacio original, mientras que el kernel trick permite la transformación de los datos a un espacio de alta dimensión donde una separación lineal puede ser posible.