MÁQUINAS DE VECTORES DE SOPORTE

Aprendizaje Automático

CEIoT - FIUBA

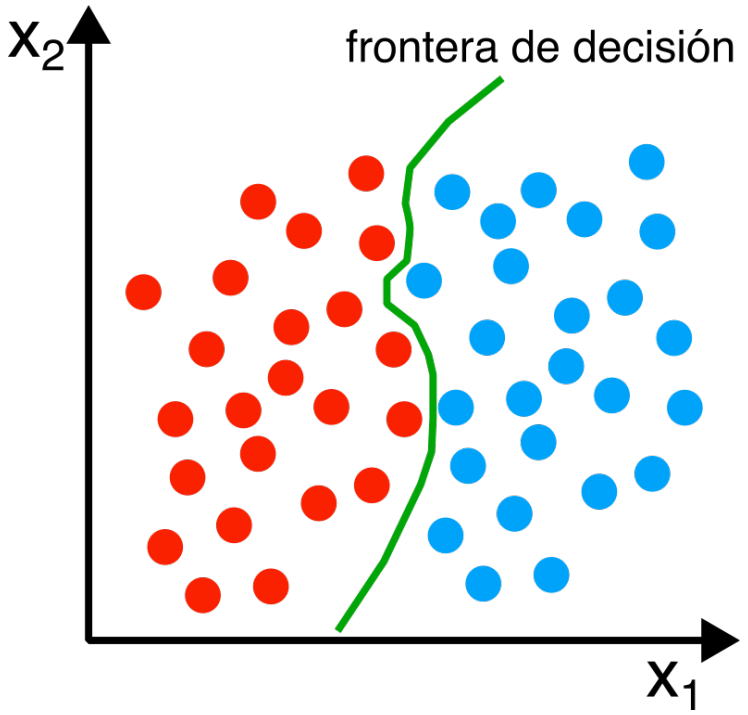
Dr. Ing. Facundo Adrián Lucianna

LO QUE VIMOS LA CLASE ANTERIOR…

CLASIFICACIÓN

Es más común encontrarnos con problema de clasificación que de regresión:

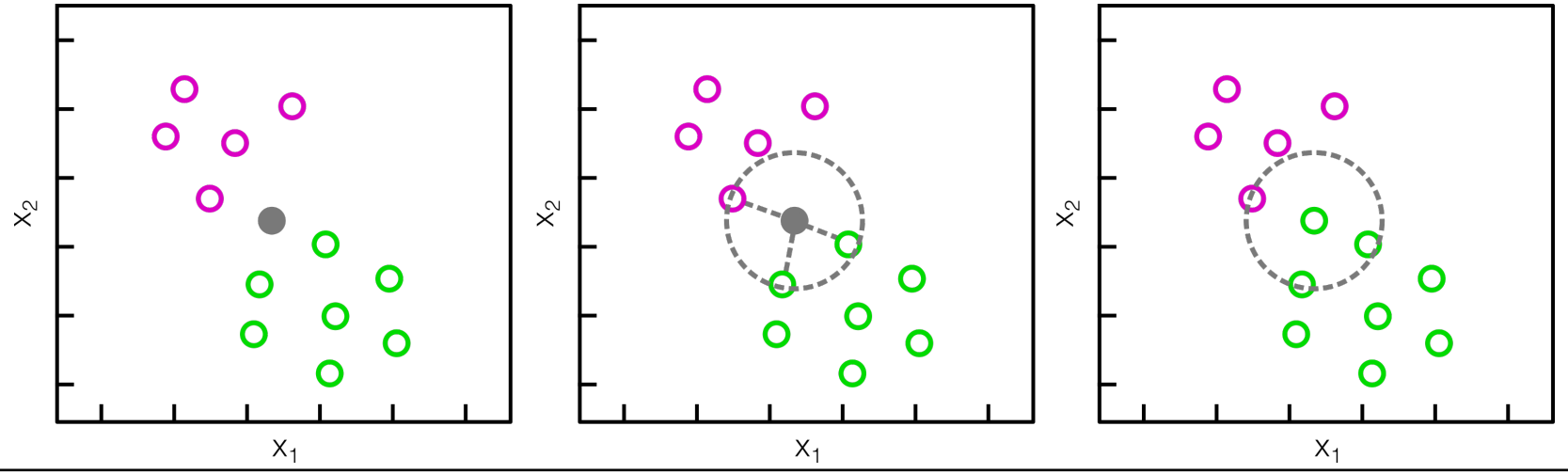
- Una persona llega a una guardia con un set de síntomas atribuidos a una de tres condiciones médicas.
- Un servicio de banca online debe determinar si una transacción en el sitio es fraudulenta o no, usando como base la dirección IP, historia de transacciones, etc.
- En base a la secuencia de ADN de un número de pacientes con y sin una enfermedad dada, un genetista debe determinar que mutaciones de ADN genera un efecto nocivo relacionado a la enfermedad o no.



CLASIFICADOR KNN

El clasificador de k vecinos más cercanos (KNN o k-NN), es un algoritmo que utiliza la proximidad de sus vecinos para hacer clasificaciones sobre la agrupación de un punto.

La idea se basa de la **suposición de que se pueden encontrar puntos similares cerca uno del otro en base a votación de pluralidad** (se elige la clase en función de la moda de la clase de sus vecinos). Este modelo no obtiene una salida de probabilidad, solo nos dice de que clase es.



REGRESIÓN LOGÍSTICA

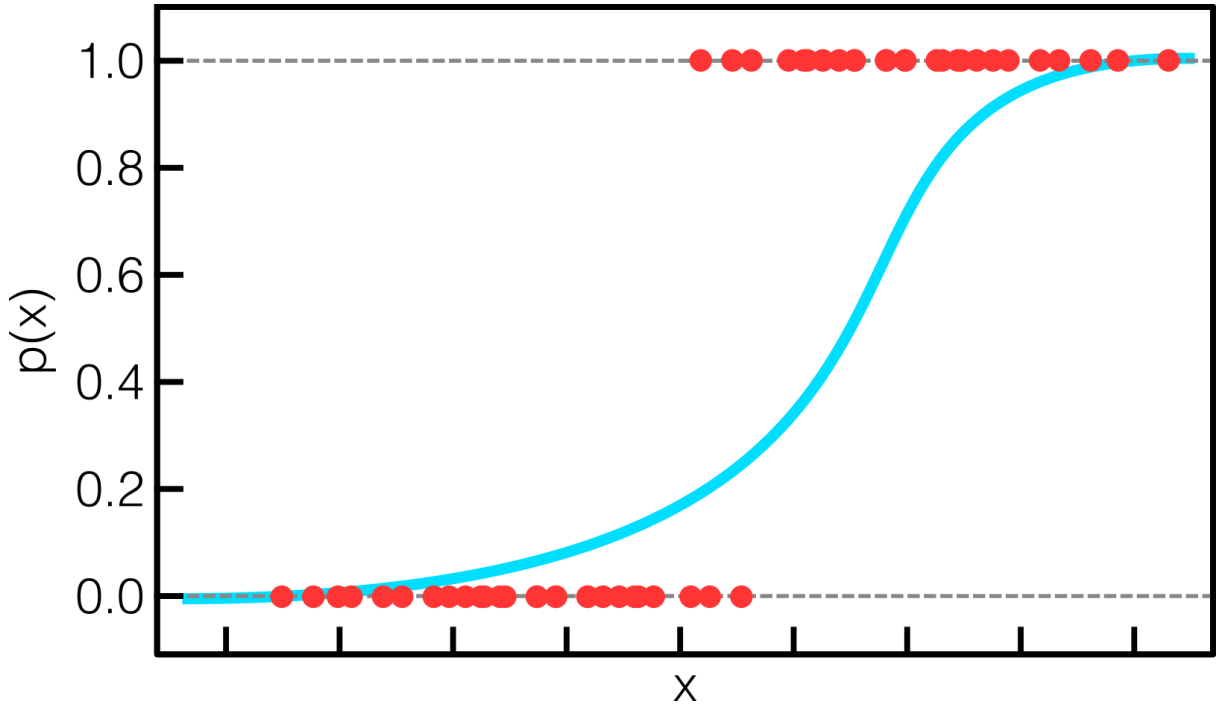
Para clasificar, en vez de modelar la salida, modelamos la probabilidad de que una observación es de una clase u otra. Para ello, podemos modelar a la probabilidad usando una función que nos asegure que siempre tendremos valores entre 0 y 1.

En regresión logística, esto lo resolvemos usando una función sigmoide:

$$p(x) = \frac{e^{b+w_0 x}}{1 + e^{b+w_0 x}} = \frac{1}{1 + e^{-(b+w_0 x)}}$$

REGRESIÓN LOGÍSTICA

Lo que visualmente se observa:



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

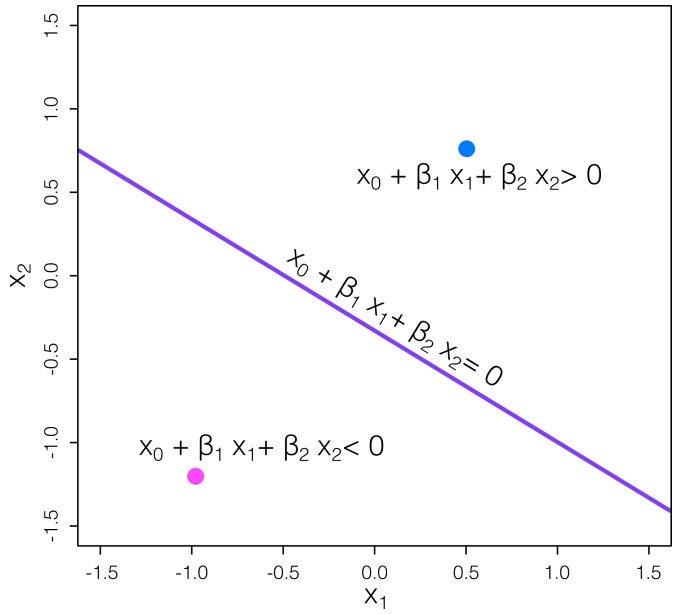
La definición matemática ya la conocemos:

- 2 dimensiones: $\beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0$ $\quad\quad$ $\beta_0, \beta_1, \beta_2 \in \mathbb{R}$
- p-dimensiones: $\beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p = 0$ $\quad\quad$ $\beta_0, \beta_1, \ldots, \beta_p \in \mathbb{R}$

MAXIMAL MARGIN CLASSIFIER

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

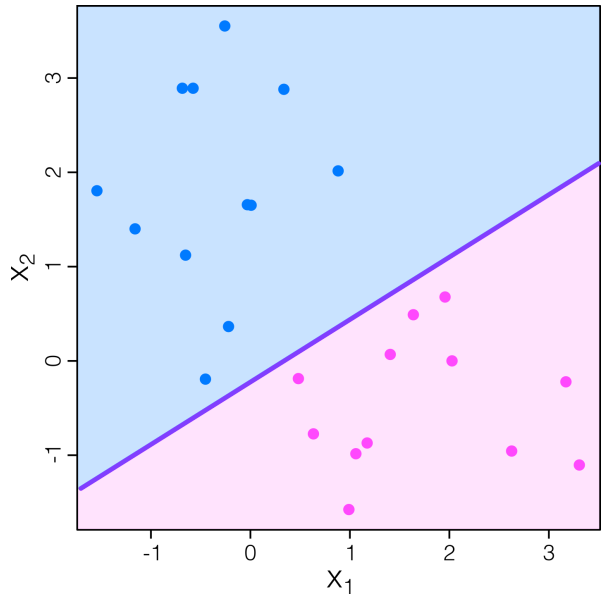
Pero más interesante, si el punto hace que,



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 con su clase, podríamos clasificar como:



$$\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p\, x_{ip} > 0 \qquad Si\ y_i = 1$$

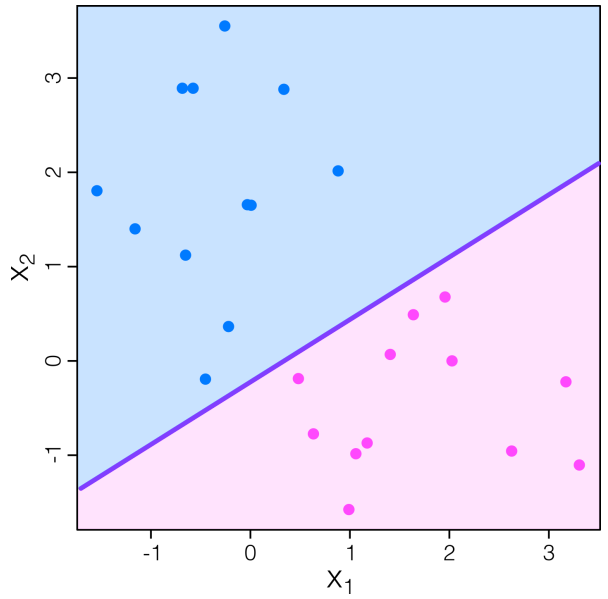
$$\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p\, x_{ip} < 0 \qquad Si\ y_i = -1$$

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

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 con su clase, podríamos clasificar como:



Para ordenarnos, vamos a definir una función f(X) que nos servirá para clasificar:

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

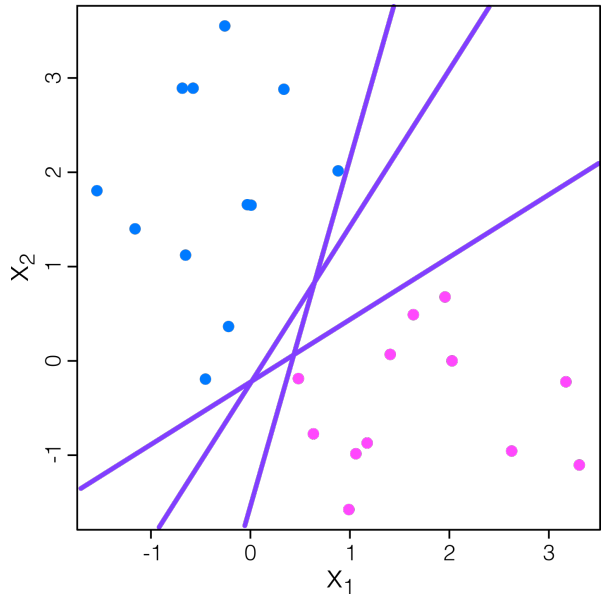
$$f(X_i) = \beta_0 + \beta_1 x_{i1} + \ldots + \beta_p\, x_{ip} < 0 \quad entonces\ \tilde{y}_i = -1$$

Que, además, nos permite ver la confidencia del clasificador. Es decir, cuanto más **grande** es el valor absoluto de f(X), más **segura o certera** 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 en busca del hiperplano que más lejos se encuentra de los datos de entrenamiento.



Es decir:

1. Tomamos un hiperplano.
2. Calculamos las distancias de cada punto de entrenamiento
3. Obtenemos la distancia más chica de todas las distancias

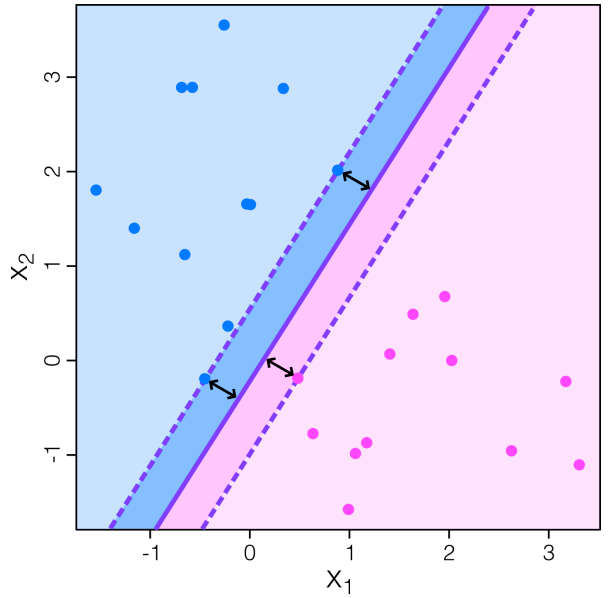
Esa distancia la llamamos **margen**.

Cada hiperplano tendrá su 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 en busca del hiperplano que más lejos se encuentra de los datos de entrenamiento.



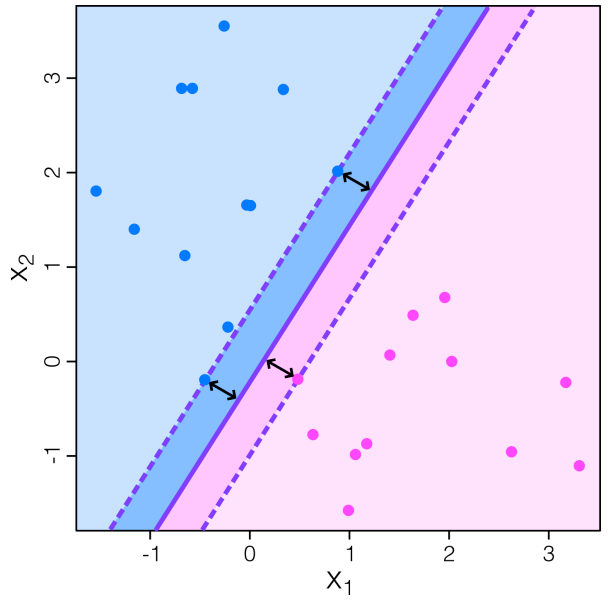
El objetivo es buscar el hiperplano que mas grande posee este margen y, el algoritmo que hace esto es el **Maximal Margin Classifier.**

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.

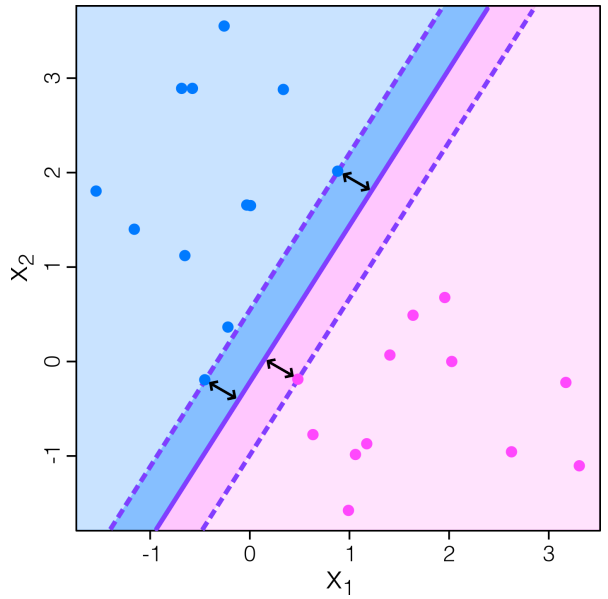


Esto es porque si uno de estos puntos se mueve, aunque sea un poco, el hiperplano también se va a mover.

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

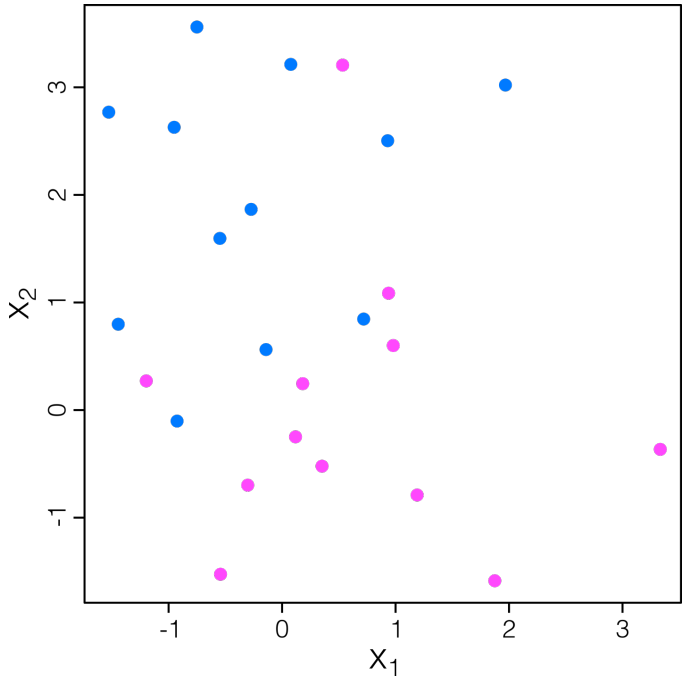
MAXIMAL MARGIN CLASSIFIER

El entrenamiento de este modelo es encontrar los $\beta_0, \ldots, \beta_p$ que maximicen el valor del margen, el cual se resuelve como un problema de optimización con criterio cuadrático y desigualdades de restricción lineales (mediante multiplicadores de Lagrange).



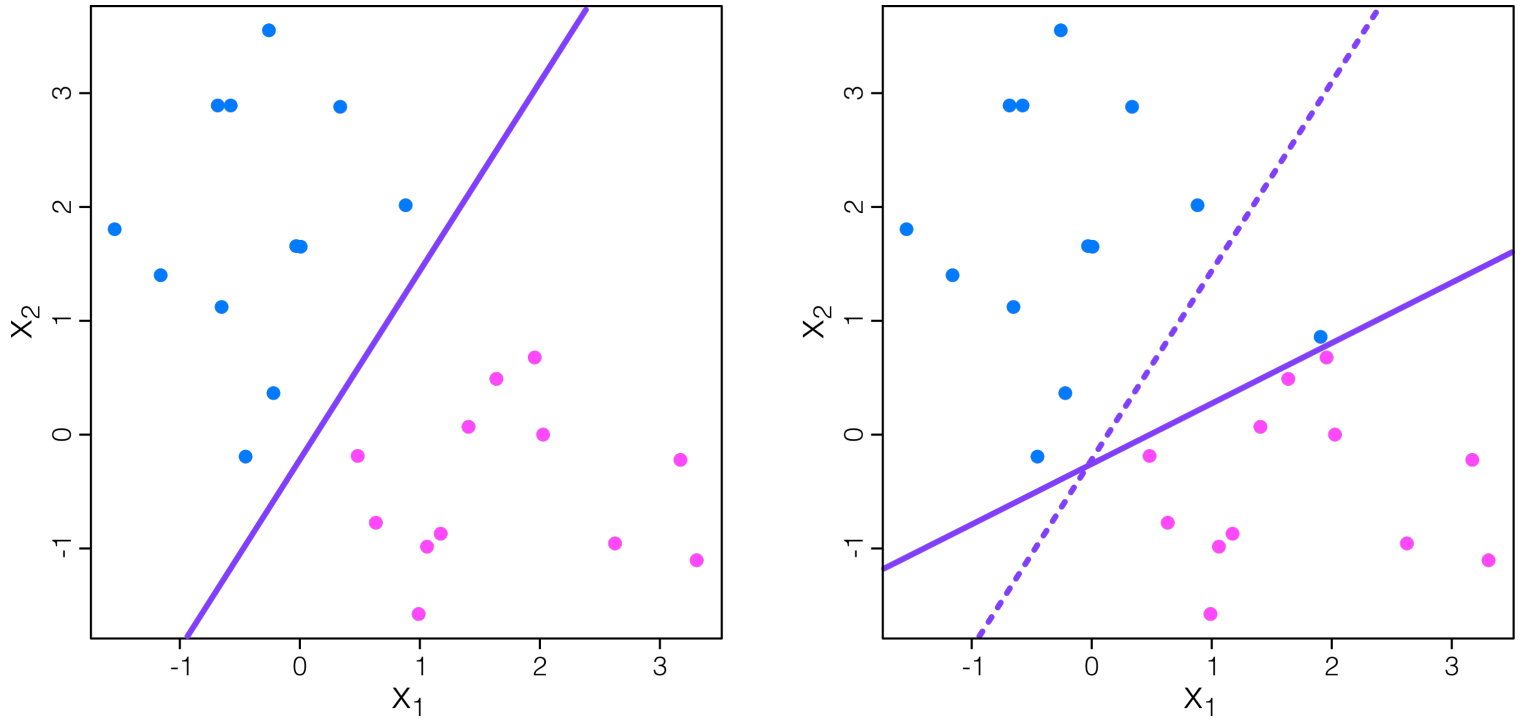
CASO NO SEPARABLE

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 sobreajuste (error de varianza)… *Necesitamos mejorar el modelo.*



CLASIFICADOR DE VECTOR DE SOPORTES

CLASIFICADOR DE VECTOR DE SOPORTES

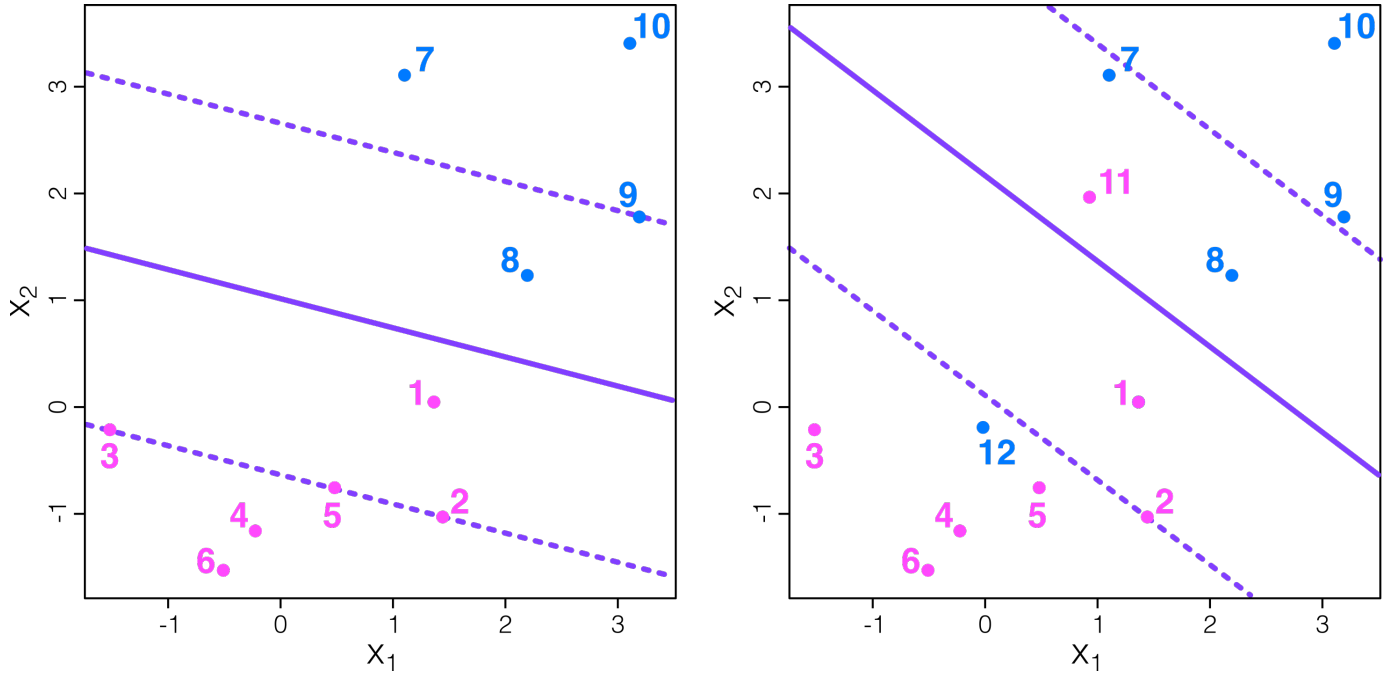
Por lo que vimos, si queremos seguir usando un hiperplano, debemos relajar las exigencias:

- 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 **margen**, sino también en el **lado equivocado del hiperplano**.



CLASIFICADOR DE VECTOR DE SOPORTES

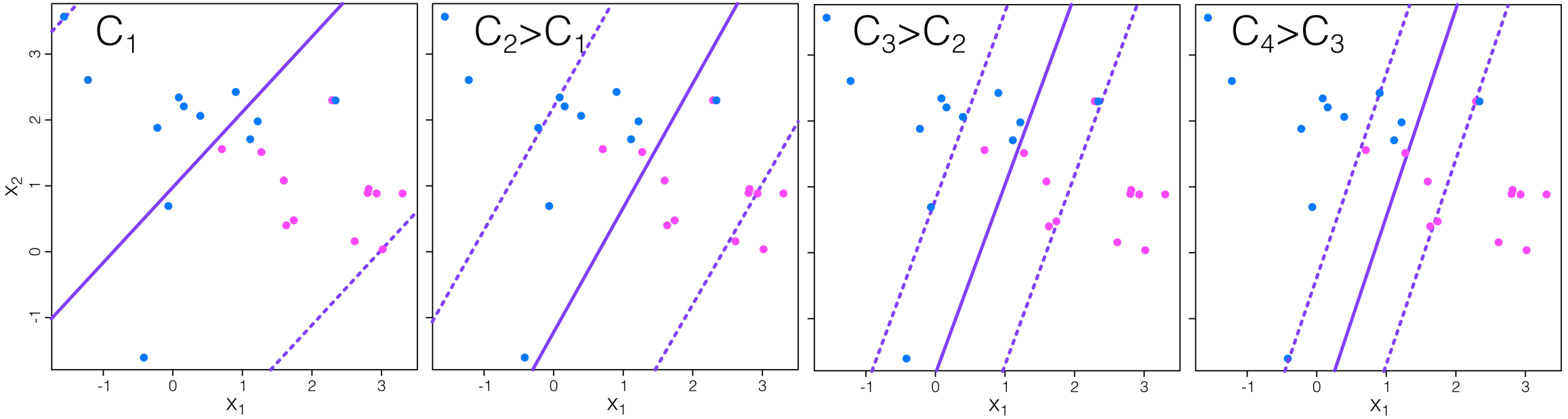
El modelo clasificador de vectores de soporte es el modelo apropiado para este caso. Con este modelo una observación puede estar no sólo en el lado equivocado del **margen**, sino también en el **lado equivocado del hiperplano**.

Esto lo resolvemos usando regularización L1 usando un hiper parámetro C.

Este parámetro determina el número y la gravedad de las violaciones del margen (y del hiperplano) que toleraremos.

Si $1/C>0$, no más que $1/C$ observaciones pueden estar en el lado equivocado del hiperplano. Si C es más chico, más laxos son las exigencias y los márgenes serán más grandes.

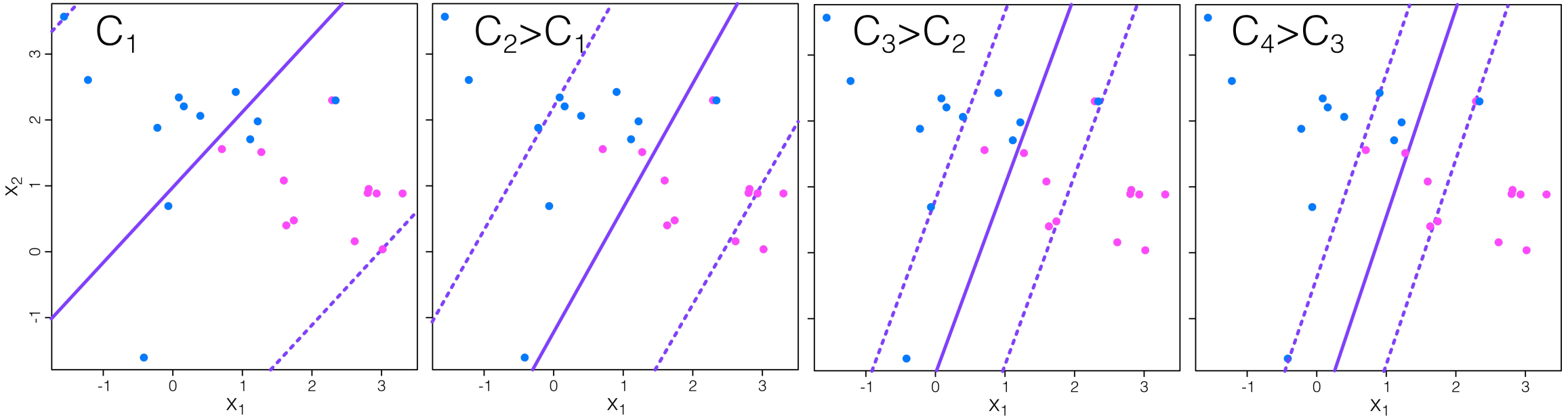
CLASIFICADOR DE VECTOR DE SOPORTES



Hay un pequeño número 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



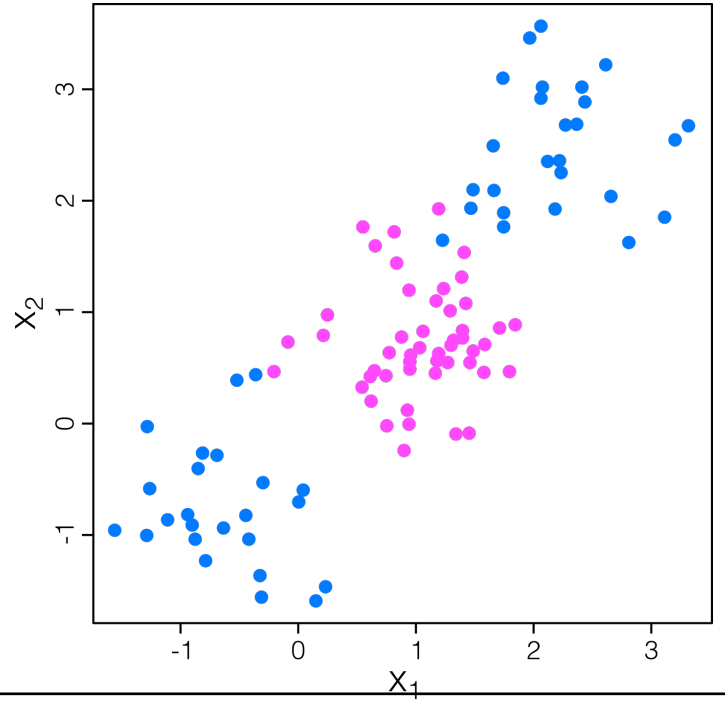
El valor de C nos da un trade-off (compensación) entre **sesgo** y **varianza**.

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

SUPPORT VECTOR MACHINE

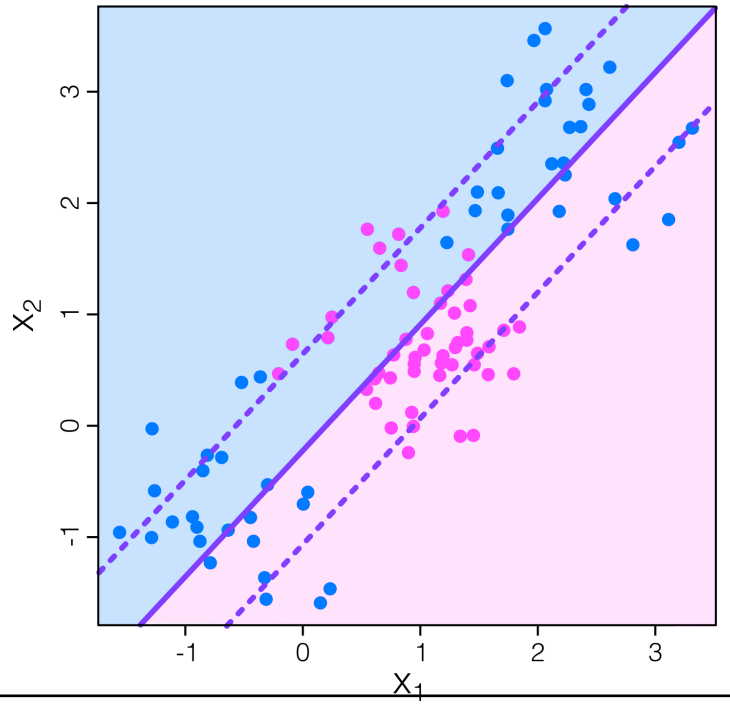
SUPPORT VECTOR MACHINE

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 $x_1,x_2,\ldots,x_p$

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

El modelo será lineal en el espacio extendido, pero cuando volvamos al espacio dado por $x_1,x_2,\ldots,x_p$, 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 de ajuste se vuelve **inmanejable**.

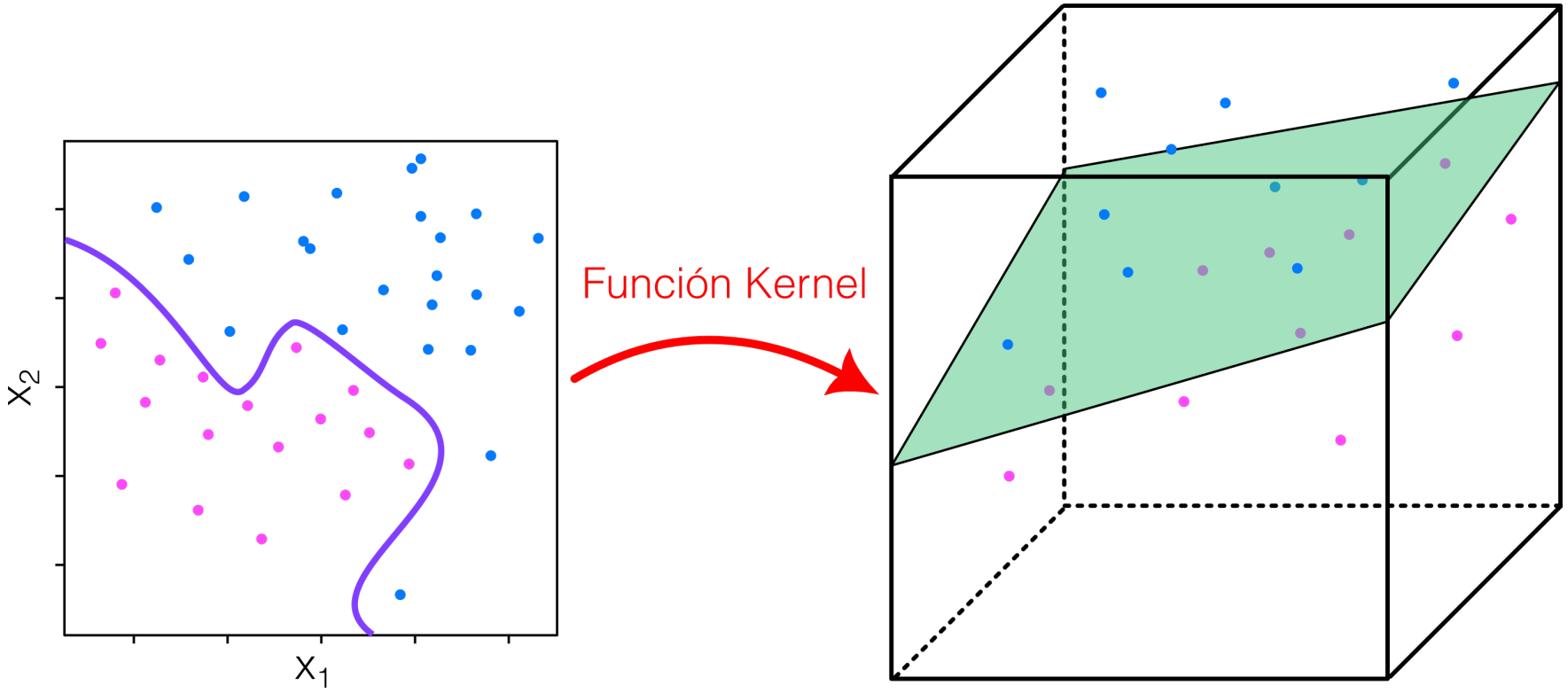
SUPPORT VECTOR MACHINE

Por eso existe el modelo llamado **Support Vector Machine (SVM)** o máquina 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



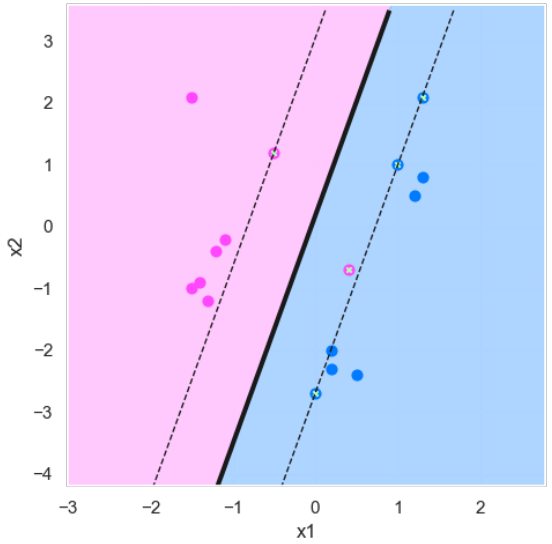
SUPPORT VECTOR MACHINE

Tenemos muchas **funciones kernels**, entre ellas podemos mencionar:

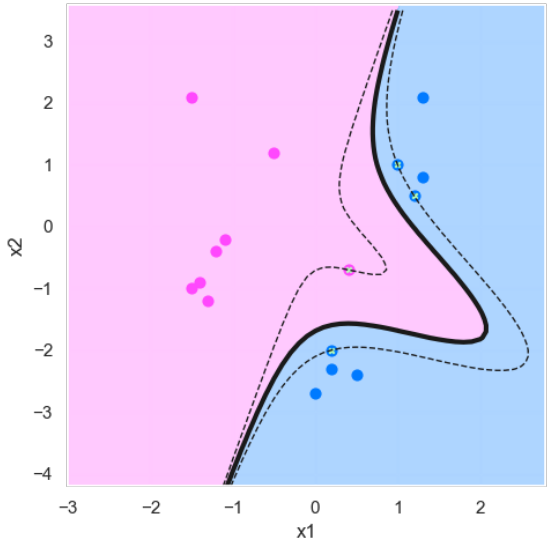
- **Lineales**: Es exactamente el modelo que vimos.
- **Polinomial**: Permite una frontera polinomial.
- **Radial**: Permite fronteras radiales. Este kernel se basa en la distancia euclidiana, con un efecto muy local.
- **Sigmoidea**: Tiene una frontera tipo S, los cuales son muy raros de observar en casos reales, por lo que no es muy usado.

SUPPORT VECTOR MACHINE

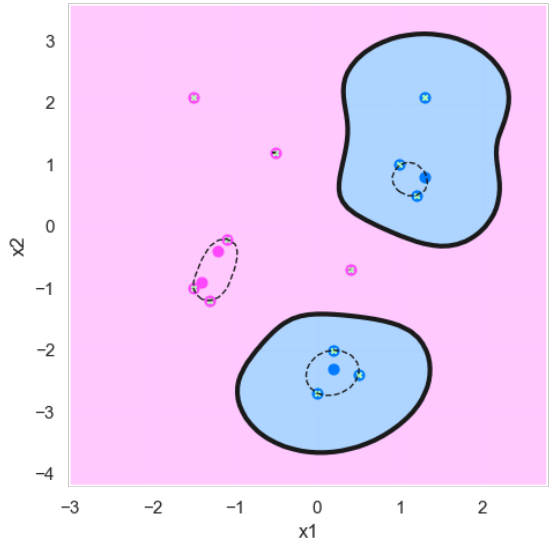
Tenemos muchas **funciones kernels**, entre ellas podemos mencionar:



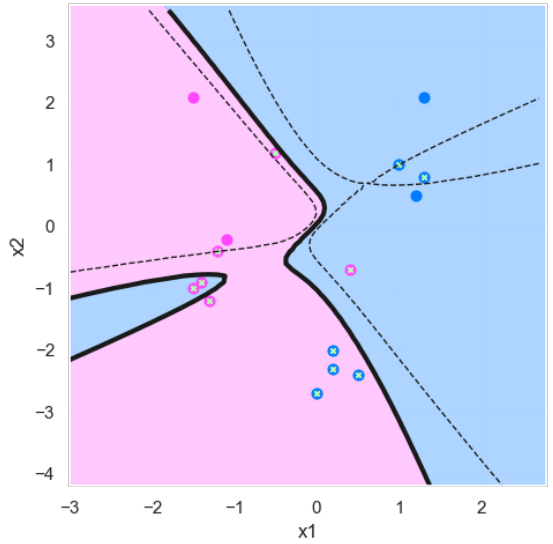
Lineales



Polinomial



Radial



Sigmoidea

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 más 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 una 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 la restricción, entendiendo que ahora tenemos que predecir un target continuo.

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

$$y_i\left(\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p\, x_{ip}\right) \geq M \qquad \forall i = 1, 2, \ldots, n$$

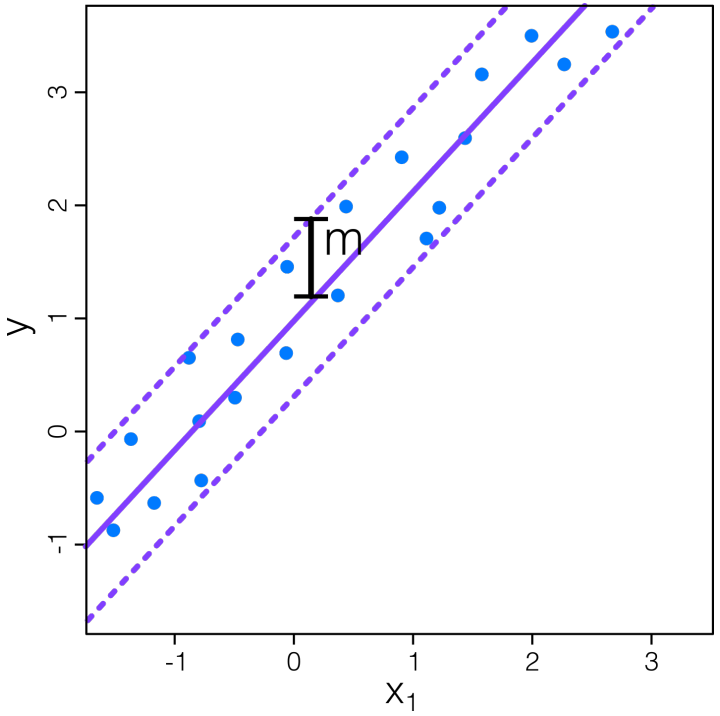
por

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

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.

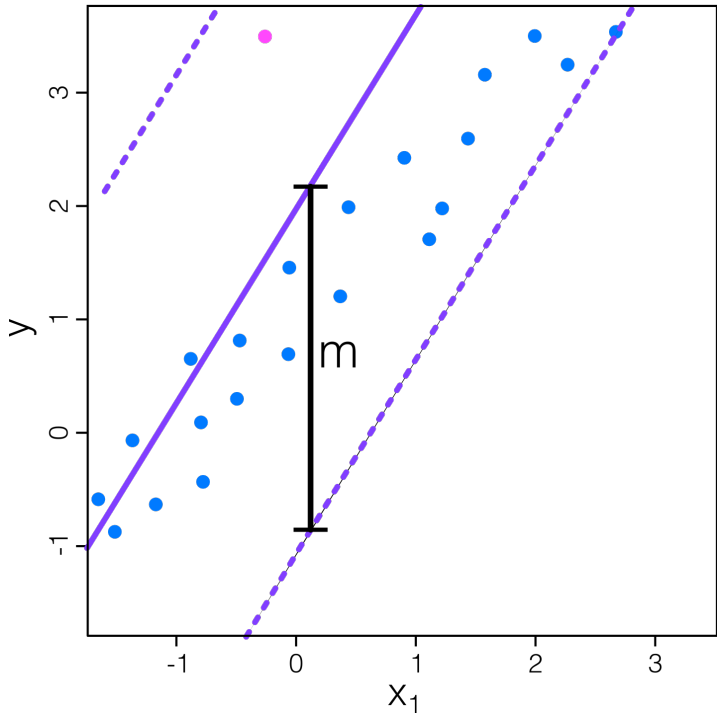


Entonces una vez que se tiene el modelo entrenado, se puede usar para regresión con la formula:

$$f(X) = b_0 + b_1 x_1 + \ldots + b_p\, x_p$$

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:

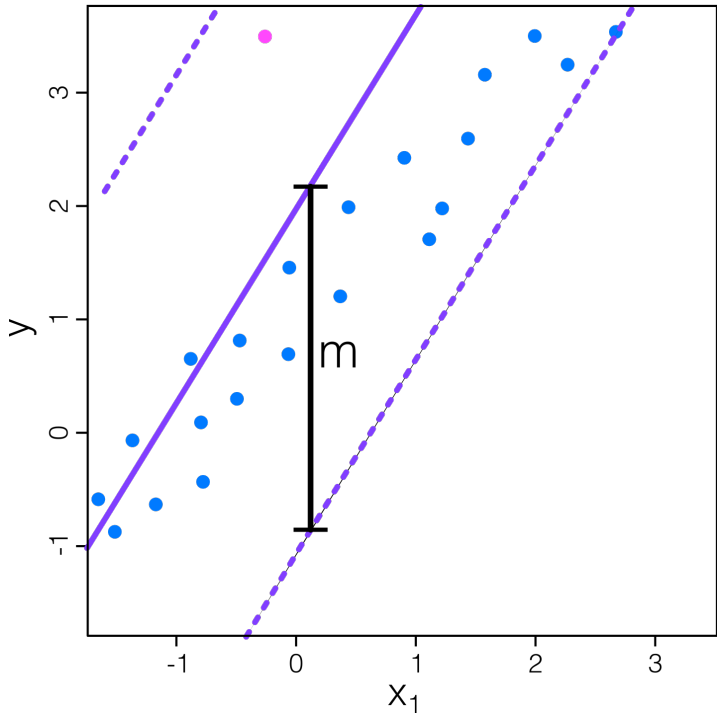
$$f(X) = b_0 + b_1 x_1 + \ldots + b_p\, x_p$$

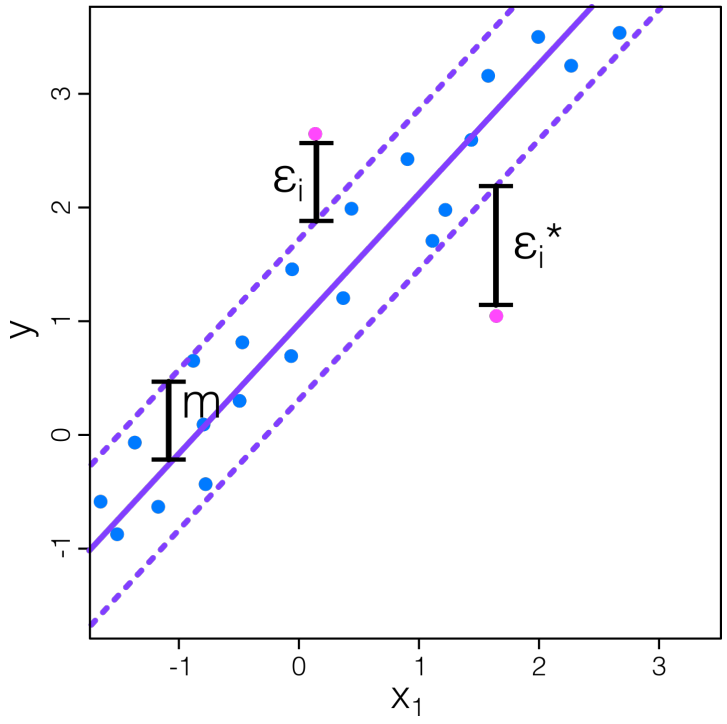
Igual que en el caso de clasificación, una observación de entrenamiento muy alejada del resto puede cambiar radicalmente a la regresión.

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





SUPPORT VECTOR MACHINE EN REGRESIÓN

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