Support Vector Machine
SVM

Dr. Mauricio Toledo-Acosta

Diplomado Ciencia de Datos con Python

Table of Contents

Introducción

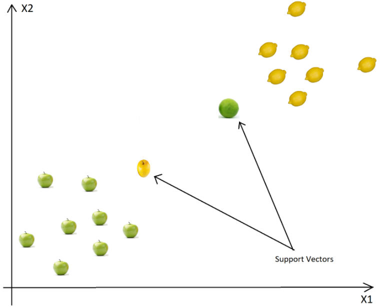
Introducción

Support Vector Machine

Modelo supervisado de clasificación binaria que busca encontrar una frontera de decisión óptima que separe las clases de puntos.

Introducción

En lugar de aprender las caracteristicas que los separan, SVM busca manzanas que son similares a limones y viceversa. Estos son los *vectores de soporte*, sobre estos vectores el algoritmo busca encontrar el mejor hiperplano que los separa.
La distancia de los vectores de soporte a la frontera de decisión es el *margen*.



El modelo lineal de clasificación

Los puntos x que satisfacen $y(x) = w^T \cdot x + w_0 = 0$ forman la frontera de decisión (FD), la cual divide al espacio de datos en dos regiones.

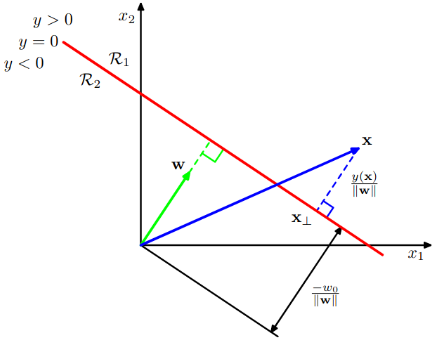
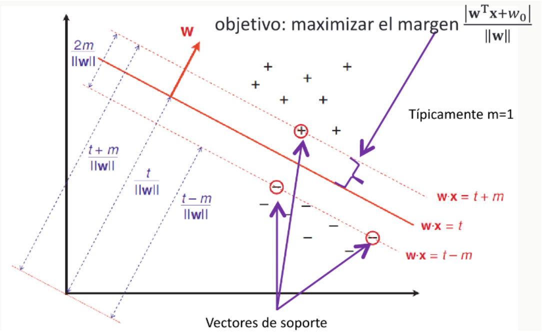


Table of Contents

SVM de margen duro

Analicemos el caso con datos linealmente separables. Cambiaremos ligeramente la notación, FD está definida por $g(x) = w^T \cdot x - t = 0$. Queremos encontrar una FD con margen $m = 1$.



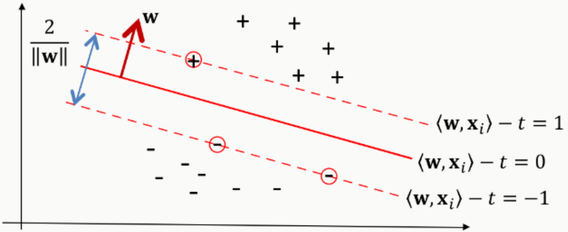
Planteamiento del problema

- Nuestro objetivo es:

$$\mathbf{w}^*, t^* = \operatorname*{argmin}_{\mathbf{w},t} \frac{1}{2}\|\mathbf{w}\|^2$$

- Sujeto a las siguientes N restricciones:

$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle - t) \geq 1, \qquad 1 \leq i \leq N$$



Planteamiento del problema

Minimizar Maximizar

- Definimos el lagrangiano

$$
\begin{aligned}
\mathcal{L}_P(\mathbf{w}, t, \alpha_1, \cdots, \alpha_N) &= \frac{1}{2}\|\boldsymbol{w}\|^2 - \textstyle\sum_{i=1}^{N} \alpha_i (yi\,(\langle \mathbf{w}, \mathbf{x}_i \rangle - t\,) - 1) \\
&= \frac{1}{2}\|\boldsymbol{w}\|^2 - \textstyle\sum_{i=1}^{N} \alpha_i y_i \langle \mathbf{w}, \mathbf{x}_i \rangle + \sum_{i=1}^{N} \alpha_i y_i t + \sum_{i=1}^{N} \alpha_i \\
&= \frac{1}{2}\langle \mathbf{w}, \mathbf{w} \rangle - \langle \mathbf{w}, \textstyle\sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i \rangle + t(\sum_{i=1}^{N} \alpha_i y_i) + \sum_{i=1}^{N} \alpha_i
\end{aligned}
$$

- Para un t óptimo $\partial_t \mathcal{L}_P = 0 \Longrightarrow \sum_{i=1}^{N} \alpha_i y_i = 0$
- Para pesos óptimos $\partial_{\mathbf{w}} \mathcal{L}_P = 0 \Longrightarrow \mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$

Planteamiento del problema

- *Reinsertando estas expresiones en $\mathcal{L}_P$ obtenemos $\mathcal{L}_D$ el lagrangiano del problema dual:*

$$\mathcal{L}_D(\alpha_1, \cdots, \alpha_N) = -\frac{1}{2}\left\langle \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i , \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i \right\rangle + \sum_{i=1}^{N} \alpha_i$$

$$= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{i=1}^{N} \alpha_i$$

Planteamiento del problema

- *El problema de optimización dual es el siguiente:*

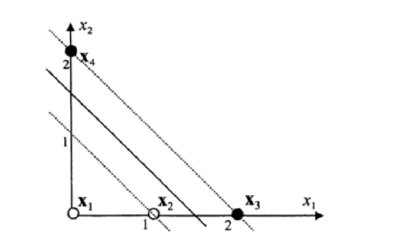
$$\alpha_1^*, \cdots, \alpha_N^* = \underset{\alpha_1,\cdots,\alpha_N}{\operatorname{argmax}} -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\, \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{i=1}^{N} \alpha_i$$

- *Sujeto a las restricciones:*

$$\alpha_i > 0 \ ,\ 1 \leq i \leq N \qquad \text{y} \qquad \sum_{i=1}^{N} \alpha_i y_i = 0$$

Ejemplo

- Encuentra W óptimo para este problema: X1=[0,0] X2=[1,0] para la clase (+1) y X3=[2,0] y X4=[0,2] para la clase (-1)



Ejemplo

$$\mathcal{L}_D(\alpha_1, \cdots, \alpha_N) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle$$

$$\mathcal{L}_D = (\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4) - \frac{1}{2}(\alpha_2{}^2 - 4\alpha_2\alpha_3 + 4\alpha_3{}^2 + 4\alpha_4{}^2)$$

Diferenciando con respecto a los α's y utilizando la restricción $\sum_{i=1}^{N} \alpha_i y_i = 0$ obtenemos:

$$\begin{cases} \alpha_1 + \alpha_2 - \alpha_3 - \alpha_4 = 0 \\ \alpha_2 - 2\alpha_3 = 1 \\ -2\alpha_2 + 4\alpha_3 = 1 \\ 4\alpha_4 = 1 \end{cases}$$

de donde: $\alpha_1 = 0,\ \alpha_2 = 1,\ \alpha_3 = \frac{3}{4},\ \alpha_4 = 1/4$

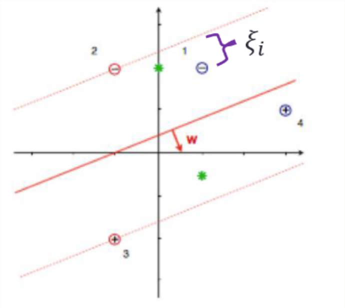
Aplicando: $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$, finalmente obtenemos

$\mathbf{w} = \begin{bmatrix} -1/2 \\ -1/2 \end{bmatrix}$, $w_0 = 3/4$ y $d(x) = 3 - 2x_1 - 2x_2 = 0$

Table of Contents

SVM de margen suave

- La SVM anterior no funciona con datos no-separables
- Introducimos variables de holgura ξ_i para cada dato de entrada, lo que les permite a algunos de ellos estar dentro del margen, o incluso del lado equivocado de la frontera de decision.



SVM de margen suave

$$\mathbf{w}^*, t^*, \xi_i^* = \underset{\mathbf{w}, t, \xi_i}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \xi_i$$

$$\text{sujeto a } y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle - t) \geq 1 - \xi_i \text{ y } \xi_i \geq 0, 1 \leq i \leq N$$

- C es un parámetro definido por el usuario que balancea la maximización del margen contra la minimización de las variables de holgura:
  - *un valor alto de C significa que los errores de margen son altamente costosos,*
  - *un valor pequeño de C permite más errores de margen con tal de hacer mas grande el margen.*
- Si permitimos más errores de margen necesitamos menos vectores de soporte, por lo tanto C controla la 'complejidad' de la SVM y por ello se le denomina el *parámetro de complejidad*.

SVM de margen suave

- Buscamos soluciones mediante el nuevo Lagrangiano:

$$\mathcal{L}(\mathbf{w}, t, \xi_i, \alpha_i, \beta_i) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i\big(yi\,(\langle\mathbf{w},\mathbf{x}_i\rangle - t\,) - (1-\xi_i)\big) - \sum_{i=1}^{N}\beta_i\xi_i$$

$$= \mathcal{L}(\mathbf{w}, t, \alpha_i\,) + \sum_{i=1}^{N}(C - \alpha_i - \beta_i)\xi_i$$

- La solución óptima es tal que $\partial_{\xi_i}\mathcal{L} = 0 \Longrightarrow$ el término añadido desaparece en el problema dual.
- Además, puesto que α_i y β_i son positivos, α_i no puede ser mayor a C:

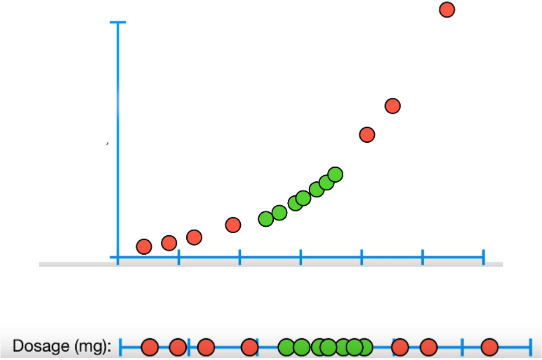
$$\alpha_1^*, \cdots, \alpha_N^* = \underset{\alpha_1,\cdots,\alpha_N}{\operatorname{argmax}} -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\,\alpha_j y_i y_j\,\langle\mathbf{x}_i, \mathbf{x}_j\rangle + \sum_{i=1}^{N}\alpha_i$$

Sujeto a las restricciones: $0 \le \alpha_i \le C$, $1 \le i \le N$ y $\sum_{i=1}^{N}\alpha_i y_i = 0$

Table of Contents

El truco del Kernel

El truco del Kernel



Tipos de Kernel

- Aunque nuevos kernels aparecen en la literatura, los siguientes cuatro son básicos y ampliamente utilizados:

Lineal:	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$
Polinomial:	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = (\langle \mathbf{x}_i, \mathbf{x}_j \rangle + r)^p, r \geq 0$
Gaussiano (*Radial Basis Function* – RBF):	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(\frac{-\lVert \mathbf{x}_i - \mathbf{x}_j \rVert^2}{2\sigma^2}\right) = \exp\left(-\gamma \lVert \mathbf{x}_i - \mathbf{x}_j \rVert^2\right)$
Sigmoide:	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\gamma \langle \mathbf{x}_i, \mathbf{x}_j \rangle + r)$

donde r, p, γ son parámetros de los modelos.

Table of Contents

Matriz de Confusión

		Predicted condition	
	Total population = P + N	**Positive (PP)**	**Negative (PN)**
Actual condition	**Positive (P)**	**True positive (TP)**	**False negative (FN)**
	Negative (N)	**False positive (FP)**	**True negative (TN)**

Métricas de desempeño

- **Accuracy**: De todos la población, ¿cuántos predije correctamente?
$$A = \frac{TP + TN}{\text{Total}}.$$
- **Recall**: De todos la población positiva, ¿cuántos predije correctamente como positivos?
$$R = \frac{TP}{TP + FN}.$$
- **Precision**: De todos los que predije como positivos, ¿cuántos son realmente positivos?
$$P = \frac{TP}{TP + FP}.$$
- **F1 score**: Media armónica de la precisión y el recall:
$$2\frac{P \cdot R}{P + R}$$

Ejemplo

Tenemos la siguiente población $\{++----\}$:

- El clasificador predice todo como $-$:

real	$+$	$+$	-	-	-	-
predicho	-	-	-	-	-	-

Accuracy: 0.66, Recall: 0, Precision: 0.

- El clasificador predice todo como $+$:

real	$+$	$+$	-	-	-	-
predicho	$+$	$+$	$+$	$+$	$+$	$+$

Accuracy: 0.33, Recall: 1, Precision: 0.33.