



MÁQUINAS DE VECTORES DE SOPORTE

APRENDIZAJE AUTOMÁTICO - CEIOT - FIUBA

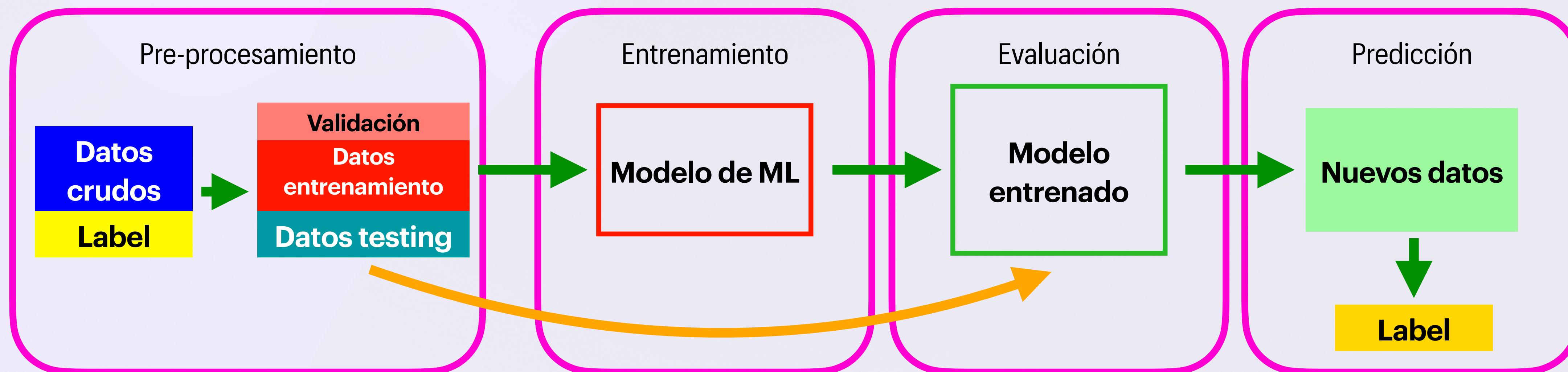
Dr. Ing. Facundo Adrián Lucianna

REPASO CLASE ANTERIOR

- Conjunto de validación
- Validación cruzada
- Arboles de decision
 - Arboles de regresión
 - Arboles de clasificación
- Bosques aleatorios

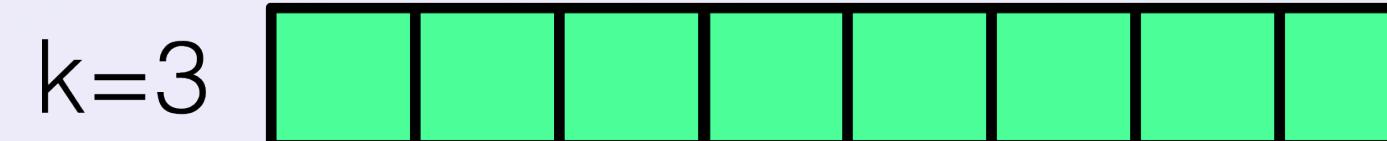
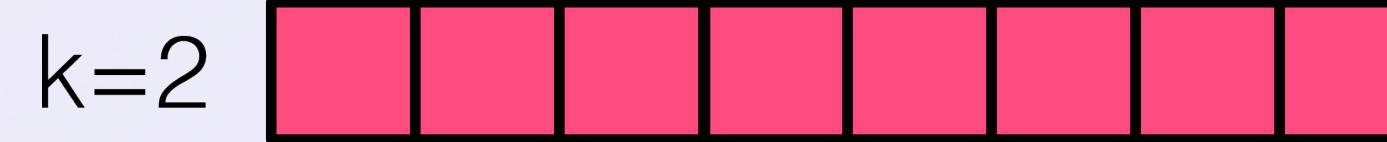
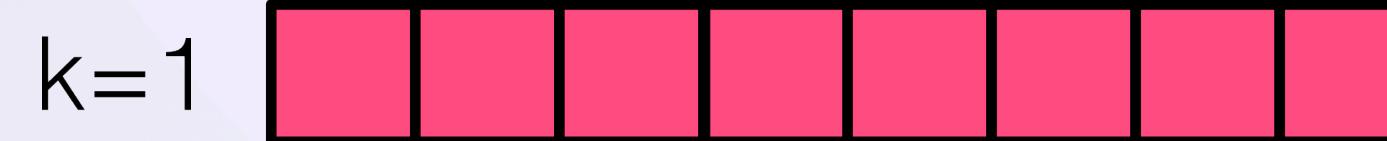
PROCESO DE MACHINE LEARNING

Cuando vimos esto, a propósito deje afuera un conjunto de datos... **Conjunto de validación**

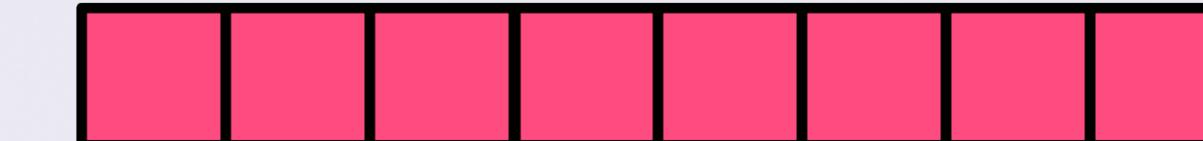
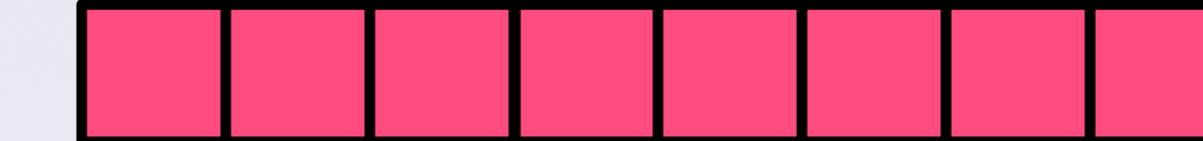


VALIDACIÓN CRUZADA

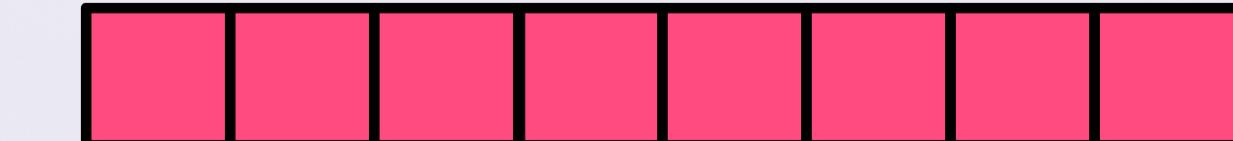
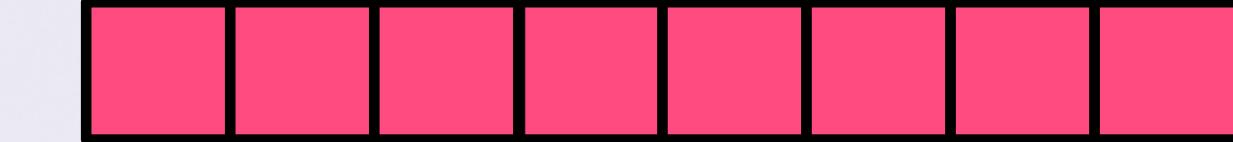
Nos sirve principalmente para buscar hiperparámetros



MAE_1



MAE_2



MAE_3

MAE

ARBOLES DE DECISIÓN

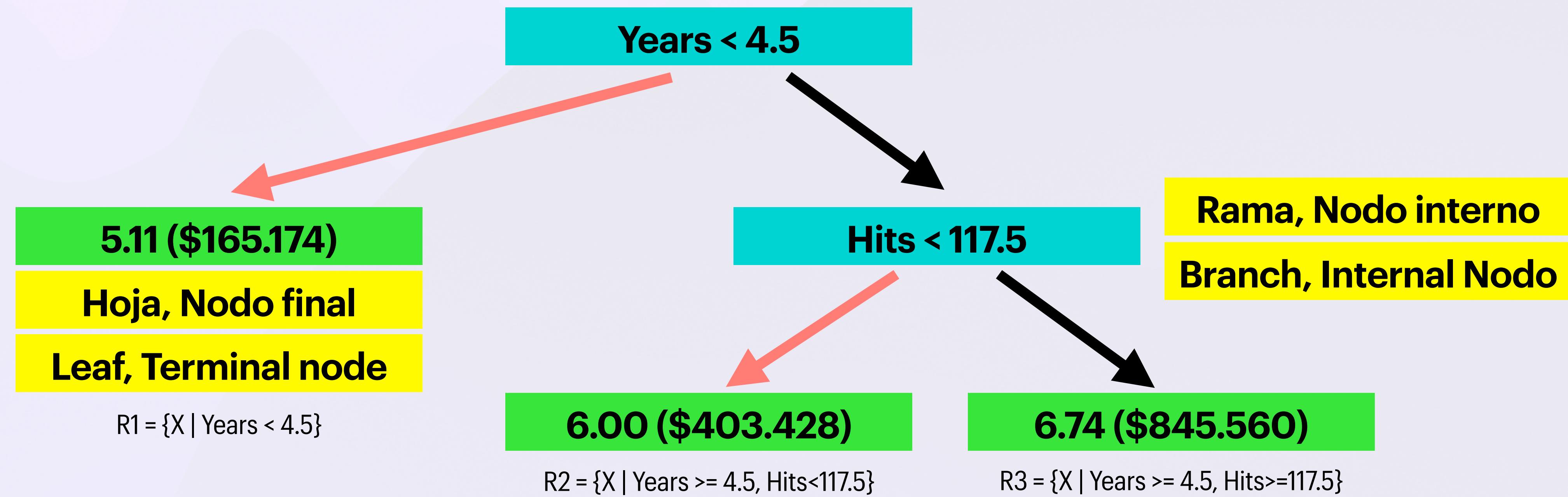
Los árboles de clasificación y regresión, conocidos como **CART** (Classification and Regression Trees), son una poderosa técnica de aprendizaje automático que se utiliza ampliamente para resolver problemas tanto de clasificación como de regresión.

Los árboles CART son modelos de decisión que utilizan una estructura de árbol para realizar predicciones basadas en reglas **lógicas sencillas y fáciles de interpretar**.

[Arboles de clasificación](#)

[Arboles de regresión](#)

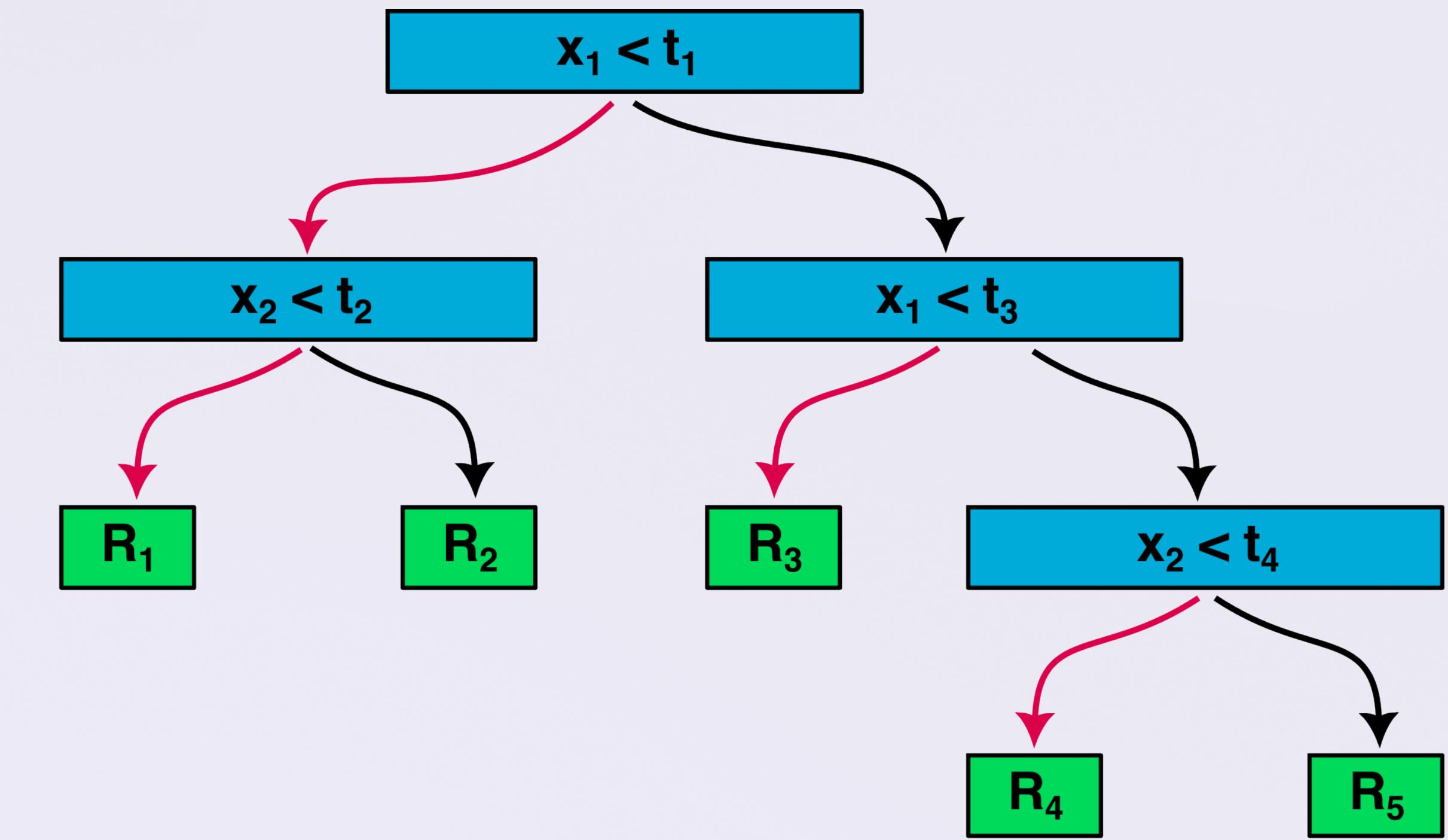
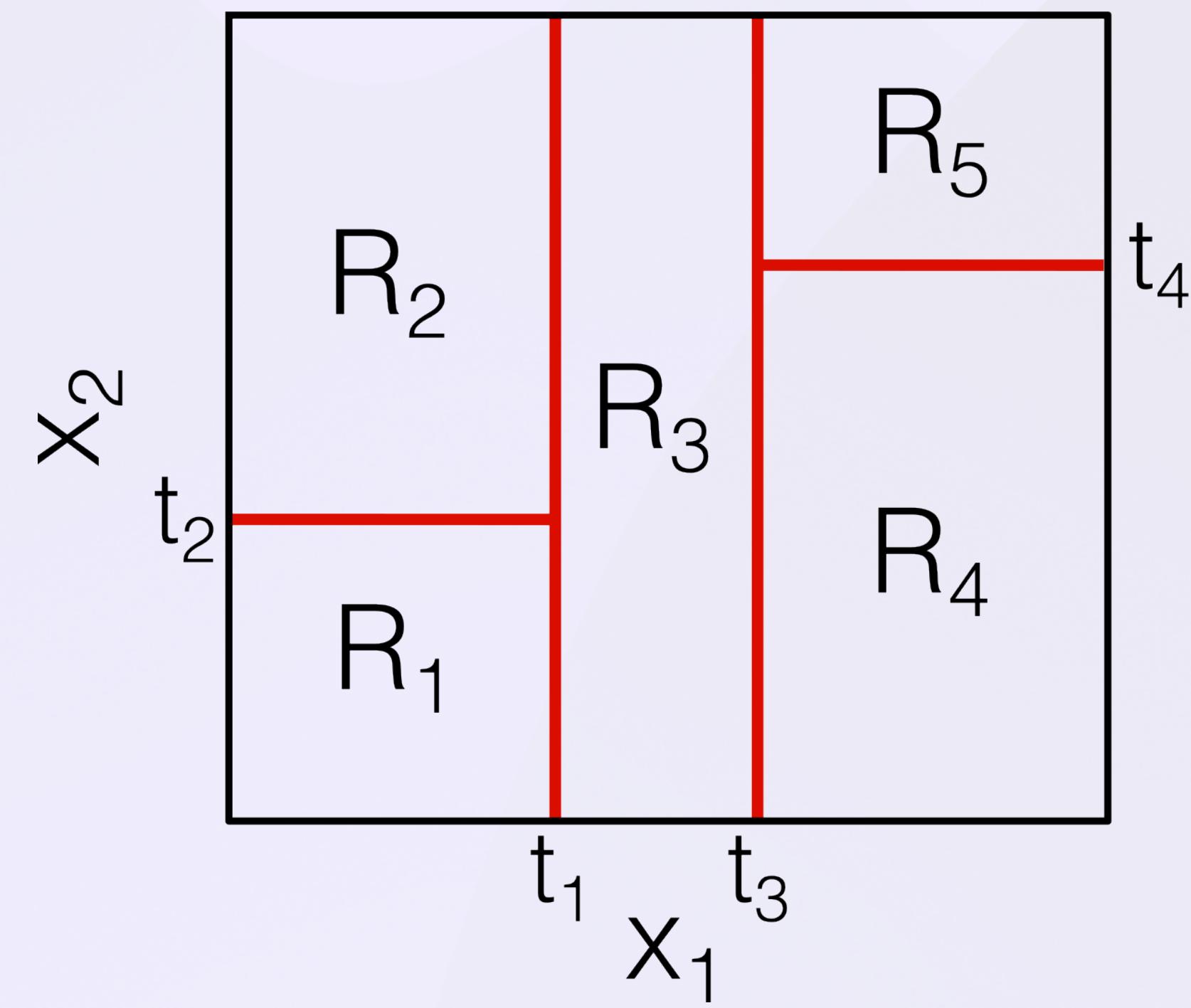
ARBOLES DE REGRESIÓN



ARBOLES DE REGRESIÓN

Recursive binary splitting

$$RSS = \sum_{j=1}^J \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$$



ARBOLES DE REGRESIÓN

Podando los arboles

Introducimos un valor de penalización α a nuestra formula de RSS. Dado un valor de α , existe un sub-árbol T perteneciente a T_0 que:

$$\sum_{m=1}^L \sum_{i \in R_j} (y_i - \hat{y}_{R_m})^2 + \alpha L$$

...es minimo.

L indica el numero de hojas del sub-árbol.

α presenta un trade-off entre complejidad del árbol y su capacidad de ajustar a los datos.

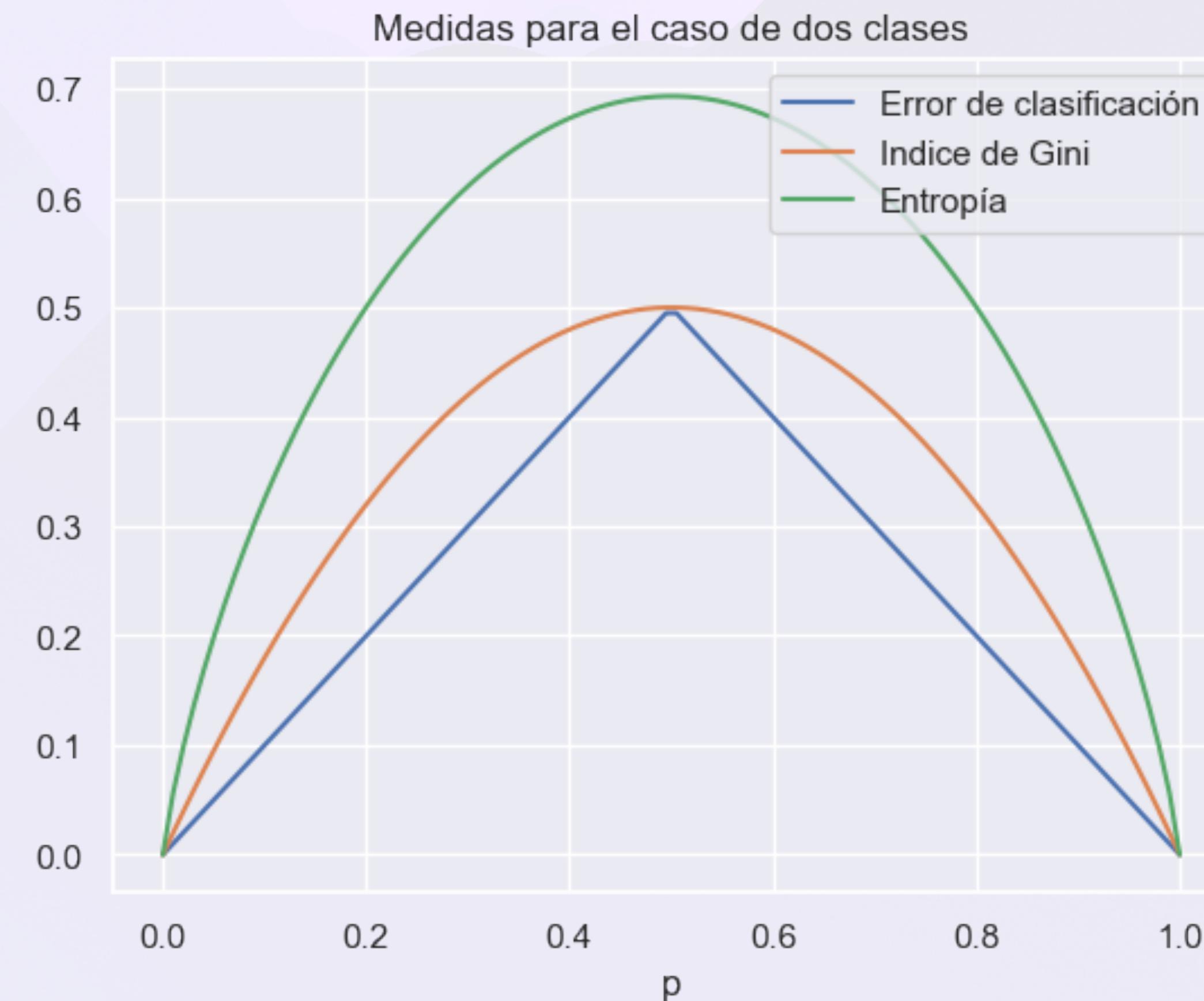
ARBOL DE CLASIFICACIÓN

Un árbol de clasificación es muy similar a uno de regresión, pero ahora se usa para predecir una **variable cualitativa**.

Obtenemos la clase en base a la **clase que más ocurre** en las muestras que están en la hoja.

Al interpretar los resultados de un **árbol de clasificación**, a menudo estamos interesados no sólo en la predicción de clase correspondiente a una región de nodo terminal particular, sino también en las **proporciones de clase entre las observaciones de entrenamiento** que caen en esa región.

ARBOL DE CLASIFICACIÓN



BOSQUES ALEATORIOS

Dado que tan exacto tienden a adaptarse a los datos de entrenamiento, es común que los arboles sobreajusten.

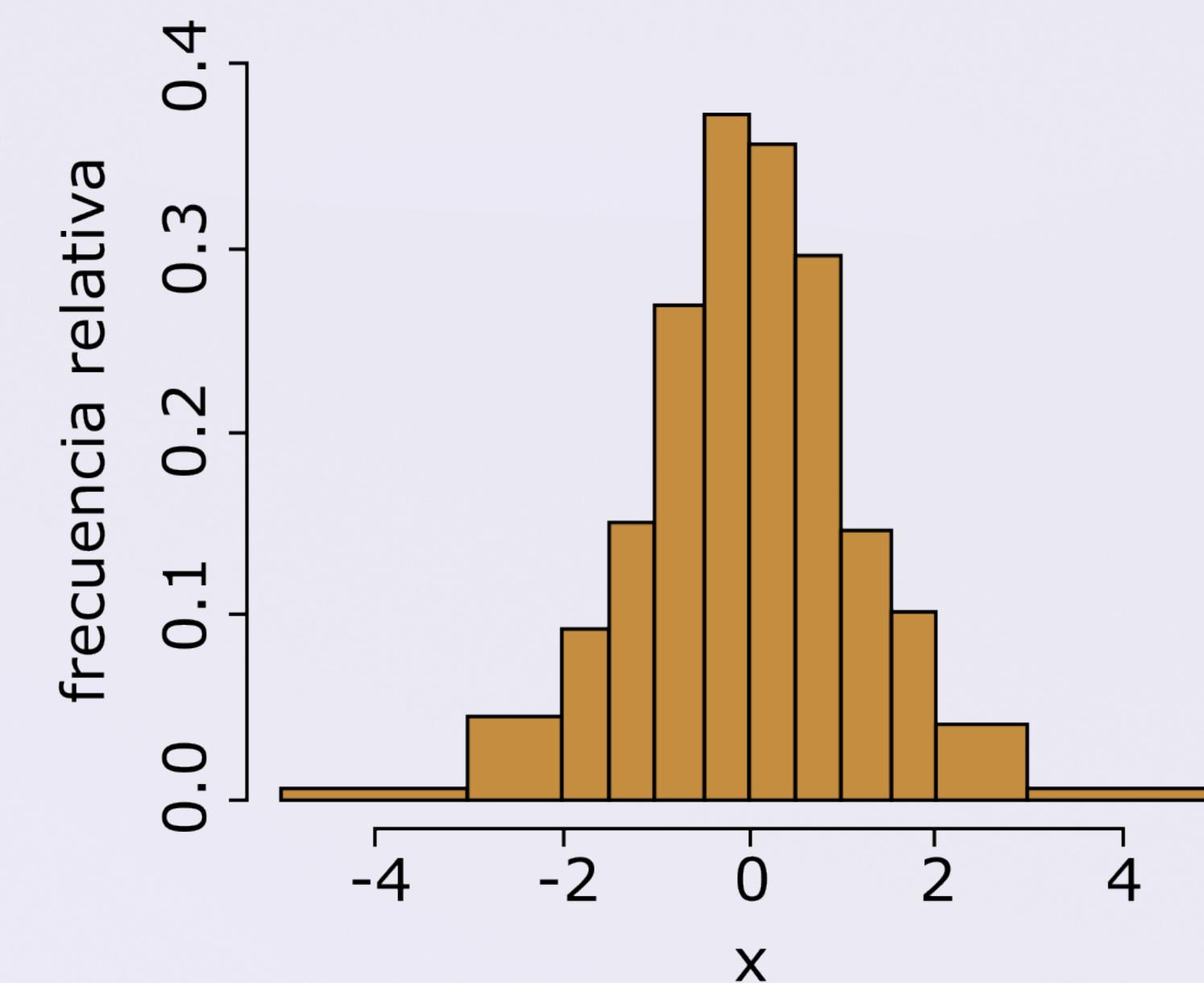
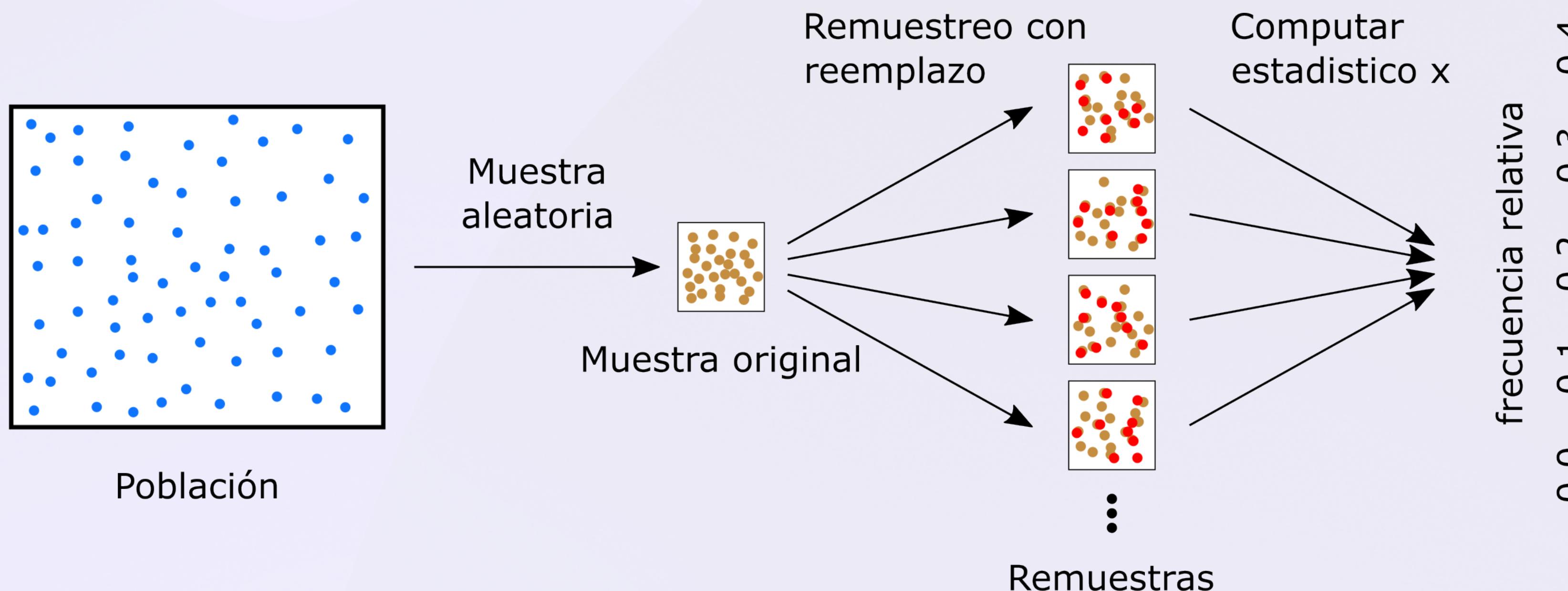
Una forma de evitar esto es mediante bosques aleatorios, en el cual se construyen múltiples arboles de decisión y combinar sus salidas.

Si son de **clasificación**, estos arboles votan la clase.

Si son de **regresión**, se promedia las predicciones.

BOSQUES ALEATORIOS

Bagging: En vez de entrenar a los arboles con todas las observaciones del set de entrenamiento, se arma un nuevo set para cada árbol, usando **Bootstrapping**.



BOSQUES ALEATORIOS

Bagging: En vez de entrenar a los arboles con todas las observaciones del set de entrenamiento, se arma un nuevo set para cada árbol, usando **Bootstrapping**.

Bosques aleatorios: Los bosques aleatorios hace lo mismo que mediante *Bagging*, pero además se usa una cantidad aleatoria de atributos menor a la cantidad original. Arboles chicos.

Boosting: En vez de construir arboles aleatoriamente dado por el proceso de Bootstrapping, los nuevos arboles que se construyen usando la información de arboles anteriores, usando los residuos. Arboles chicos.



MÉTODOS DE AJUSTE DE LOS HIPERPARÁMETROS

MÉTODOS DE AJUSTE DE LOS HIPERPARÁMETROS

La clase anterior vimos validación cruzada para buscar ayudarnos a buscar los hiperparámetros que mejor se nos ajustan a nuestros modelos, permitiendo mantener la **generalidad**.

Pero por si solo, no alcanza, necesitamos de alguna forma “**movernos**” por el espacio de búsqueda.

Una búsqueda consiste en:

- Un modelo (de regresión o clasificación)
- Un espacio de parámetros
- Un método de búsqueda o de muestreo de candidatos
- Un esquema de validación cruzada
- Una función de puntaje

MÉTODOS DE AJUSTE DE LOS HIPERPARÁMETROS

La clase anterior vimos validación cruzada para buscar ayudarnos a buscar los hiperparámetros que mejor se nos ajustan a nuestros modelos, permitiendo mantener la **generalidad**.

Pero por si solo, no alcanza, necesitamos de alguna forma “**movernos**” por el espacio de búsqueda.

Una búsqueda consiste en:

- Un modelo (de regresión o clasificación)
- Un espacio de parámetros
- **Un método de búsqueda o de muestreo de candidatos**
- Un esquema de validación cruzada
- Una función de puntaje

MÉTODOS DE AJUSTE DE LOS HIPERPARÁMETROS

Dos métodos de búsqueda típicos:

Búsqueda de grilla: Busca exhaustiva de toda las combinaciones de los parámetros. Es el mas completo pero el mas ineficiente.

Búsqueda aleatoria: Busca aleatoriamente tomando datos del espacio de combinaciones de los parámetros, bajo una distribución aleatoria dada. Termina dado una cierta cantidad arbitraria de iteraciones.

MÉTODOS DE AJUSTE DE LOS HIPERPARÁMETROS

Una mejora que se puede aplicar a ambos métodos, a expensas de una mayor duración, es incorporar la reducción a la mitad sucesiva (successive halving).

La idea es tener una especie de torneo. En el cual, se arranca buscando los hiperparámetros pero usando un subset de entrenamiento chico.

Una vez que termina la búsqueda, se elige la mitad de las combinaciones que mejor parámetros dieron. Con esa mitad, se aumenta el set de entrenamiento y este proceso se repite, hasta que quede uno.

MAXIMAL MARGIN CLASSIFIER

MAXIMAL MARGIN CLASSIFIER

Empecemos con álgebra, en un espacio p-dimensional, un hiperplano es un subespacio de dimensión p-1.

Por ejemplo,

- En 2 dimensiones, el hiperplano es la recta
- En 3 dimensiones, el hiperplano es el plano

La definición matemática ya la conocemos:

2 dimensiones:

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0$$

$\beta_0, \beta_1, \beta_2 \in \mathbb{R}$

p-dimensiones:

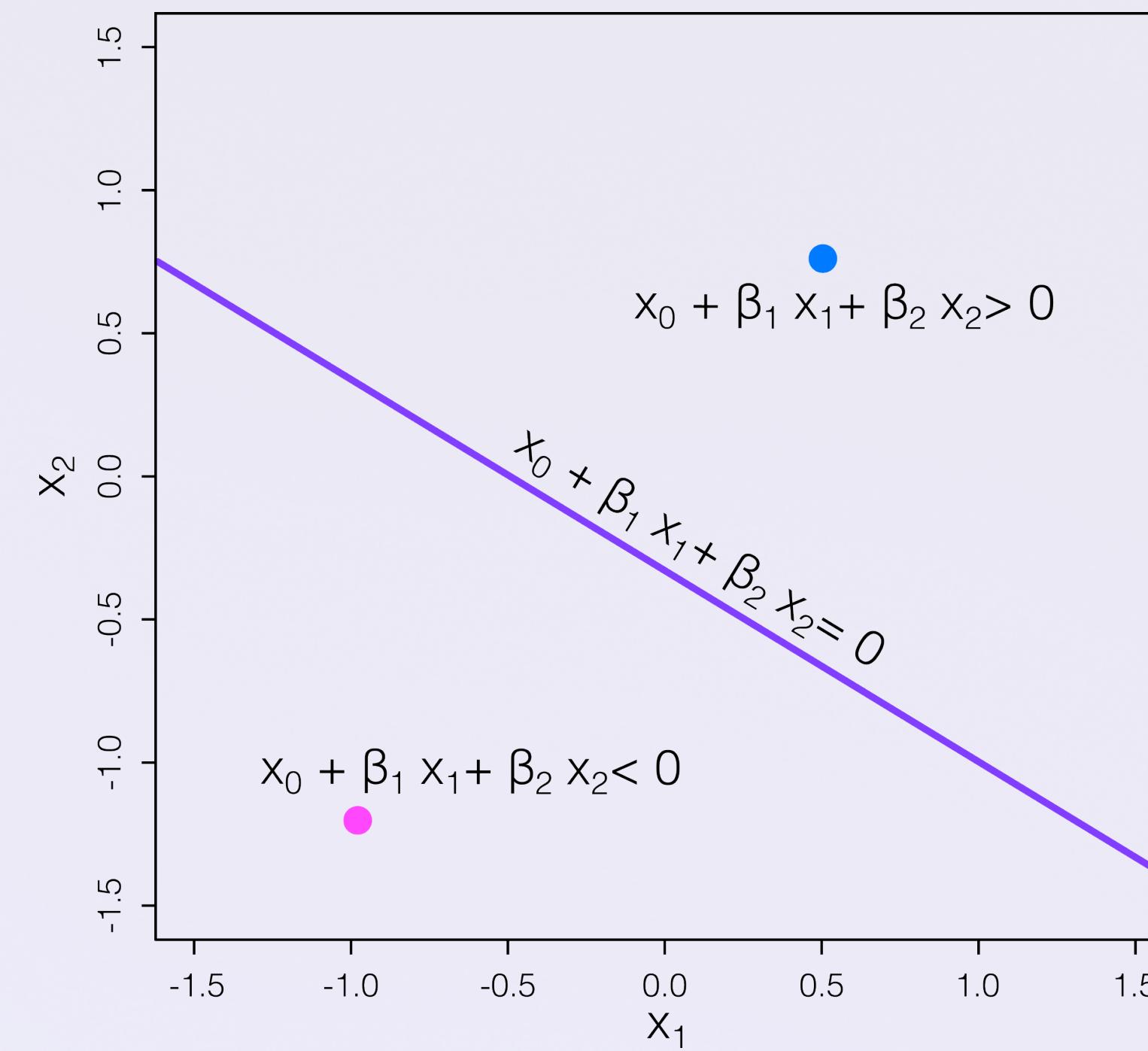
$$\beta_0 + \beta_1 x_1 + \dots + \beta_p x_p = 0$$

$\beta_0, \beta_1, \dots, \beta_p \in \mathbb{R}$

MAXIMAL MARGIN CLASSIFIER

Dado un punto $X = (x_1, x_2, \dots, x_p)$, si verifica la ecuación, decimos que pertenece al hiperplano.

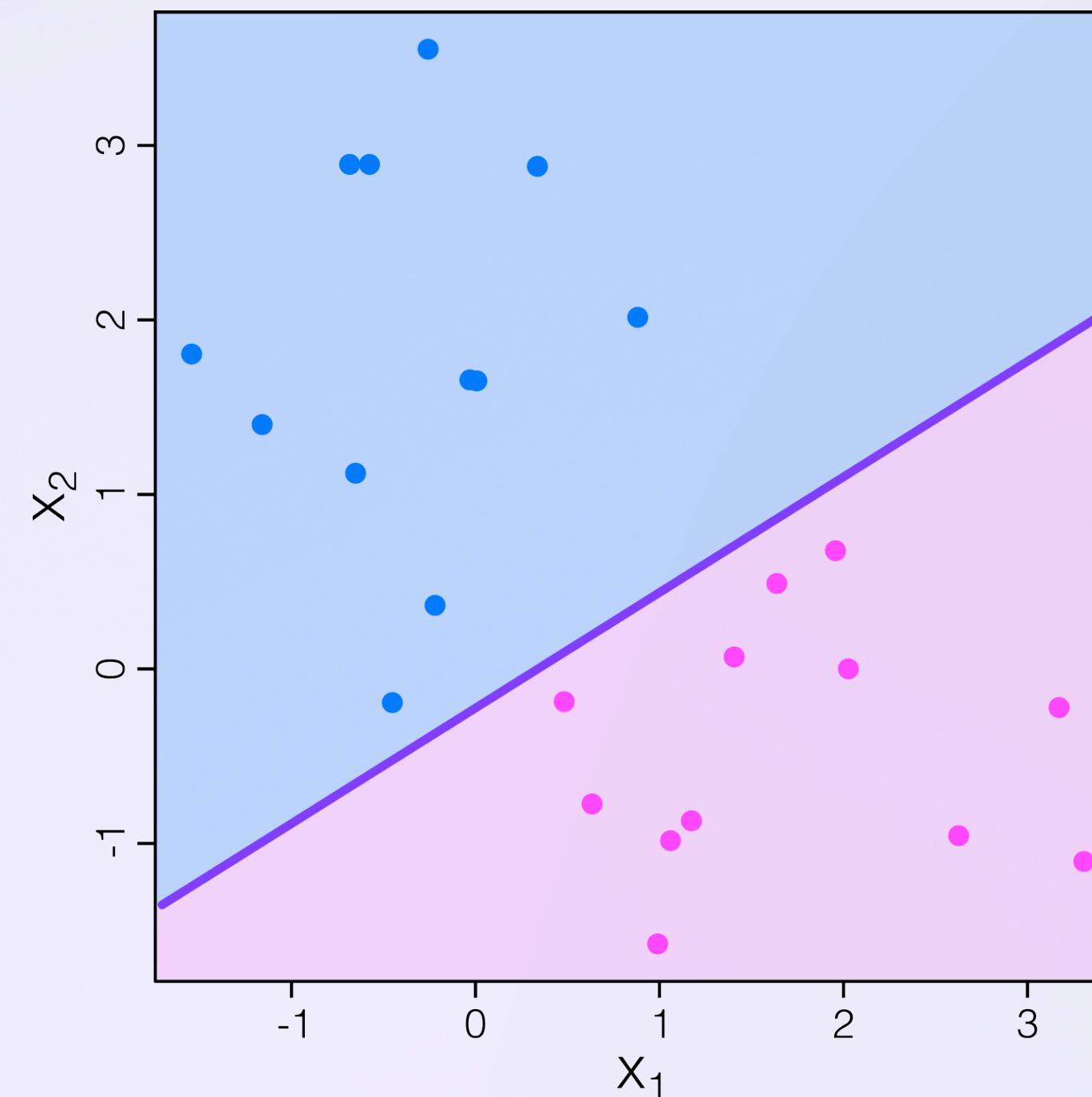
Pero mas interesante, si el punto hace que,



MAXIMAL MARGIN CLASSIFIER

Podemos usar esta idea para clasificar, si tenemos observaciones con p atributos, y estas caen en dos posibles clases $\{-1, 1\}$.

Si de alguna forma, podemos construir un hiperplano que separa los datos de entrenamiento **perfectamente** de acuerdo a su clase, podríamos clasificar como:



$$\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip} > 0 \quad \text{Si } y_i = 1$$

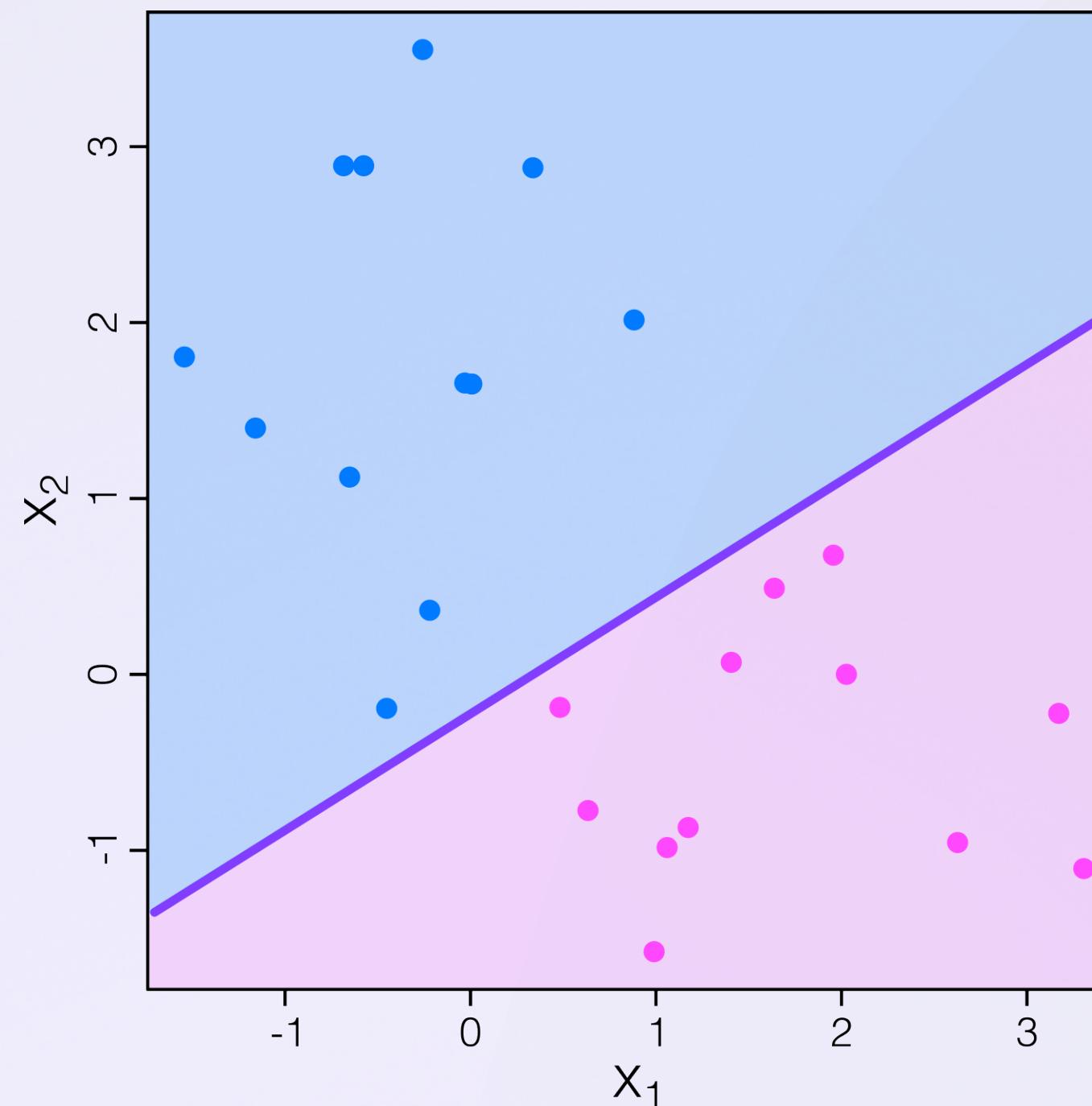
$$\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip} < 0 \quad \text{Si } y_i = -1$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) > 0 \quad i = 1, \dots, n$$

MAXIMAL MARGIN CLASSIFIER

Podemos usar esta idea para clasificar, si tenemos observaciones con p atributos, y estas caen en dos posibles clases $\{-1, 1\}$.

Si de alguna forma, podemos construir un hiperplano que separa los datos de entrenamiento **perfectamente** de acuerdo a su clase, podríamos clasificar como:



Además, podemos definir a $f(X)$ que nos servirá a clasificar:

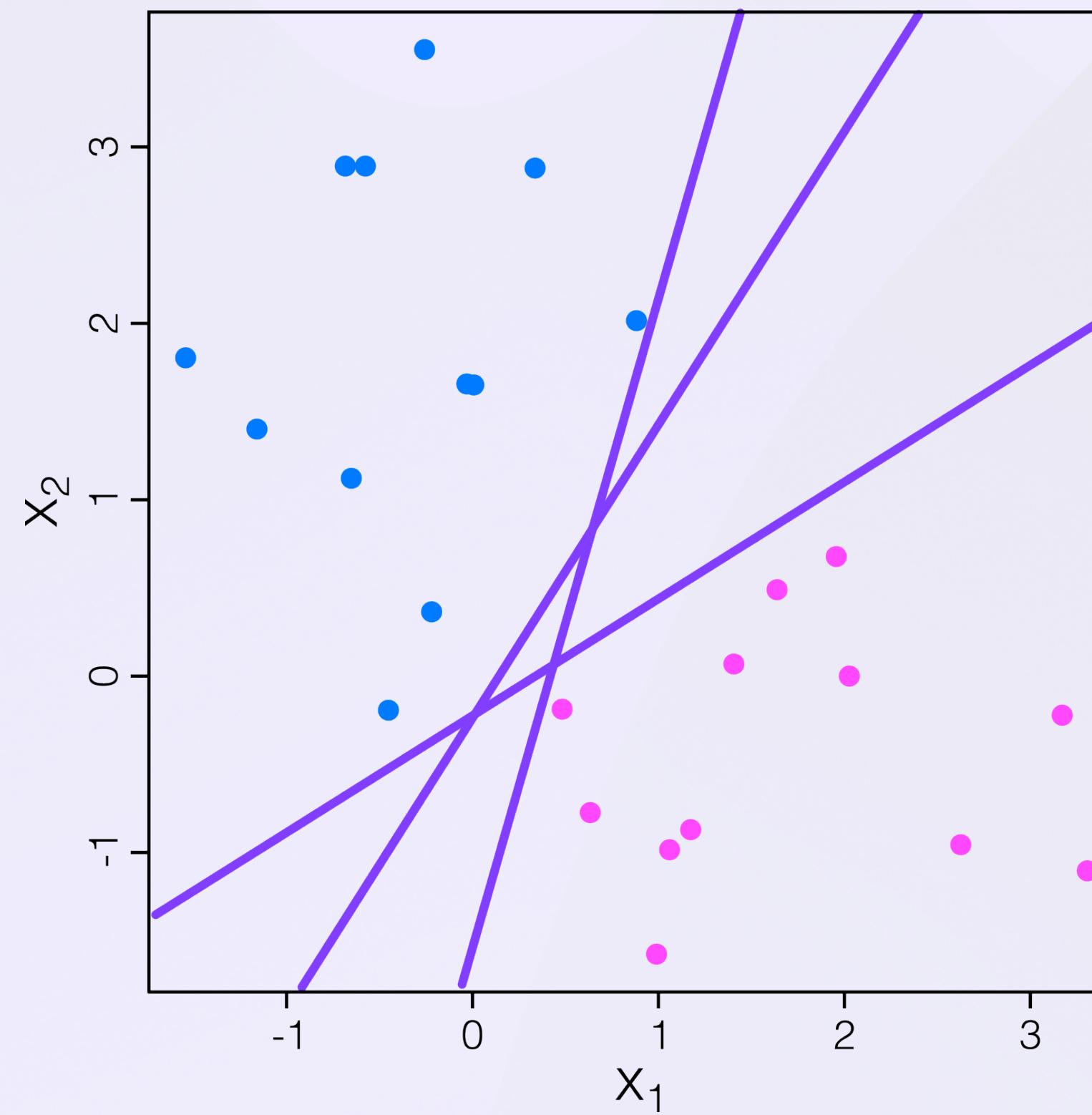
$$f(X_i) = \beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip} > 0 \text{ entonces } \tilde{y}_i = 1$$

$$f(X_i) = \beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip} < 0 \text{ entonces } \tilde{y}_i = -1$$

Y además nos permite ver la confidencia del clasificador, cuando más **grande** es el valor absoluto de $f(X)$, más **segura** es la clasificación

MAXIMAL MARGIN CLASSIFIER

En general si nuestra data es linealmente separable, puede existir un numero infinito de hiperplanos que van a funcionar



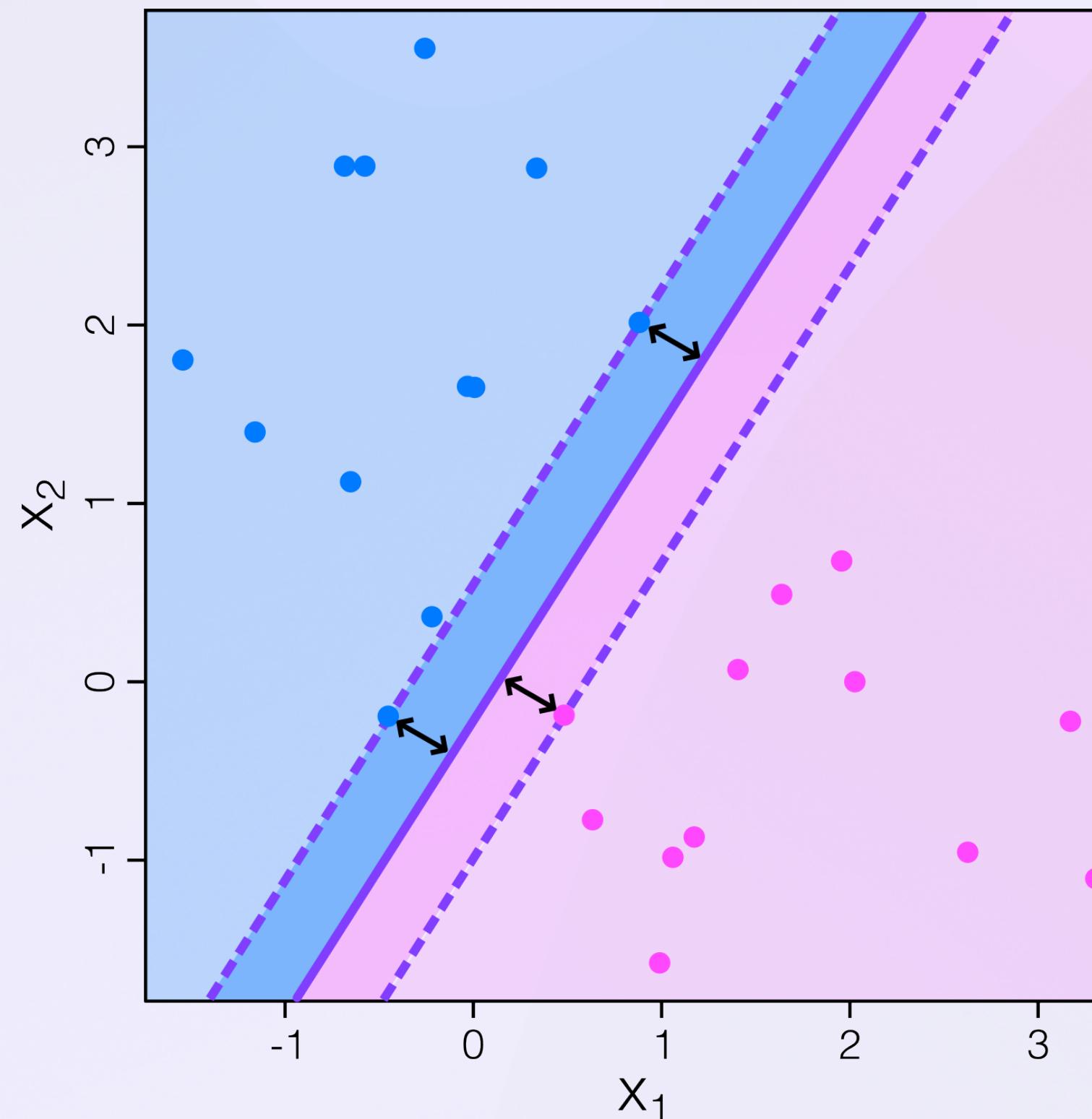
Por lo que necesitamos algún criterio de selección.

El caso que aquí estamos viendo busca el hiperplano que más lejos se encuentra de los datos de entrenamiento.

Es decir, computamos la distancia mínima de cada observación de entrenamiento y obtenemos la distancia más chica de las distancias, que llamamos **margen**.

MAXIMAL MARGIN CLASSIFIER

En general si nuestra data es linealmente separable, puede existir un numero infinito de hiperplanos que van a funcionar



Por lo que necesitamos algún criterio de selección.

El caso que aquí estamos viendo busca el hiperplano que más lejos se encuentra de los datos de entrenamiento.

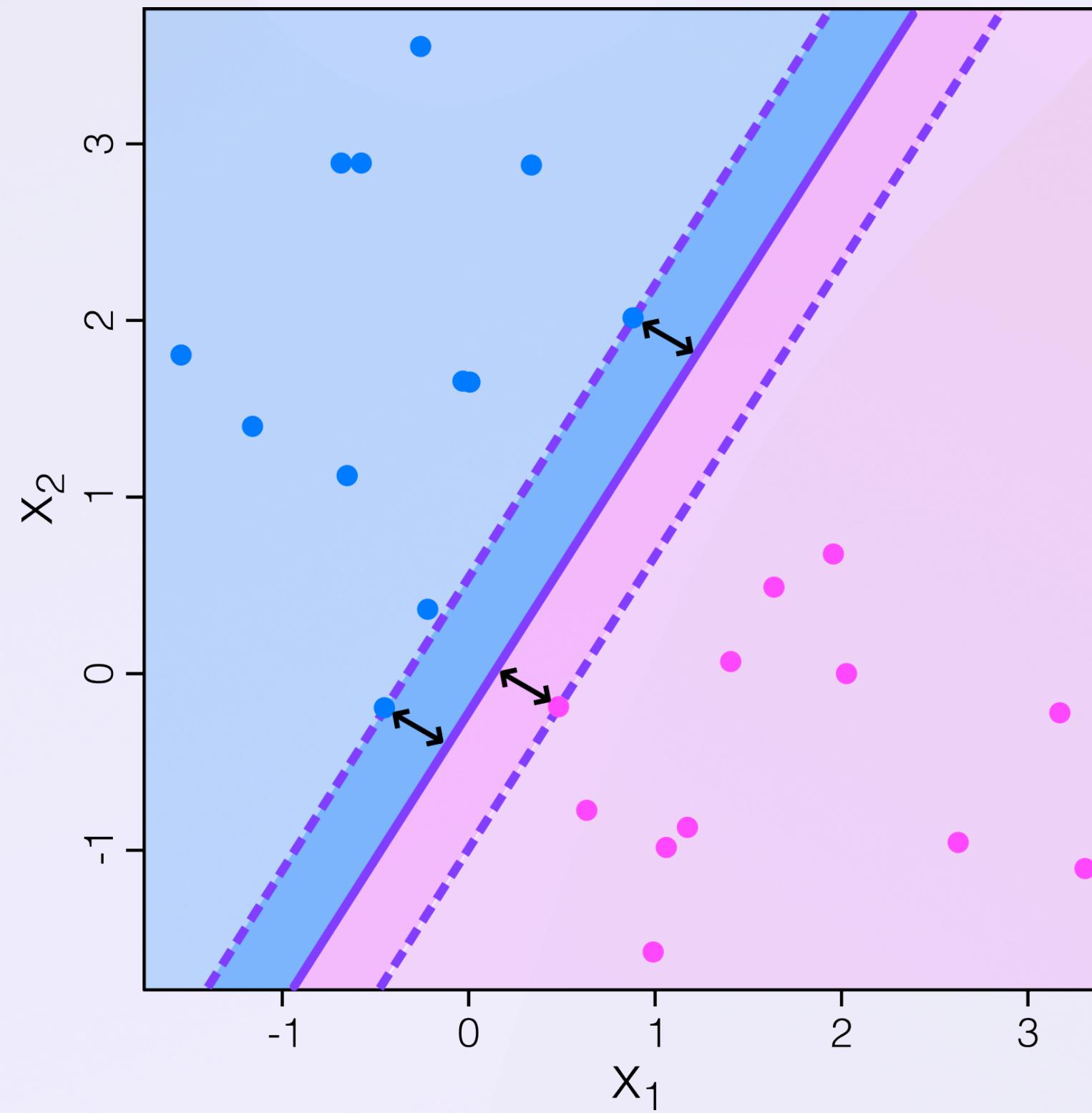
Es decir, computamos la distancia mínima de cada observación de entrenamiento y obtenemos la distancia más chica de las distancias, que llamamos **margen**.

El objetivo es buscar el hiperplano que mas grande posee este margen. Y el algoritmo que hace esto es el **Maximal Margin Classifier**

Podemos pensar que el clasificador busca el máximo **grosor** de recta que puede pasar entre las clases

MAXIMAL MARGIN CLASSIFIER

En este ejemplo, vemos que hay tres observaciones que están equidistante desde el hiperplano de máximo margen.



Estas tres observaciones son conocidas como **vectores de soportes**, dado que soportan el hiperplano.

Esto es porque si uno de estos puntos se mueve aunque sea un poco, el hiperplano también se va a mover.

El hiperplano depende solo de esos tres puntos y no del resto de las observaciones.

MAXIMAL MARGIN CLASSIFIER

Para construir este hiperplano, necesitamos resolver el problema de optimización:

$$\underset{\beta_0, \beta_1, \dots, \beta_p, M}{\text{maximizar}} \quad M$$

$$\text{sujeto a} \quad \sum_{j=1}^p \beta_j^2 = 1$$

$$y_i (\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M \quad \forall i = 1, 2, \dots, n$$

MAXIMAL MARGIN CLASSIFIER

Para construir este hiperplano, necesitamos resolver el problema de optimización:

$$\underset{\beta_0, \beta_1, \dots, \beta_p, M}{\text{maximizar}} \quad M$$

$$\text{sujeto a} \quad \sum_{j=1}^p \beta_j^2 = 1$$

$$\begin{aligned} \exists \beta_0, \dots, \beta_p : \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p &= 0 \\ \Rightarrow \kappa(\beta_0 + \beta_1 x_1 + \dots + \beta_p x_p) &= 0 \quad \kappa \in \mathbb{R} \end{aligned}$$

Con esta restricción (usamos un versor) aseguramos el valor de M y ademas que:

$$y_i (\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip})$$

para una observación i, es la medida de la distancia perpendicular al hiperplano

$$y_i (\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M \quad \forall i = 1, 2, \dots, n$$

MAXIMAL MARGIN CLASSIFIER

Para construir este hiperplano, necesitamos resolver el problema de optimización:

$$\underset{\beta_0, \beta_1, \dots, \beta_p, M}{\text{maximizar}} \quad M$$

$$\text{sujeto a} \quad \sum_{j=1}^p \beta_j^2 = 1$$

$$\exists \beta_0, \dots, \beta_p : \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p = 0 \\ \Rightarrow \kappa(\beta_0 + \beta_1 x_1 + \dots + \beta_p x_p) = 0 \quad \kappa \in \mathbb{R}$$

Con esta restricción (usamos un versor) aseguramos el valor de M y ademas que:

$$y_i (\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip})$$

para una observación i, es la medida de la distancia perpendicular al hiperplano

$$y_i (\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M \quad \forall i = 1, 2, \dots, n$$

Con estas dos restricciones logramos que cada observación este del lado correcto y al menos a la distancia de M

MAXIMAL MARGIN CLASSIFIER

Para construir este hiperplano, necesitamos resolver el problema de optimización:

$$\underset{\beta_0, \beta_1, \dots, \beta_p, M}{\text{maximizar}} \quad M$$

$$\text{sujeto a} \quad \sum_{j=1}^p \beta_j^2 = 1$$

$$\exists \beta_0, \dots, \beta_p : \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p = 0 \\ \Rightarrow \kappa(\beta_0 + \beta_1 x_1 + \dots + \beta_p x_p) = 0 \quad \kappa \in \mathbb{R}$$

Con esta restricción (usamos un versor) aseguramos el valor de M y ademas que:

$$y_i (\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip})$$

para una observación i, es la medida de la distancia perpendicular al hiperplano

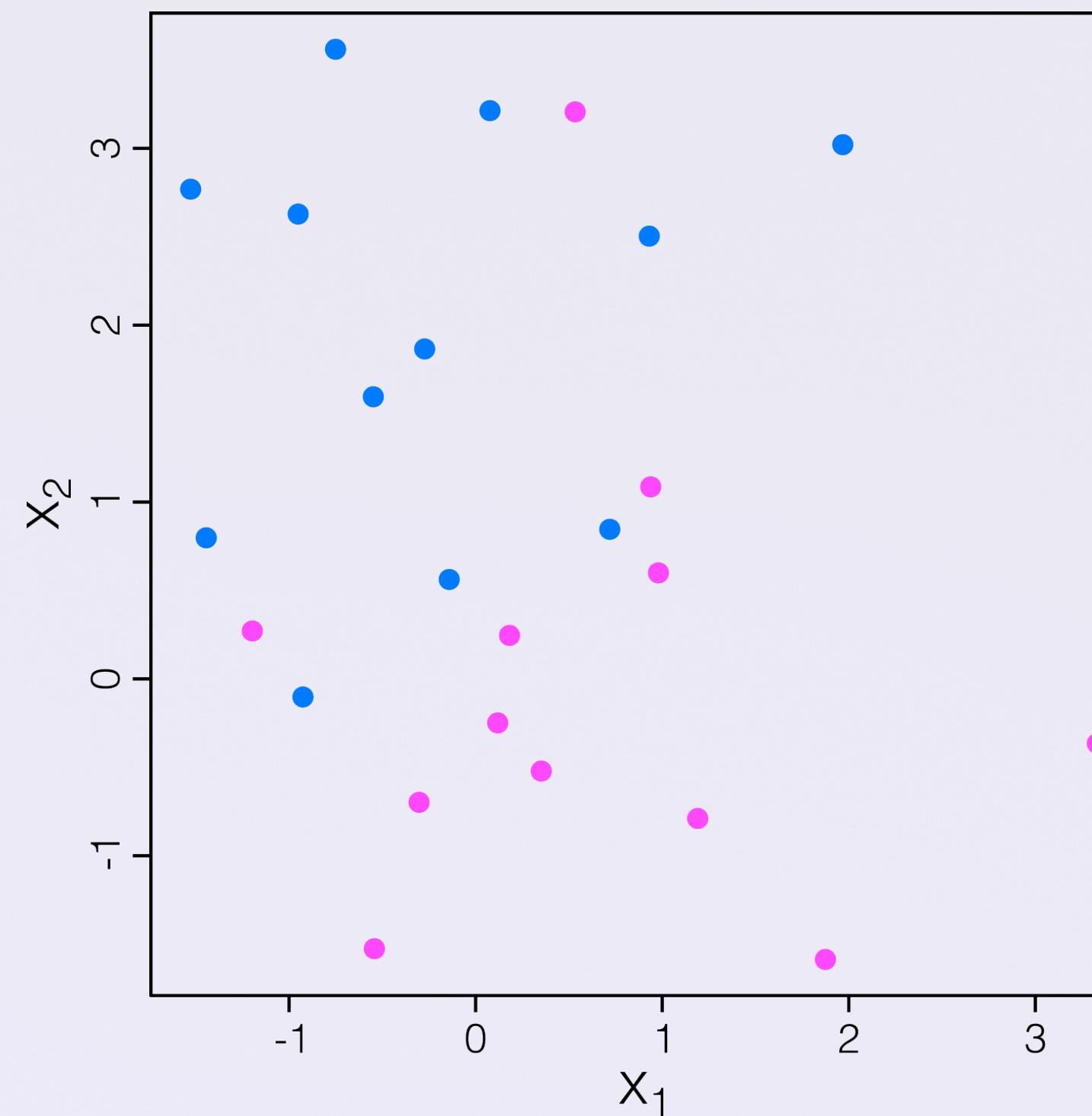
$$y_i (\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M \quad \forall i = 1, 2, \dots, n$$

Con estas dos restricciones logramos que cada observación este del lado correcto y al menos a la distancia de M

El problema es de programación cuadrática y se resuelve mediante multiplicadores de Lagrange.

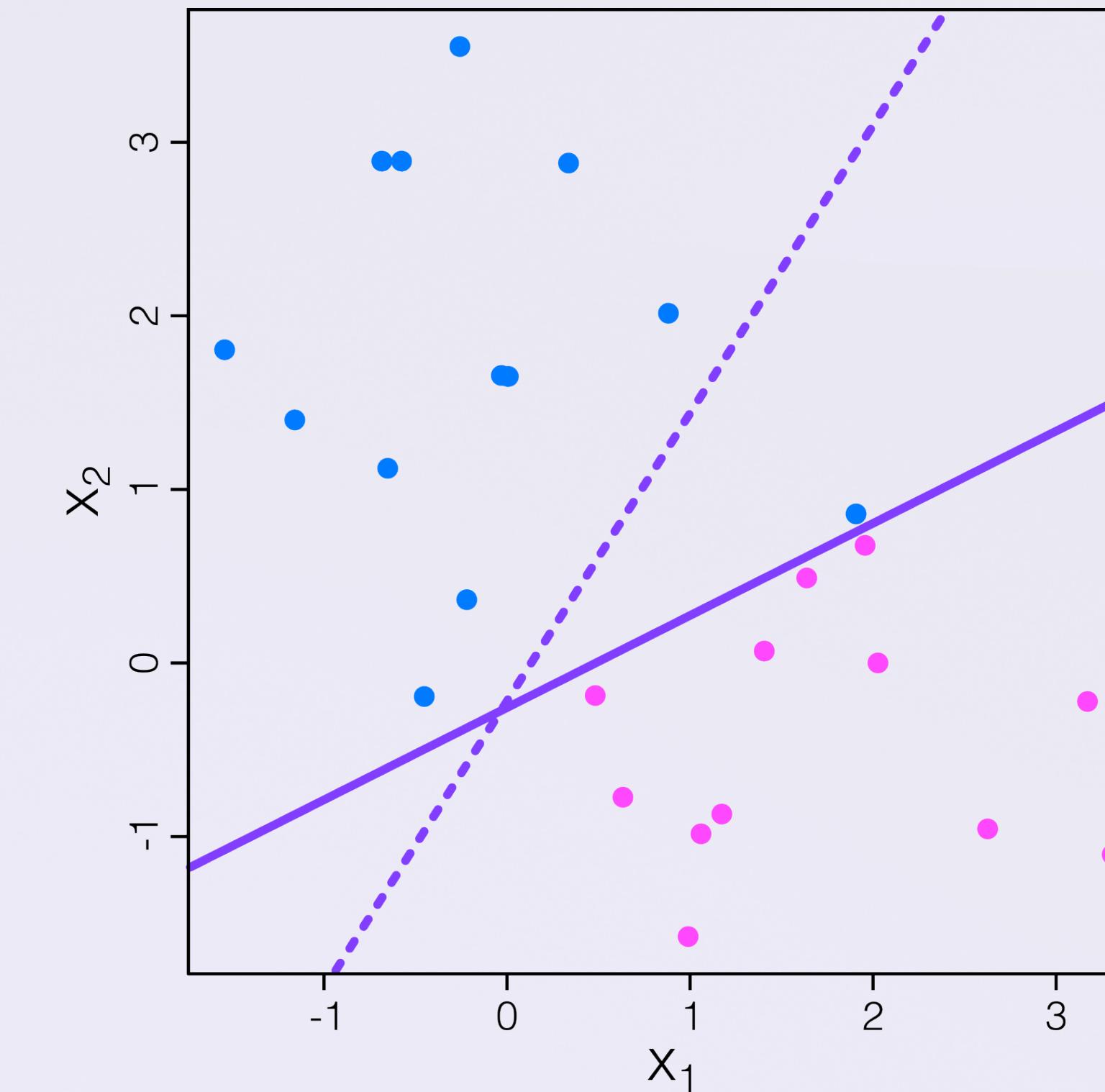
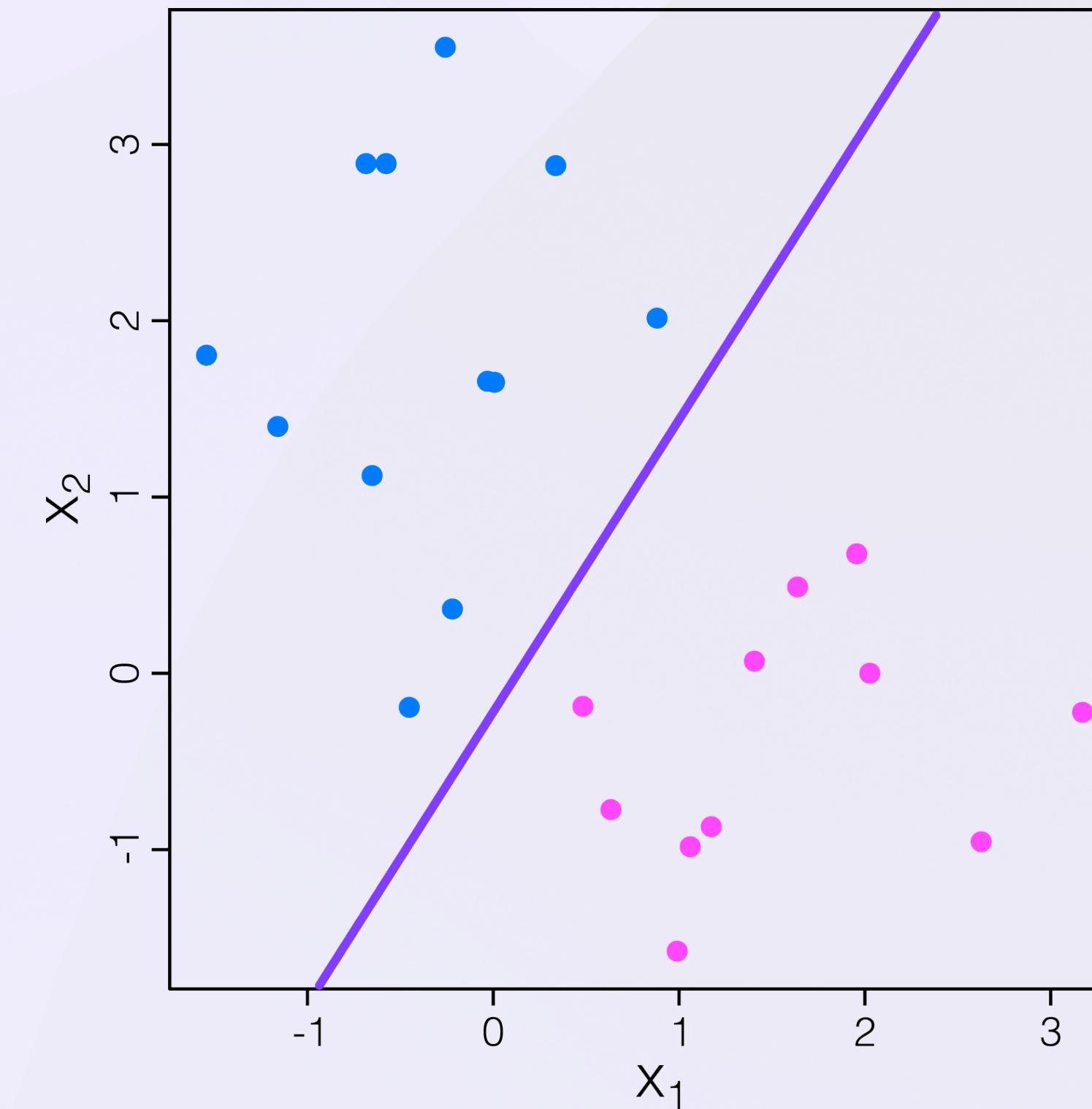
CASO NO SEPARABLE

Este modelo que vimos solo funciona si es posible separar. Si el problema no es separable, no hay hiperplano que maximiza el margen. Y por consiguiente, el problema de optimización no tiene solución...



CASO NO SEPARABLE

No solo eso, el modelo como está planteado es excesivamente sensible a set de entrenamiento, es decir tiene overfitting (error de varianza)... **Necesitamos mejorar el modelo.**



CLASIFICADOR DE VECTOR DE SOPORTES

CLASIFICADOR DE VECTOR DE SOPORTES

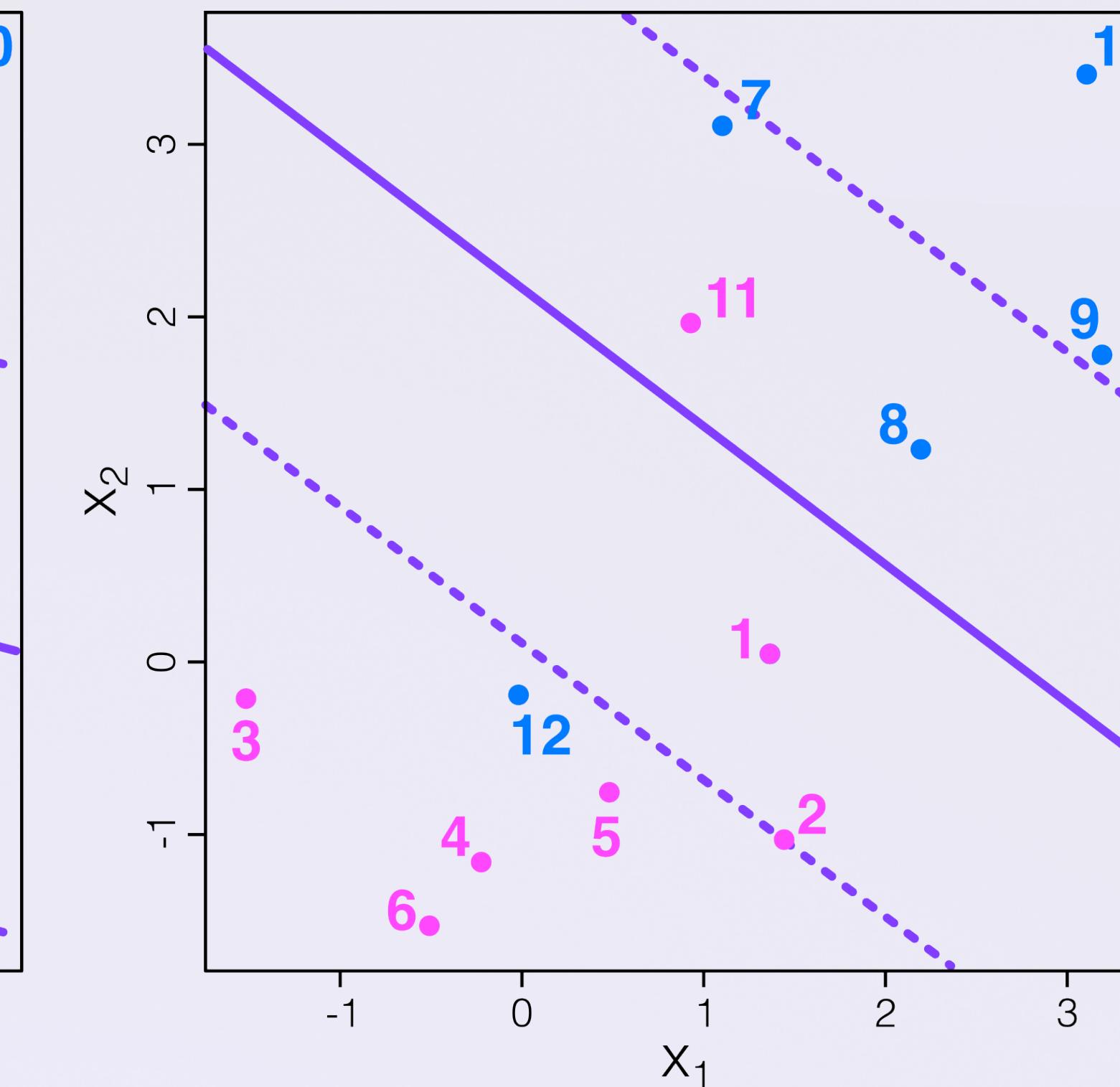
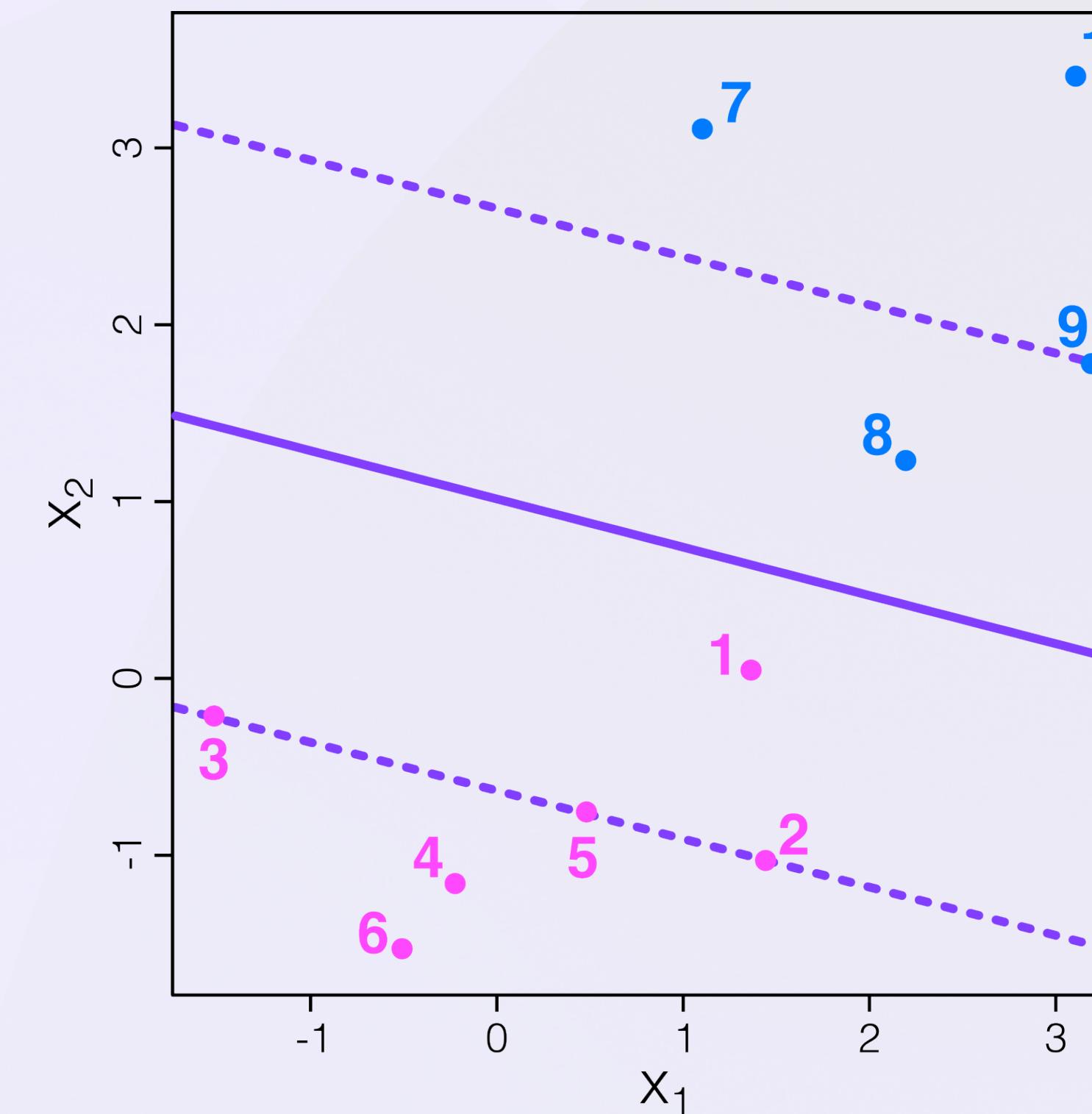
Por lo que vimos, si queremos seguir usando un hiperplano, debemos relajar las exigencias:

- Mayor robustez a observaciones individuales.
- Mejor clasificación de la **mayoría** (no todas) de las observaciones de entrenamiento.

Es decir, podría valer la pena clasificar erróneamente algunas observaciones de entrenamiento para poder clasificar mejor las observaciones restantes.

CLASIFICADOR DE VECTOR DE SOPORTES

El modelo clasificador de vectores de soporte es el modelo apropiado para este caso. Con este modelo una observación puede estar no sólo en el lado equivocado del **margin**, sino también en el **lado equivocado del hiperplano**.



CLASIFICADOR DE VECTOR DE SOPORTES

Para construir este caso, se modifica el problema de optimización de recién:

$$\underset{\beta_0, \beta_1, \dots, \beta_p, M}{\text{maximizar}} \quad M$$

$$\text{sujeto a} \quad \sum_{j=1}^p \beta_j^2 = 1$$

$$y_i (\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M(1 - \epsilon_i) \quad \forall i = 1, 2, \dots, n$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^n \epsilon_i \leq C, \quad C \geq 0$$

CLASIFICADOR DE VECTOR DE SOPORTES

Para construir este caso, se modifica el problema de optimización de recién:

$$\underset{\beta_0, \beta_1, \dots, \beta_p, M}{\text{maximizar}} \quad M$$

$$\text{sujeto a} \quad \sum_{j=1}^p \beta_j^2 = 1$$

$$y_i (\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M(1 - \epsilon_i) \quad \forall i = 1, 2, \dots, n$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^n \epsilon_i \leq C, \quad C \geq 0 \quad \text{Donde } C \text{ es un hiperparámetro.}$$

CLASIFICADOR DE VECTOR DE SOPORTES

Para construir este caso, se modifica el problema de optimización de recién:

$$\underset{\beta_0, \beta_1, \dots, \beta_p, M}{\text{maximizar}} \quad M$$

$$\text{sujeto a} \quad \sum_{j=1}^p \beta_j^2 = 1$$

$$y_i (\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M(1 - \epsilon_i) \quad \forall i = 1, 2, \dots, n$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^n \epsilon_i \leq C, \quad C \geq 0 \quad \text{Donde } C \text{ es un hiperparámetro.}$$

variables que permiten que las observaciones individuales estén en el lado equivocado del margen o el hiperplano

CLASIFICADOR DE VECTOR DE SOPORTES

Para construir este caso, se modifica el problema de optimización de recién:

$$\underset{\beta_0, \beta_1, \dots, \beta_p, M}{\text{maximizar}} \quad M$$

$$\text{sujeto a} \quad \sum_{j=1}^p \beta_j^2 = 1$$

$$y_i (\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M(1 - \epsilon_i) \quad \forall i = 1, 2, \dots, n$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^n \epsilon_i \leq C, \quad C \geq 0 \quad \text{Donde } C \text{ es un hiperparámetro.}$$

variables que permiten que las observaciones individuales estén en el lado equivocado del margen o el hiperplano

Si para una observación i , si $\epsilon_i = 0$, la observación está en el lado correcto.

CLASIFICADOR DE VECTOR DE SOPORTES

Para construir este caso, se modifica el problema de optimización de recién:

$$\underset{\beta_0, \beta_1, \dots, \beta_p, M}{\text{maximizar}} \quad M$$

$$\text{sujeto a} \quad \sum_{j=1}^p \beta_j^2 = 1$$

$$y_i (\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M(1 - \epsilon_i) \quad \forall i = 1, 2, \dots, n$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^n \epsilon_i \leq C, \quad C \geq 0 \quad \text{Donde } C \text{ es un hiperparámetro.}$$

variables que permiten que las observaciones individuales estén en el lado equivocado del margen o el hiperplano

Si para una observación i , si $0 < \epsilon_i < 1$, la observación está en el lado equivocado del margen

CLASIFICADOR DE VECTOR DE SOPORTES

Para construir este caso, se modifica el problema de optimización de recién:

$$\underset{\beta_0, \beta_1, \dots, \beta_p, M}{\text{maximizar}} \quad M$$

$$\text{sujeto a} \quad \sum_{j=1}^p \beta_j^2 = 1$$

$$y_i (\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M(1 - \epsilon_i) \quad \forall i = 1, 2, \dots, n$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^n \epsilon_i \leq C, \quad C \geq 0 \quad \text{Donde } C \text{ es un hiperparámetro.}$$

variables que permiten que las observaciones individuales estén en el lado equivocado del margen o el hiperplano

Si para una observación i , si $\epsilon_i > 1$, la observación está en el lado equivocado del hiperplano

CLASIFICADOR DE VECTOR DE SOPORTES

Para construir este caso, se modifica el problema de optimización de recién:

$$\underset{\beta_0, \beta_1, \dots, \beta_p, M}{\text{maximizar}} \quad M$$

$$\text{sujeto a} \quad \sum_{j=1}^p \beta_j^2 = 1$$

$$y_i (\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M(1 - \epsilon_i) \quad \forall i = 1, 2, \dots, n$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^n \epsilon_i \leq C, \quad C \geq 0 \quad \text{Donde } C \text{ es un hiperparámetro.}$$

C limita la suma de los ϵ_i y, por tanto, determina el número y la gravedad de las violaciones del margen (y del hiperplano) que toleraremos.

CLASIFICADOR DE VECTOR DE SOPORTES

Para construir este caso, se modifica el problema de optimización de recién:

$$\underset{\beta_0, \beta_1, \dots, \beta_p, M}{\text{maximizar}} \quad M$$

$$\text{sujeto a} \quad \sum_{j=1}^p \beta_j^2 = 1$$

$$y_i (\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M(1 - \epsilon_i) \quad \forall i = 1, 2, \dots, n$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^n \epsilon_i \leq C, \quad C \geq 0 \quad \text{Donde } C \text{ es un hiperparámetro.}$$

Si $C > 0$, no más que C observaciones pueden estar en el lado equivocado del hiperplano. Si C es más grande, las exigencias son más laxas y los márgenes serán más grandes.

CLASIFICADOR DE VECTOR DE SOPORTES

Podemos reescribir usando el producto escalar

$$\underset{\beta_0, \beta_1, \dots, \beta_p, M}{\text{maximizar}} \quad M$$

sujeto a $\|\hat{\beta}\| = 1$

$$y_i (\beta_0 + \langle \hat{\beta}, X_i \rangle) \geq M(1 - \epsilon_i) \quad \forall i = 1, 2, \dots, n$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^n \epsilon_i \leq C, \quad C \geq 0$$

CLASIFICADOR DE VECTOR DE SOPORTES

Podemos reescribir usando el producto escalar

$$\underset{\beta_0, \beta_1, \dots, \beta_p, M}{\text{maximizar}} \quad M$$

$$y_i (\beta_0 + \langle \hat{\beta}, X_i \rangle) \geq M(1 - \epsilon_i) \quad \forall i = 1, 2, \dots, n$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^n \epsilon_i \leq C, \quad C \geq 0$$

Si quitamos la restricción del versor, donde definimos a $M = 1/\|\hat{\beta}\|$, debemos modificar a la medida de distancia teniendo en cuenta que ya no es un versor

CLASIFICADOR DE VECTOR DE SOPORTES

Podemos reescribir usando el producto escalar

$$\underset{\beta_0, \beta_1, \dots, \beta_p}{\text{minimizar}} \hat{\beta}$$

$$\frac{y_i (\beta_0 + \langle \hat{\beta}, X_i \rangle)}{\|\hat{\beta}\|} \geq \frac{1 - \epsilon_i}{\|\hat{\beta}\|} \quad \forall i = 1, 2, \dots, n$$
$$\epsilon_i \geq 0, \quad \sum_{i=1}^n \epsilon_i \leq C, \quad C \geq 0$$

Si quitamos la restricción del versor, donde definimos a $M = 1/\|\hat{\beta}\|$, debemos modificar a la medida de distancia teniendo en cuenta que ya no es un versor

CLASIFICADOR DE VECTOR DE SOPORTES

Podemos reescribir usando el producto escalar

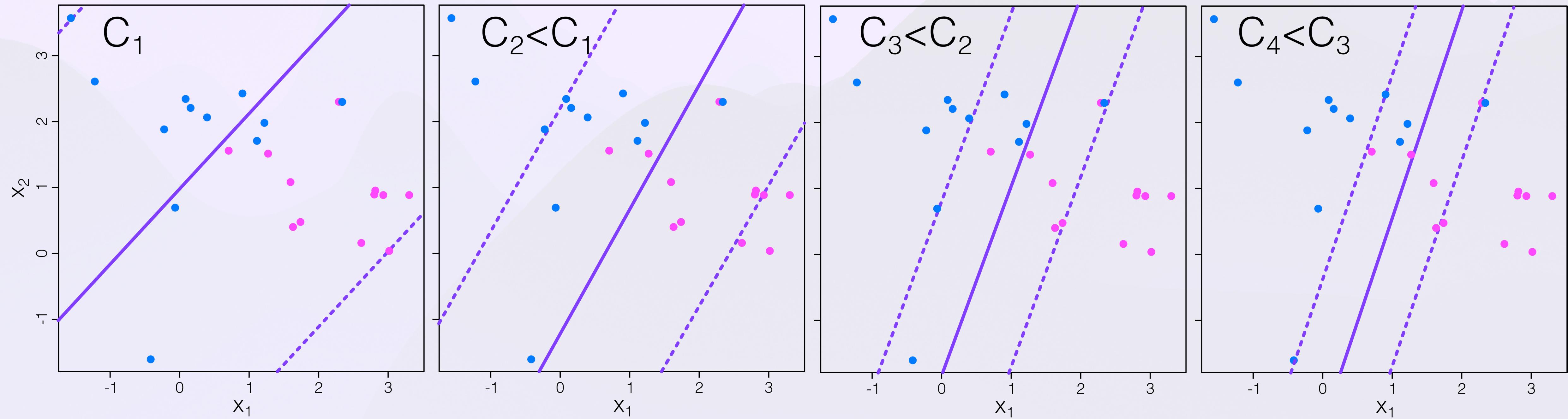
$$\underset{\beta_0, \beta_1, \dots, \beta_p}{\text{minimizar}} \hat{\beta}$$

$$y_i (\beta_0 + \langle \hat{\beta}, X_i \rangle) \geq 1 - \epsilon_i \quad \forall i = 1, 2, \dots, n$$

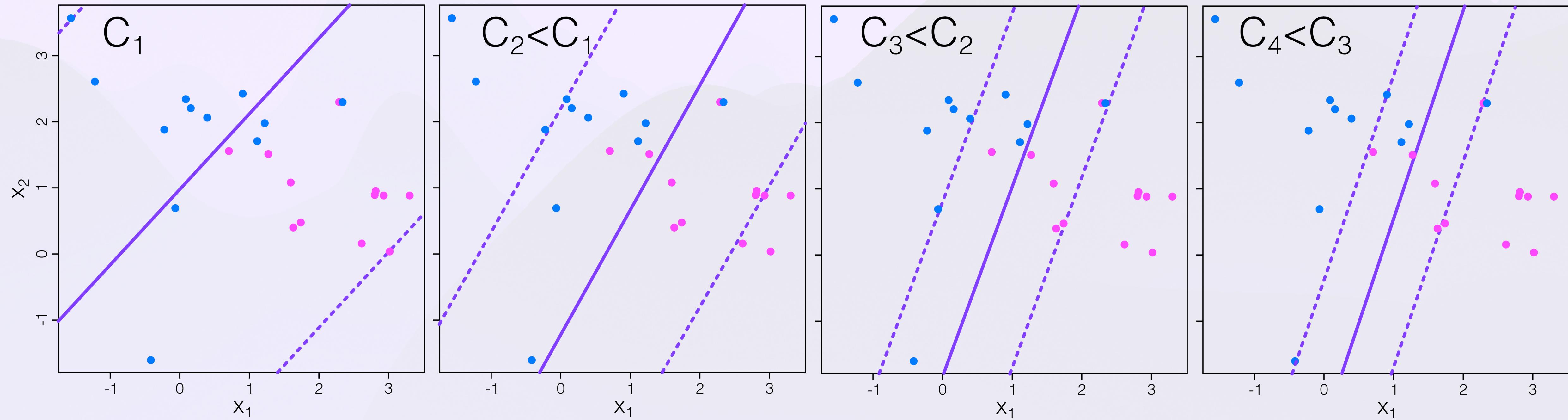
$$\epsilon_i \geq 0, \quad \sum_{i=1}^n \epsilon_i \leq C, \quad C \geq 0$$

Si quitamos la restricción del versor, donde definimos a $M = 1/\|\hat{\beta}\|$, debemos modificar a la medida de distancia teniendo en cuenta que ya no es un versor

CLASIFICADOR DE VECTOR DE SOPORTES



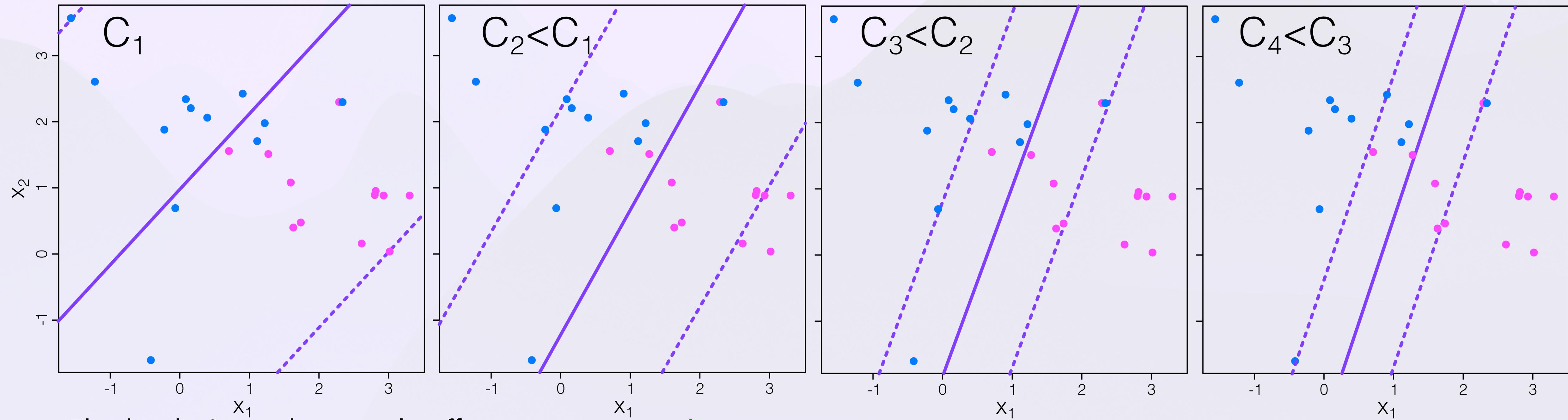
CLASIFICADOR DE VECTOR DE SOPORTES



Hay un pequeño numero de observaciones en el margen o dentro de él, que son los que determinan el hiperplano, esto se llaman **vectores de soporte**.

Las demás observaciones no tienen importancia para el modelo.

CLASIFICADOR DE VECTOR DE SOPORTES



El valor de C nos da un trade-off entre **sesgo** y **varianza**.

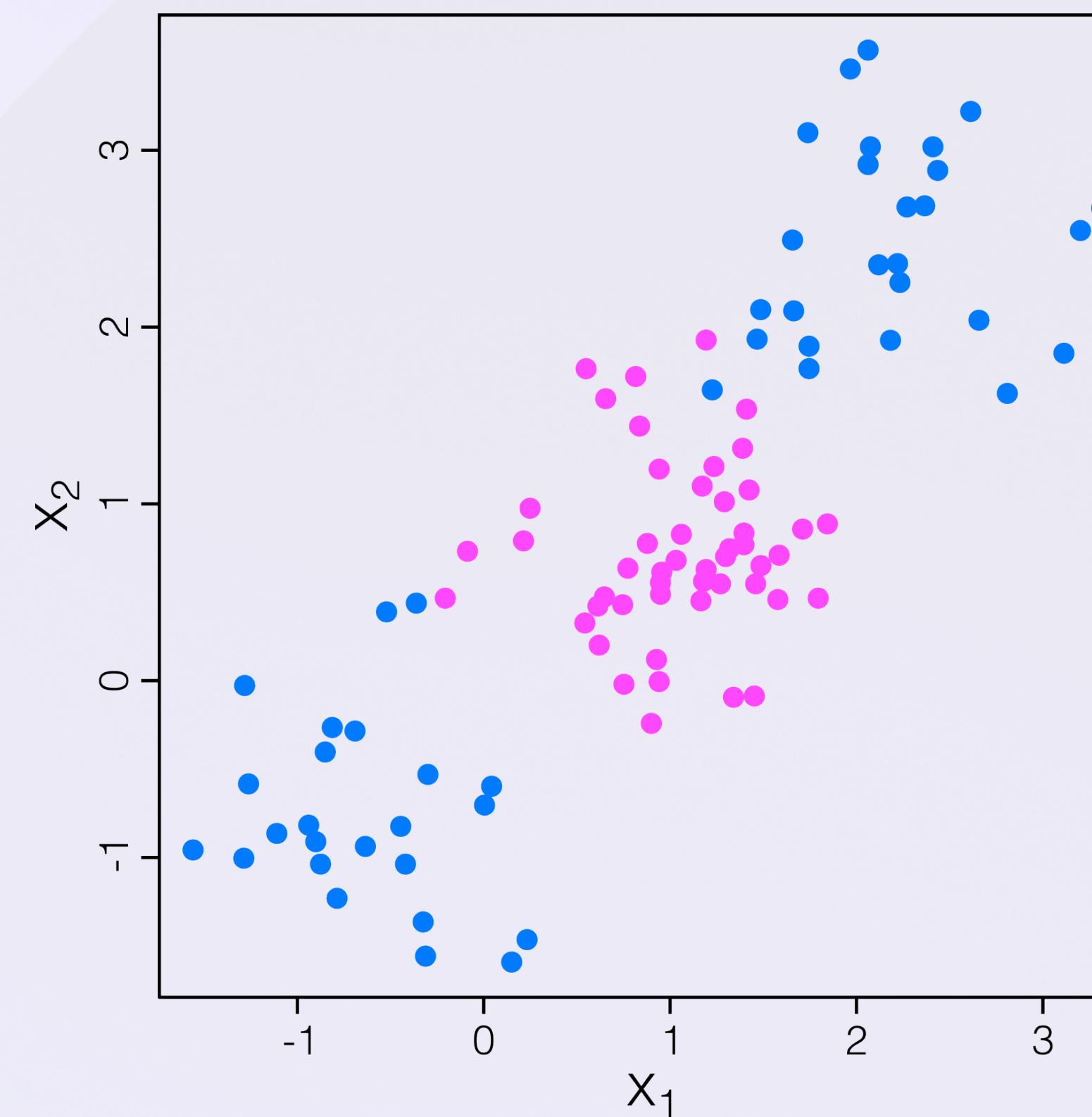
- Si C es grande, entonces el margen es grande y hay muchos vectores de soporte. Este clasificador tiene poca varianza pero potencialmente poca exactitud.
- Si C es pequeño, hay menos vectores de soporte, y por consiguiente, más varianza a expensas de una potencial mejor exactitud.



SUPPORT VECTOR MACHINE

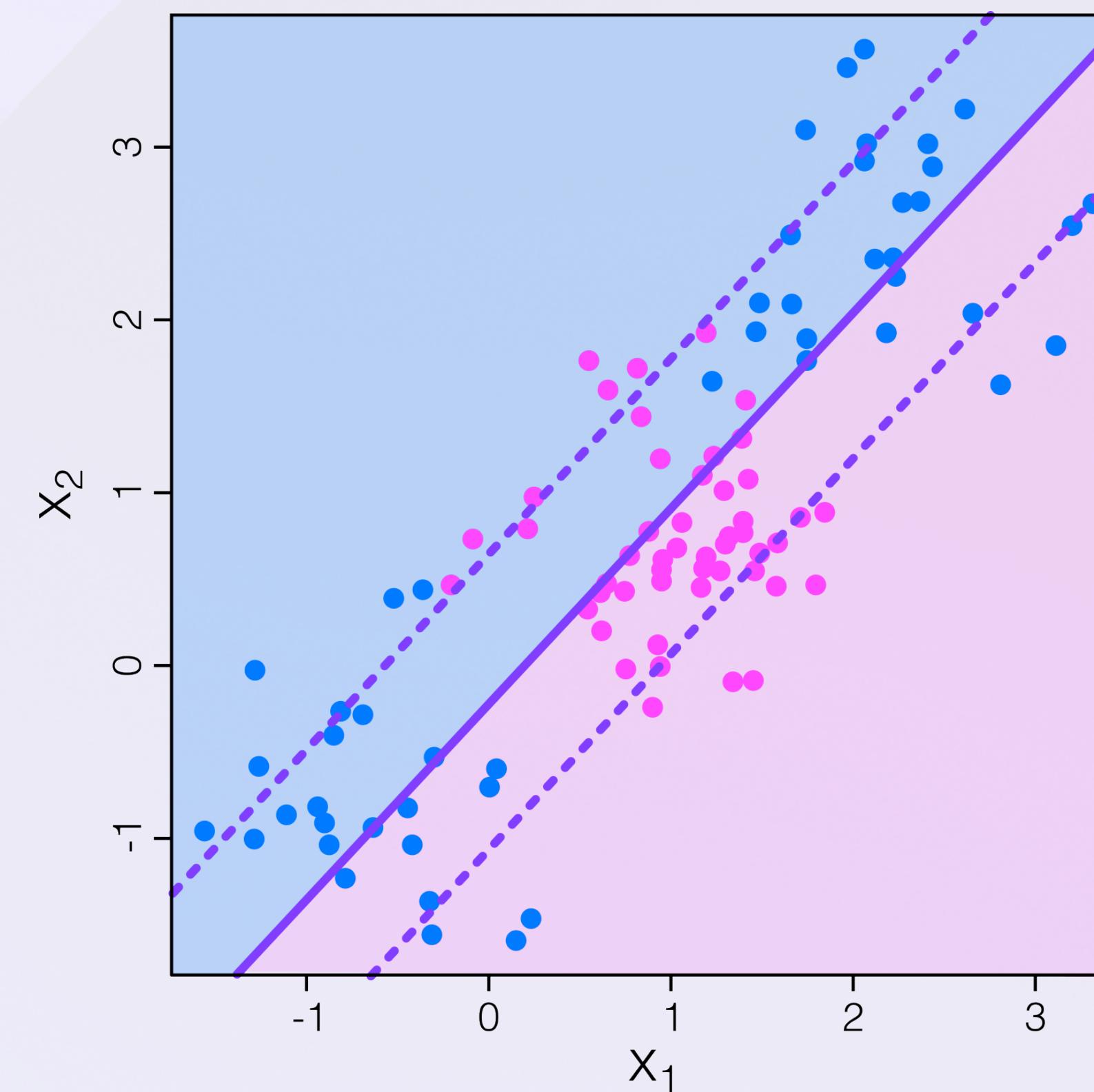
SUPPORT VECTOR MACHINE

El clasificador que hemos visto hasta ahora solo funciona para casos de separación lineal, pero en casos de fronteras no lineales es completamente inútil.



SUPPORT VECTOR MACHINE

El clasificador que hemos visto hasta ahora solo funciona para casos de separación lineal, pero en casos de fronteras no lineales es completamente inútil.



SUPPORT VECTOR MACHINE

Podríamos, similar al caso de regresión lineal y polinomial, modificar a nuestros predictores, agregando la versión cuadrática, cúbica, etc.

Es decir, en vez de para cada observación entrenar con los atributos x_1, x_2, \dots, x_p

Entrenamos con $x_1, x_1^2, x_2, x_2^2, x_2^3, \dots, x_p, x_p^2$

El modelo será lineal pero en el espacio extendido, pero cuando volvamos al espacio dado por x_1, x_2, \dots, x_p , la frontera de decisión será polinómica.

Esto lo podríamos llevar a otro tipo de relaciones, así podemos usar otro tipo de fronteras, pero el problema es que el problema de ajuste se vuelve **inmanejable**.

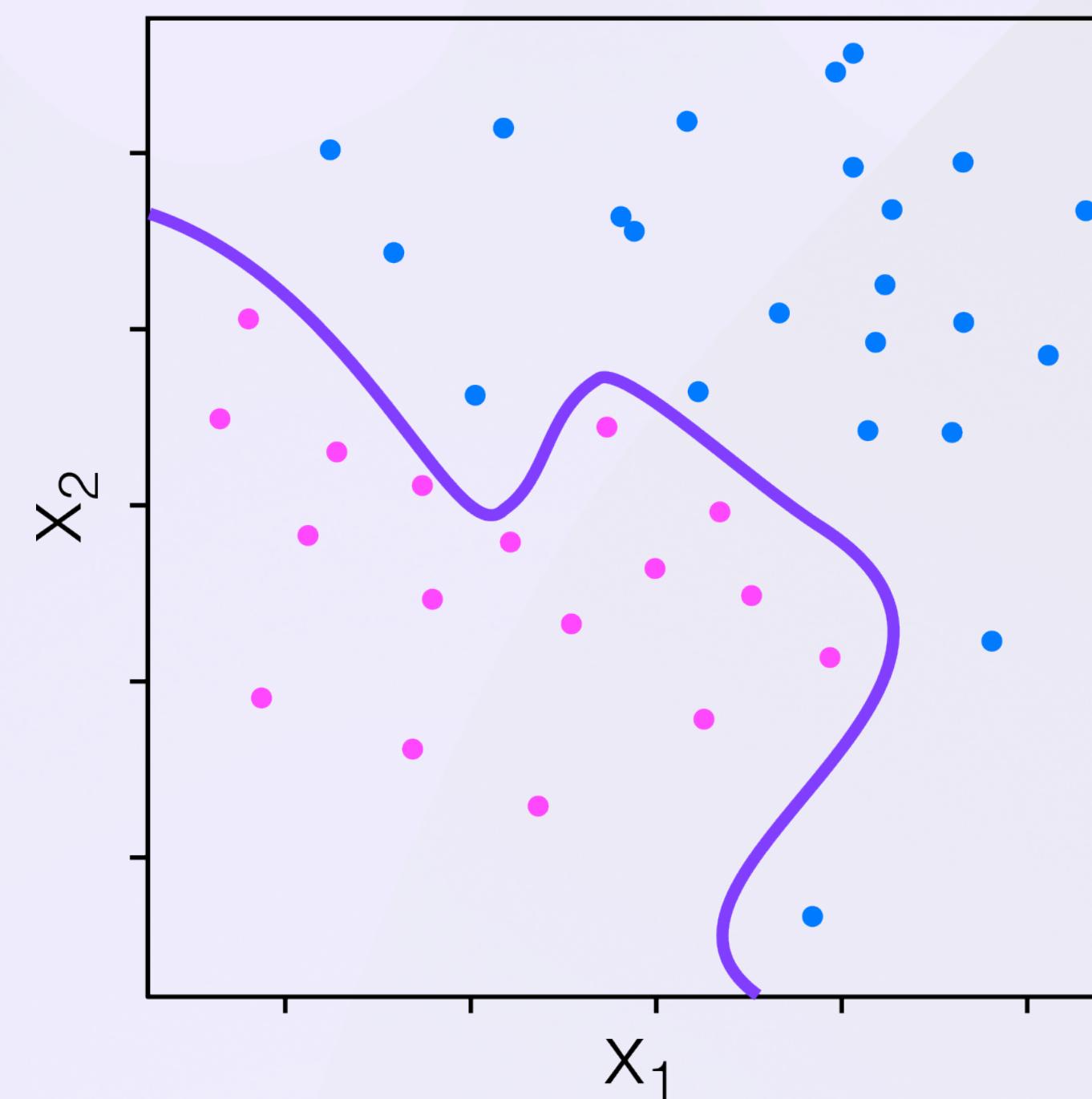
SUPPORT VECTOR MACHINE

Por eso existe el modelo llamado **Support Vector Machine (SVM)** o maquina de vector de soportes que extienden al Clasificador de Vector de Soportes permitiendo extender el espacio de atributos, usando funciones kernels.

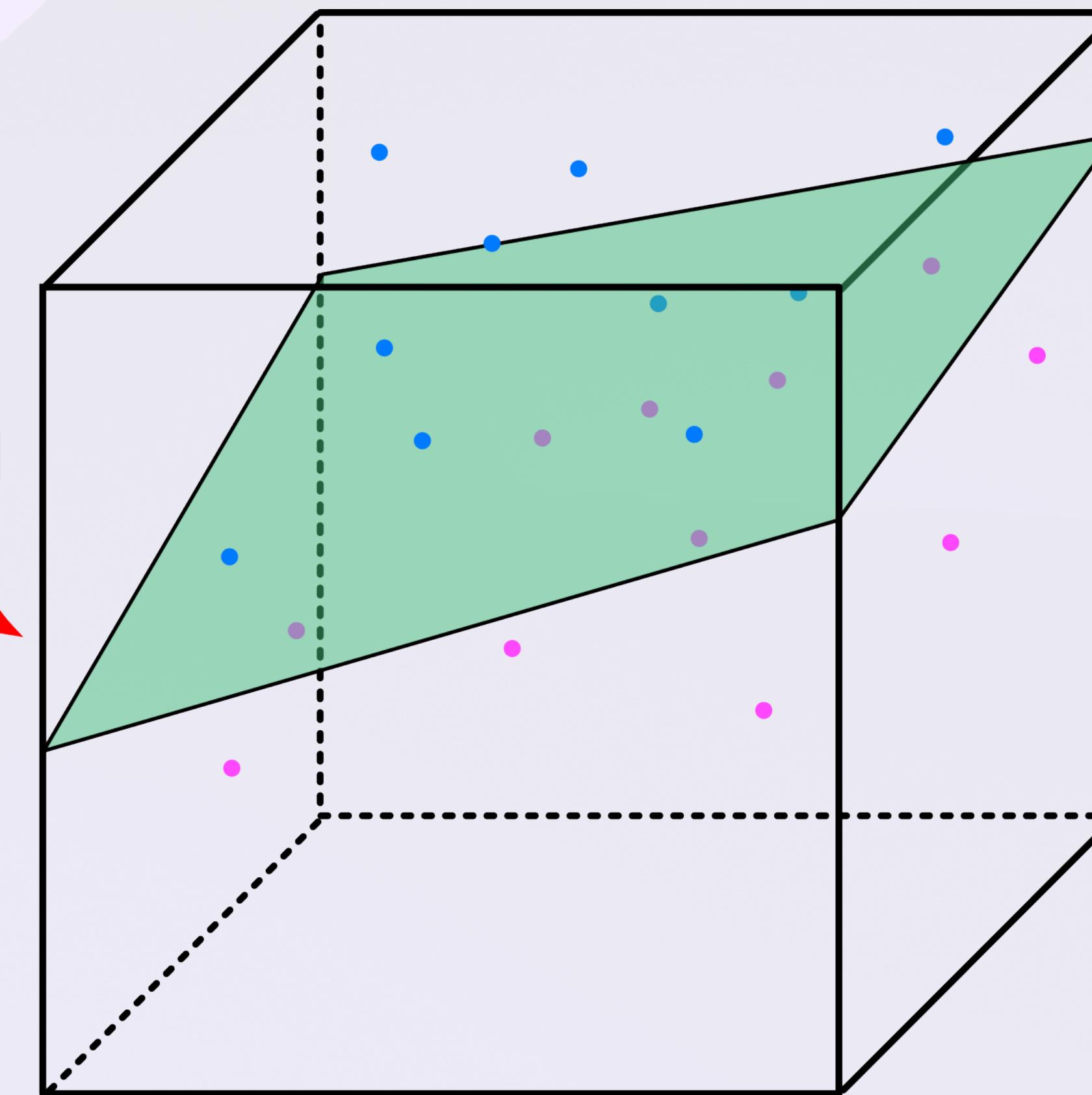
Este tipo de extensión es eficiente computacionalmente.

Veamos como lo hace...

SUPPORT VECTOR MACHINE



Función Kernel



SUPPORT VECTOR MACHINE

La función que usábamos para clasificar usando el **clasificador de vector de soportes** era:

$$f(X) = \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p$$

Se puede trabajar para llegar a la siguiente expresión:

$$f(X) = \beta_0 + \sum_{i=1}^n \alpha_i \langle X, X_i \rangle$$

- $\langle X, X_i \rangle$ es el producto interno entre la observación de entrada y la i-esima observación de entrenamiento.
- Hay n parámetros α_i , uno por observación de entrenamiento.
- Para estimar los parámetros $\alpha_1, \dots, \alpha_n$ y β_0 , se necesita $\binom{n}{2}$ productos internos $\langle X, X_i \rangle$ entre todos los pares de entrenamiento.

SUPPORT VECTOR MACHINE

$$f(X) = \beta_0 + \sum_{i=1}^n \alpha_i \langle X, X_i \rangle$$

Esta función para predecir, necesita de todos las observaciones del set de entrenamiento.

Sin embargo, resulta que α_i es **cero para aquellas observaciones que no son vectores de soporte**. De tal forma que realmente la función se puede reescribir:

$$f(X) = \beta_0 + \sum_{i \in S} \alpha_i \langle X, X_i \rangle$$

Que normalmente involucra muchos menos términos.

Es decir, para usar el modelo de clasificador de vector de soportes y computar sus coeficientes, todo lo que se necesita es el **producto interno**.

SUPPORT VECTOR MACHINE

Ahora, podemos reemplazar al producto interno por una generalización del producto interno:

$$K(X_i, X_{i'})$$

donde K es una función que llamamos **función kernel**. Un kernel o núcleo es una función que cuantifica la similitud entre dos vectores. Por ejemplo si tenemos:

$$K(X_i, X_{i'}) = \sum_{j=1}^p x_{ij}x_{i'j}$$

Volvemos al clasificador de vector de soportes con frontera lineal. Este kernel lineal en esencia esta midiendo la similitud mediante de las observaciones mediante una correlación de Pearson.

SUPPORT VECTOR MACHINE

Pero también podemos obtener un kernel polinomial de grado d ($d > 1$):

$$K(X_i, X_{i'}) = \left(1 + \sum_{j=1}^p x_{ij}x_{i'j}\right)^d$$

Lo que nos generará una frontera de decisión polinomial.

Otro kernel popular es kernel radial definido como:

$$K(X_i, X_{i'}) = \exp\left(-\gamma \sum_{j=1}^p (x_{ij} - x_{i'j})^2\right)$$

Donde γ es una constante positiva. Este kernel nos permite obtener fronteras radiales. Este kernel se basa en la **distancia euclíadiana**, con un efecto muy local.

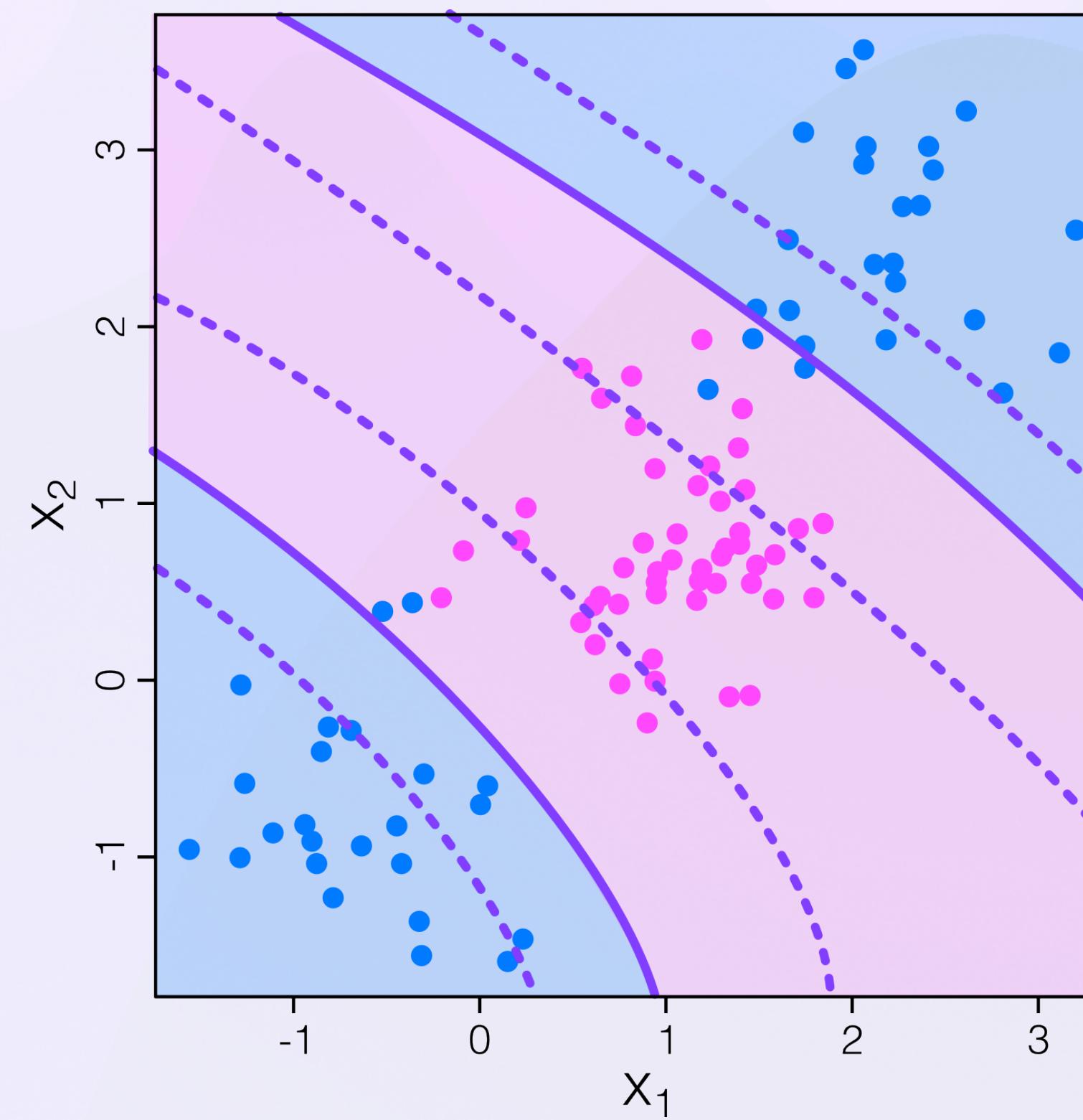
SUPPORT VECTOR MACHINE

Por lo que podemos a la función de decisión de la **máquina de vector de soportes** como:

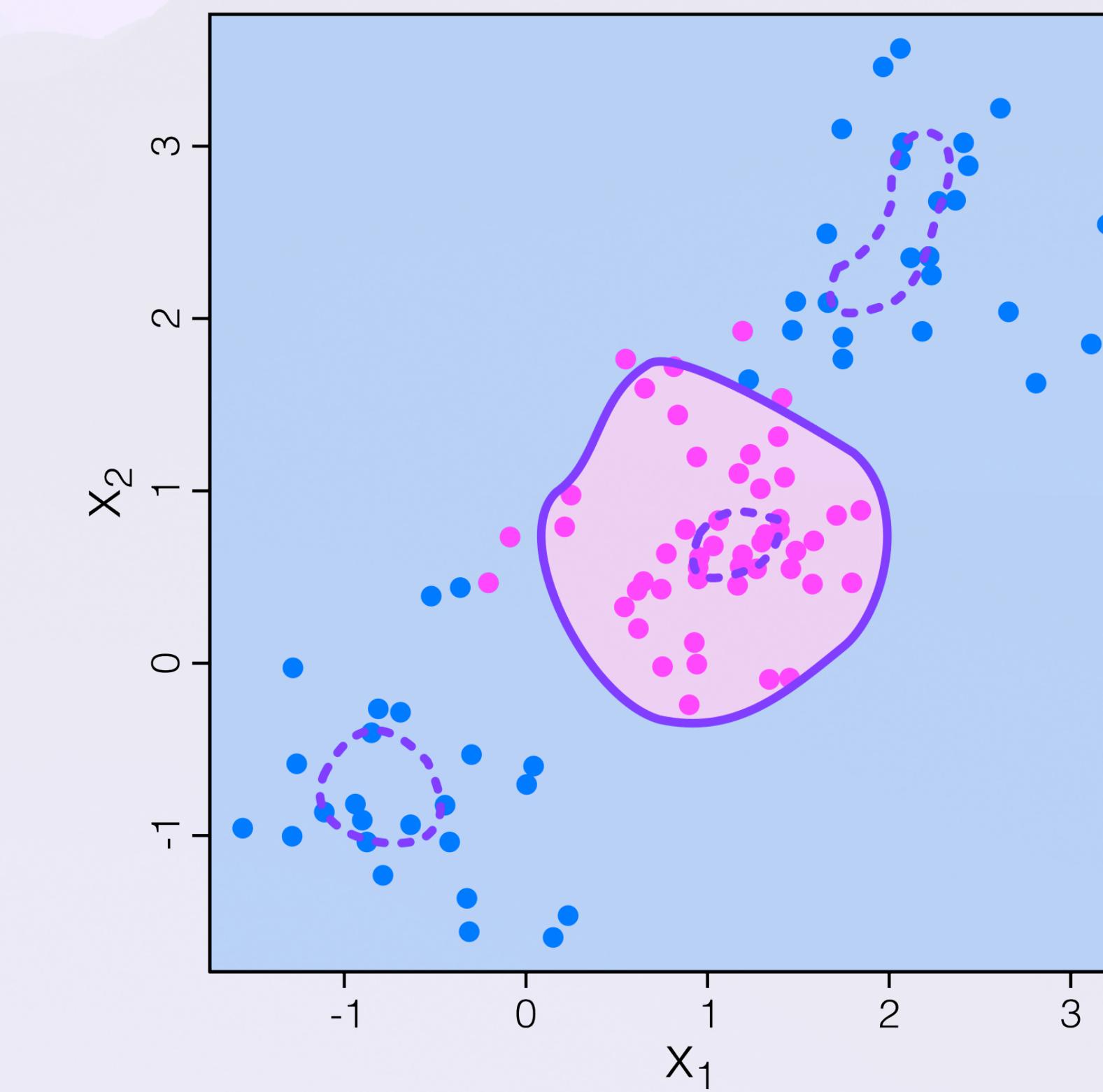
$$f(X) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i K(X, X_i)$$

La forma de entrenamiento, la importancia de los vectores de soporte y el hiperparámetro C se mantienen. La gran diferencia es que ahora las fronteras de decisión no son necesariamente lineales y determinada por la función **kernel elegida**.

SUPPORT VECTOR MACHINE



Kernel polinomial de orden 3



Kernel radial

SUPPORT VECTOR MACHINE

Todo lo que estuvimos viendo, este clasificador es binario. Ese es su punto fuerte y está preparado para clasificar entre **dos clases**.

Cómo hacemos para clasificar con mas de dos clases:

- **One-vs-one**: En este caso es comparar a cada clase con otra, construyendo un SVM para cada par posible de clases. Una vez que tenemos entrenados los SVMs, la clasificación final de un predicción en una clase particular es dada por la clase que más votada por los clasificadores.
- **One-vs-all**: En este caso comparamos una clase contra el resto de las clases. Por lo que tendremos K SVMs, cada uno entrenado para comparar una clase en particular contra las restantes como un solo conjunto. Se asigna la clase a una predicción para el modelo que predice la clase con mayor confidencia usando los $f(X)$ de los SVMs.



SUPPORT VECTOR MACHINE EN REGRESIÓN

SUPPORT VECTOR MACHINE EN REGRESIÓN

Hasta ahora vimos a los SVM como un modelo de clasificación, y son más popular como modelo de clasificación pero podemos usar esta idea para realizar regresiones.

Con solo cambiar las restricción, entendiendo que ahora tenemos que predecir un target continuo.

Si cambiamos la restricción del **Maximal Margin Classifier**:

$$y_i (\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M \quad \forall i = 1, 2, \dots, n$$

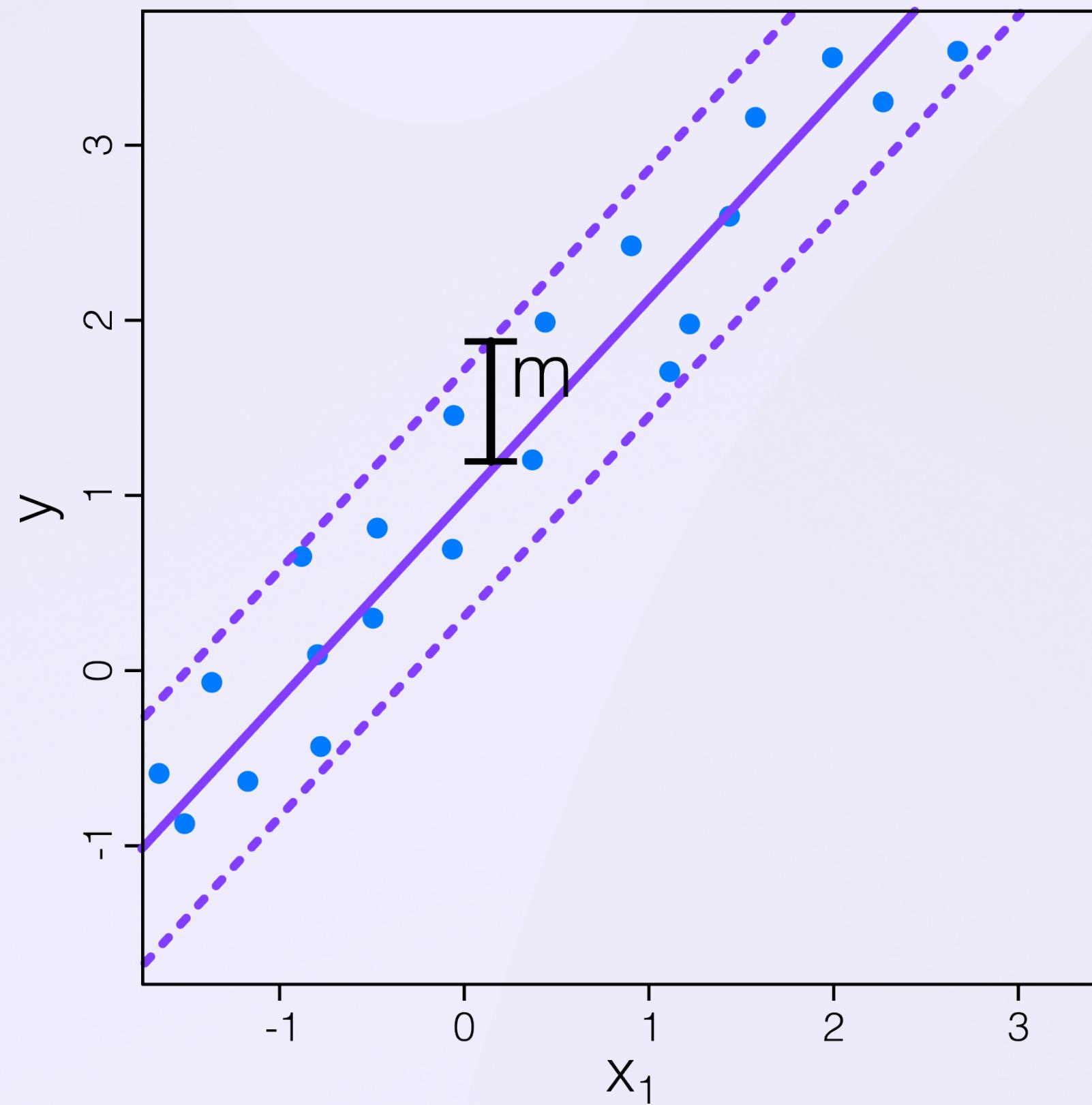
por

$$-m \leq y_i - (b_0 + b_1 x_{i1} + \dots + b_p x_{ip}) \leq m \quad \forall i = 1, 2, \dots, n$$

podemos realizar regresiones.

SUPPORT VECTOR MACHINE EN REGRESIÓN

Lo que se optimiza ahora es el hiperplano que mejor que logra meter a todos los puntos de entrenamiento dentro del margen, minimizando el valor del margen.

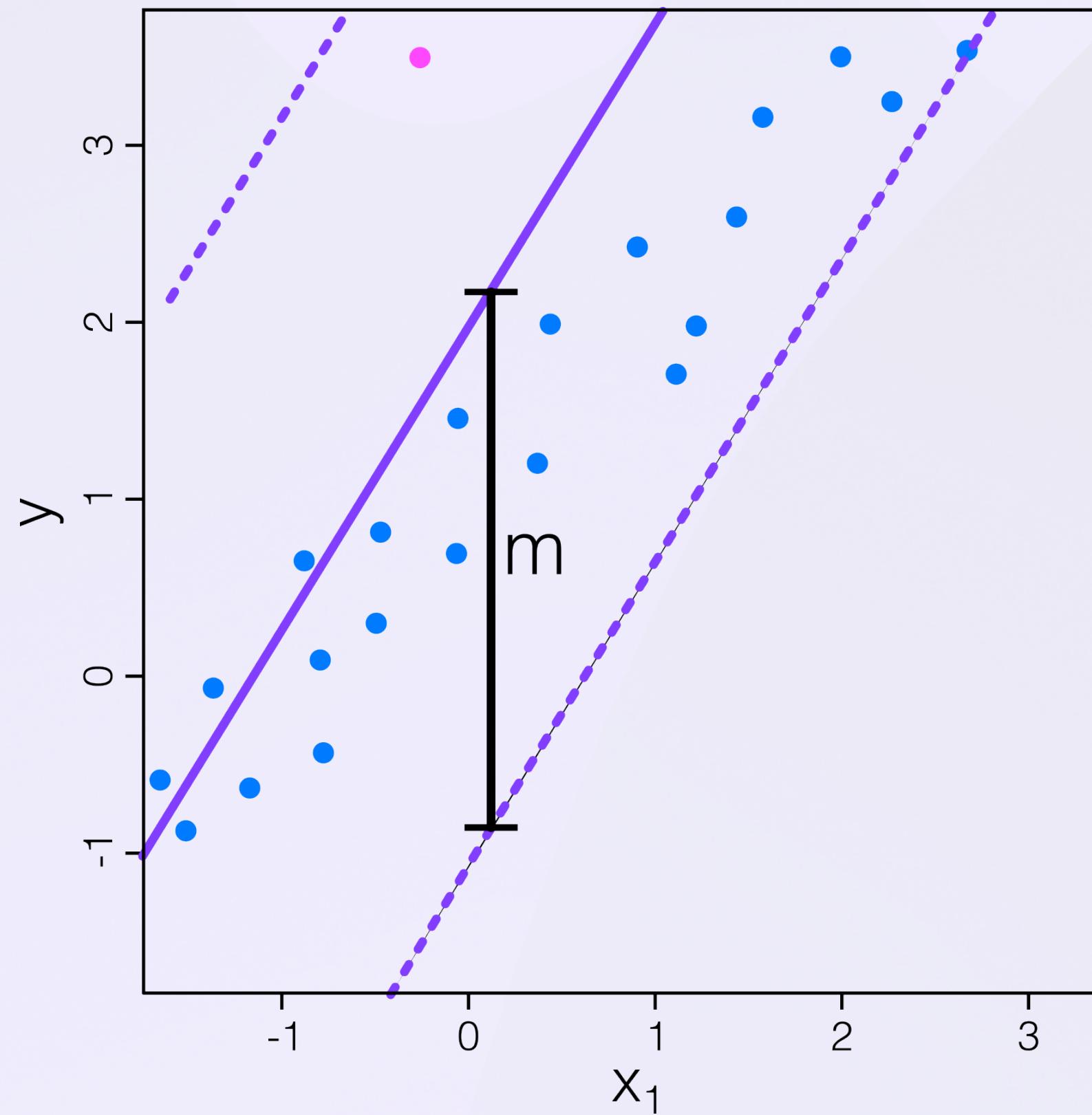


Entonces una vez que se tiene el modelo entrenado, se puede usar para regresión con la formula:

$$f(X) = b_0 + b_1x_1 + \dots + b_px_p$$

SUPPORT VECTOR MACHINE EN REGRESIÓN

Lo que se optimiza ahora es el hiperplano que mejor que logra meter a todos los puntos de entrenamiento dentro del margen, minimizando el valor del margen.



Entonces una vez que se tiene el modelo entrenado, se puede usar para regresión con la formula:

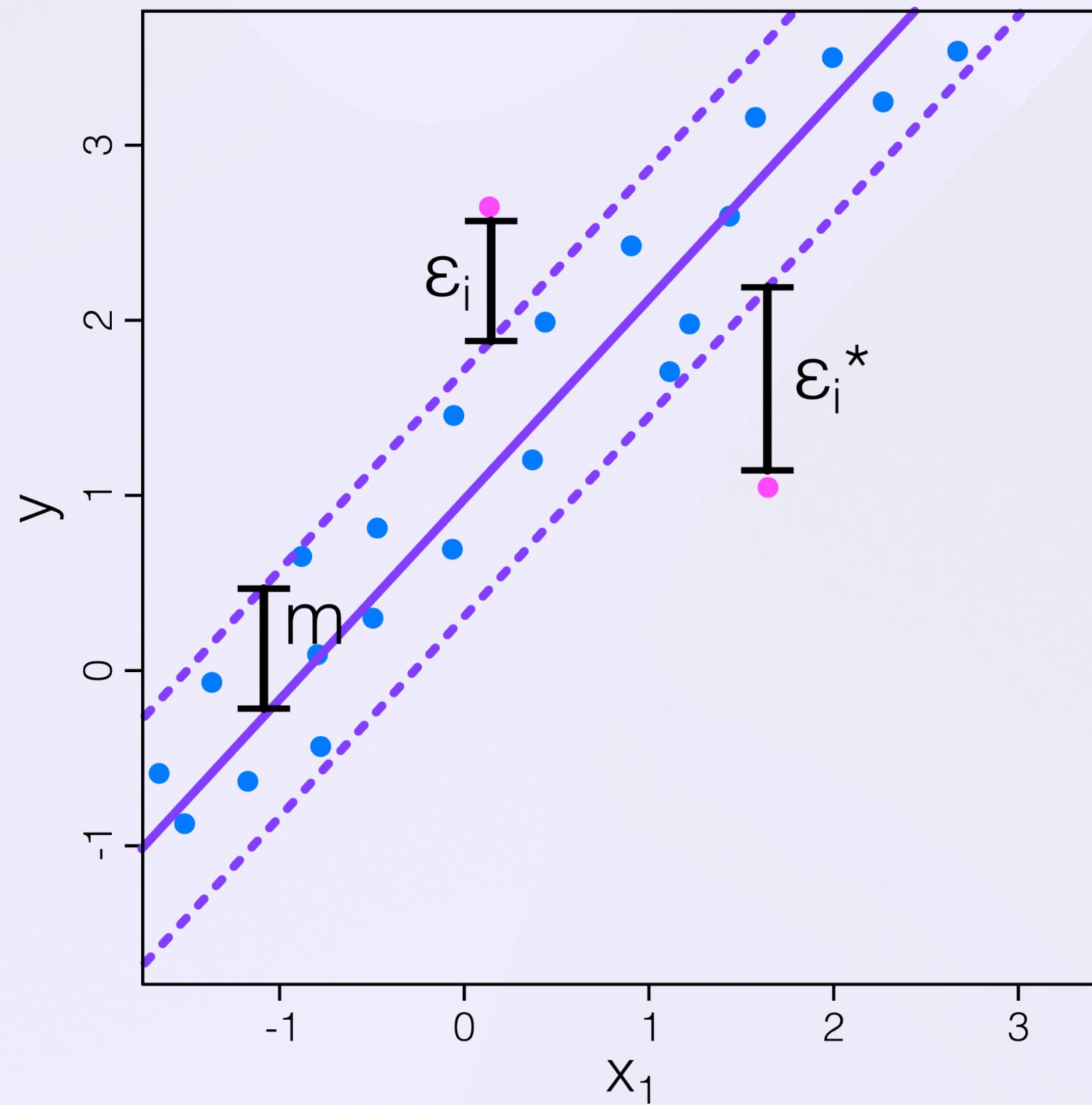
$$f(X) = b_0 + b_1x_1 + \dots + b_px_p$$

Igual que en el caso de clasificación, una observación de entrenamiento muy alejada del resto puede cambiar radicalmente a la regresión.

Por lo que se debe incorporar la misma idea de permitir que algunas observaciones puedan estar por fuera.

SUPPORT VECTOR MACHINE EN REGRESIÓN

Entonces para evitar esto, agregamos un hiperparámetro que nos permite tener algunas observaciones fuera de la zona de margen.



$$\sum_{i=1}^n (\epsilon_i + \epsilon_i^*) \leq C \quad \epsilon_i, \epsilon_i^* \geq 0, \quad \forall i = 1, \dots, n$$

Y la restricción es:

$$y_i - (\langle \hat{\beta}, X \rangle + b) \leq m + \epsilon_i \quad \forall i = 1, \dots, n$$
$$(\langle \hat{\beta}, X \rangle + b) - y_i \leq m + \epsilon_i^*$$

SUPPORT VECTOR MACHINE EN REGRESIÓN

Y podemos extender a usar funciones kernels para extender a casos no lineales, extendiendo a un espacio dimensional mayor.

The background features a minimalist design with abstract, wavy shapes in shades of purple and dark blue. These shapes are layered and overlap, creating a sense of depth and movement across the entire frame.

VAMOS A PRÁCTICAR UN POCO...