



ÁRBOLES DE DECISIÓN

APRENDIZAJE DE MAQUINA I - CEIA - FIUBA

Antonio Zarauz Moreno

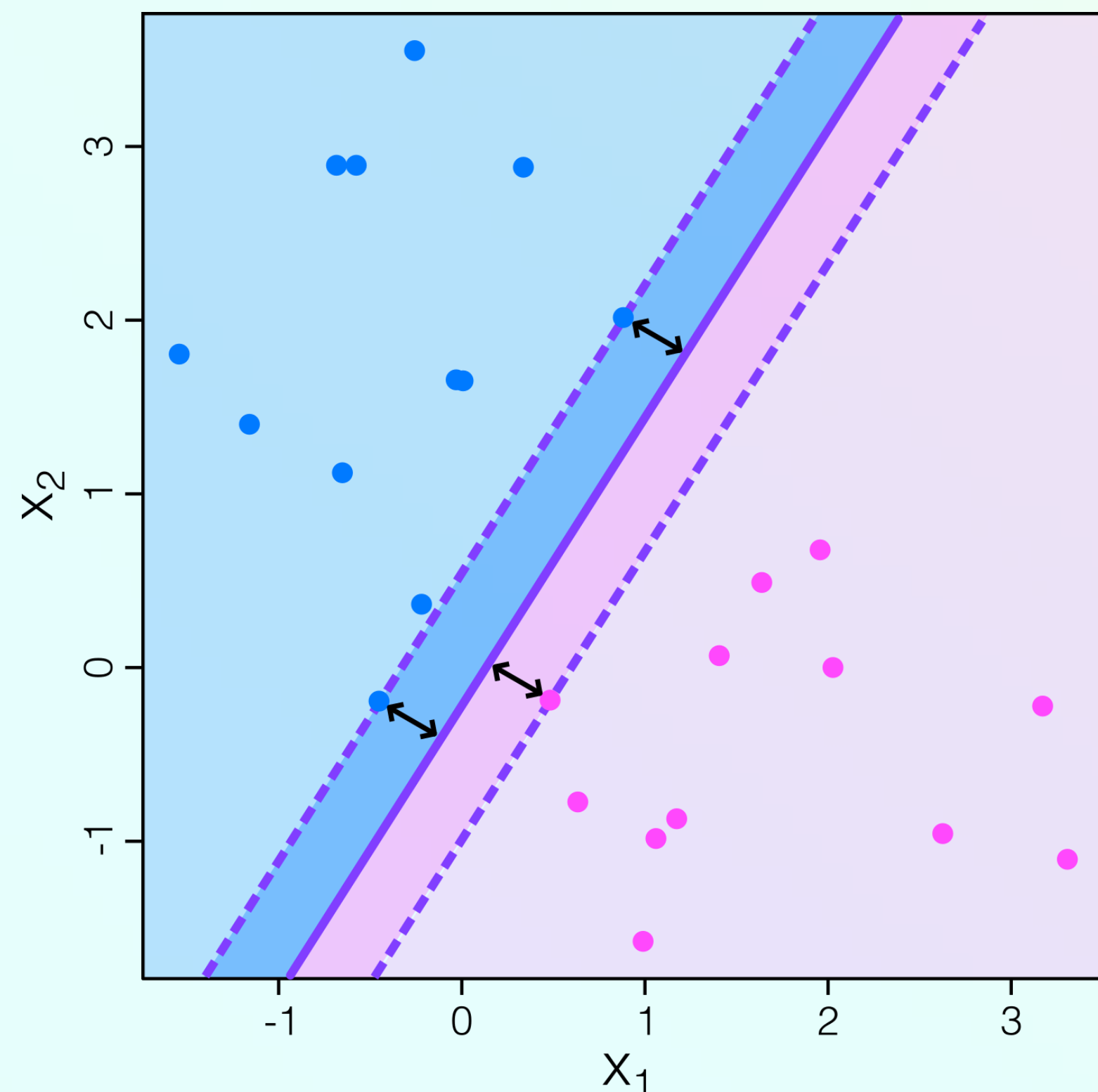
REPASO CLASE ANTERIOR

- Maximal Margin Classifier
- Clasificador de vector de soportes
- Support Vector Machines para clasificar
- Support Vector Machines para regresión

SVM se puede entender como la "busqueda" de la mejor separacion lineal entre dos conjuntos separables.

MAXIMAL MARGIN CLASSIFIER

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

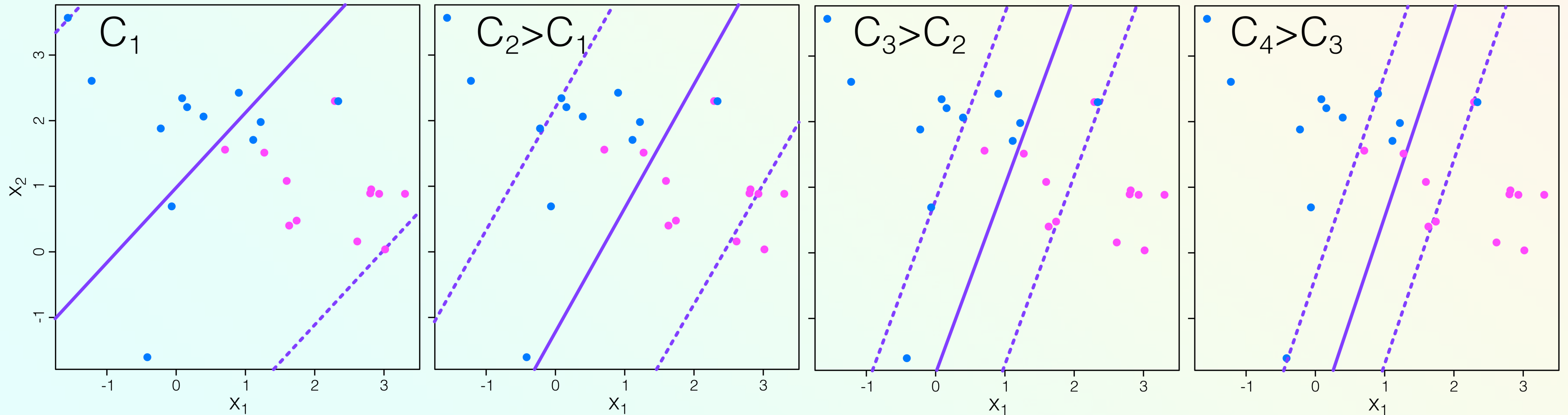
CLASIFICADOR DE VECTOR DE SOPORTES

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



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.

Cuanto mas grande es C , se reduce el margen de error, es decir se reduce el tamaño de las franjas.

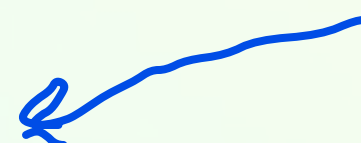
SUPPORT VECTOR MACHINE

El modelo llamado Support Vector Machine (SVM) o máquina de vector de soportes extiende al Clasificador de vector de soportes permitiendo extender el espacio de features, usando funciones kernels.

La frontera de decisión la podemos describir como:

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

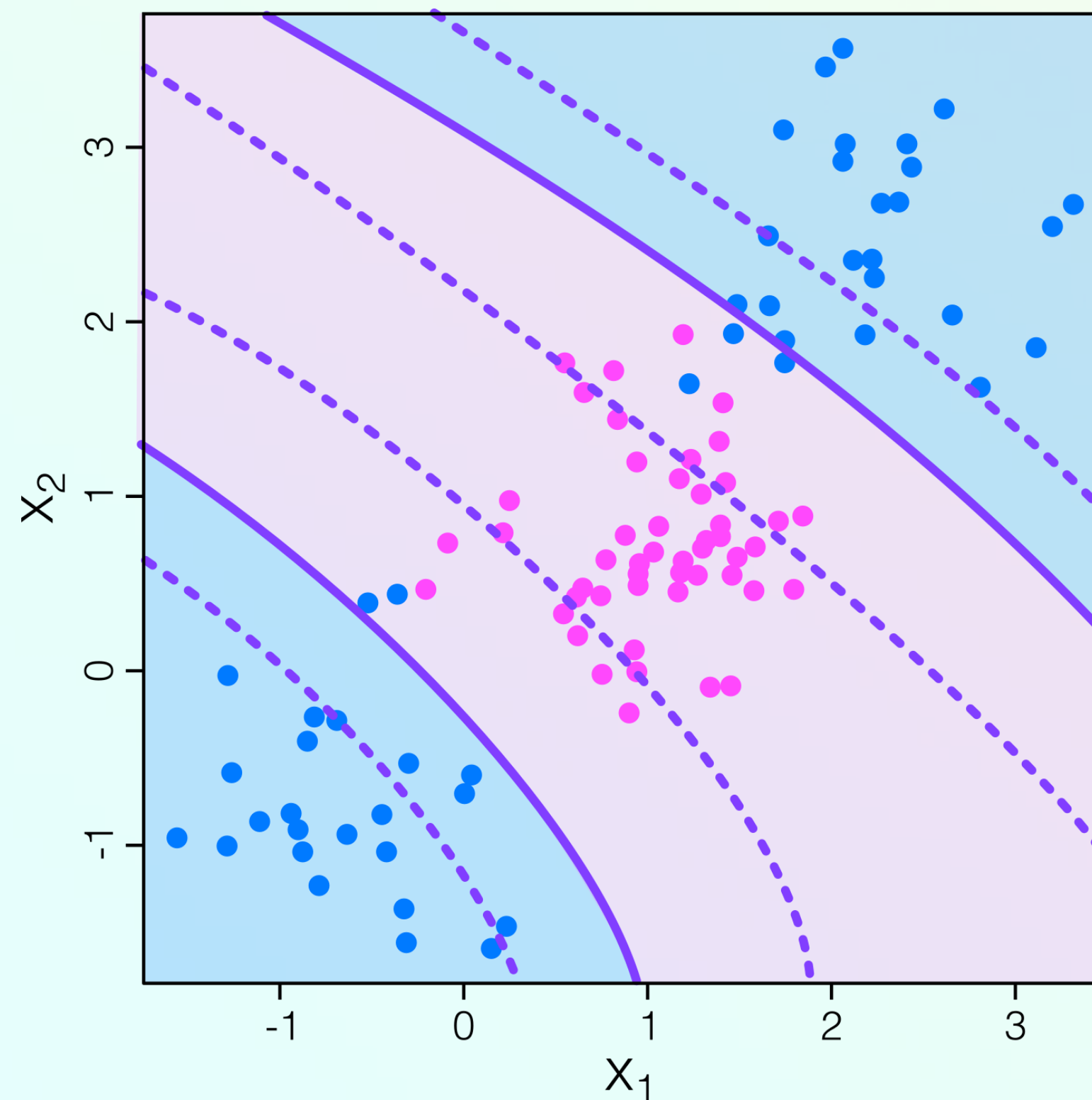
Kernel



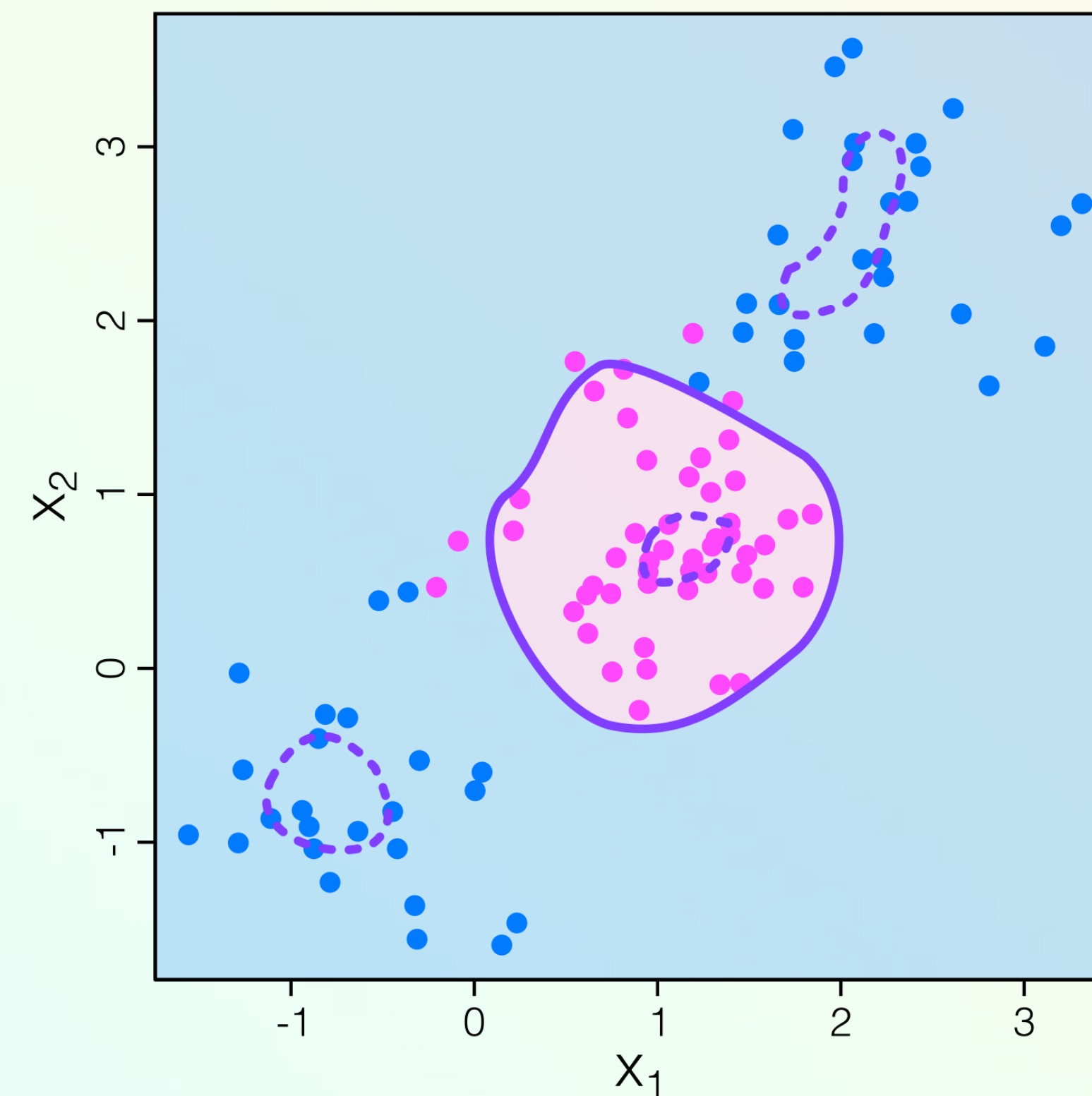
La forma de entrenamiento, la importancia de los vectores de soporte y el hiperparámetro C se mantienen. La gran diferencia es que ahora las fronteras de decisión no son necesariamente lineales y determinada por la función kernel elegida.

SUPPORT VECTOR MACHINE

El Kernel permite definir fronteras de decisiones no lineales, como polinómicas o radial

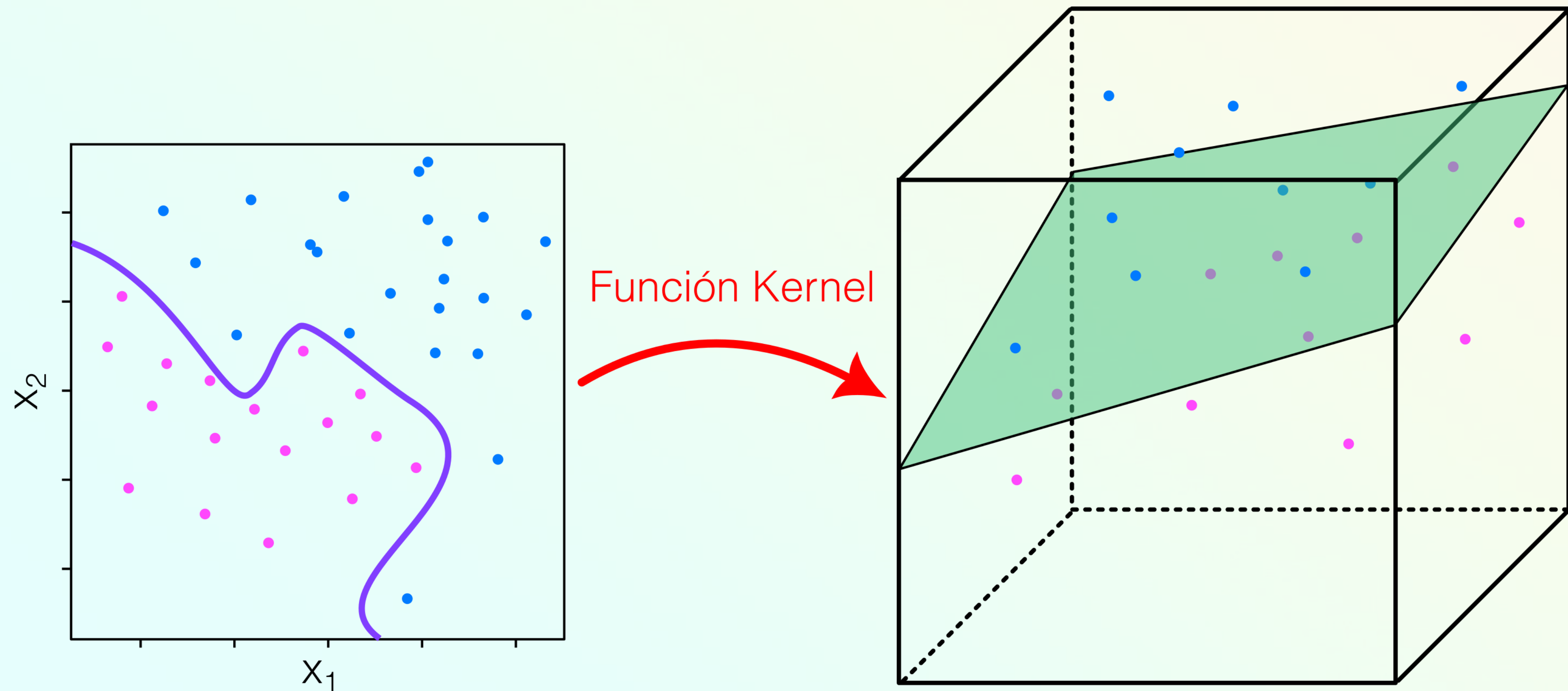


Kernel polinomial de orden 3



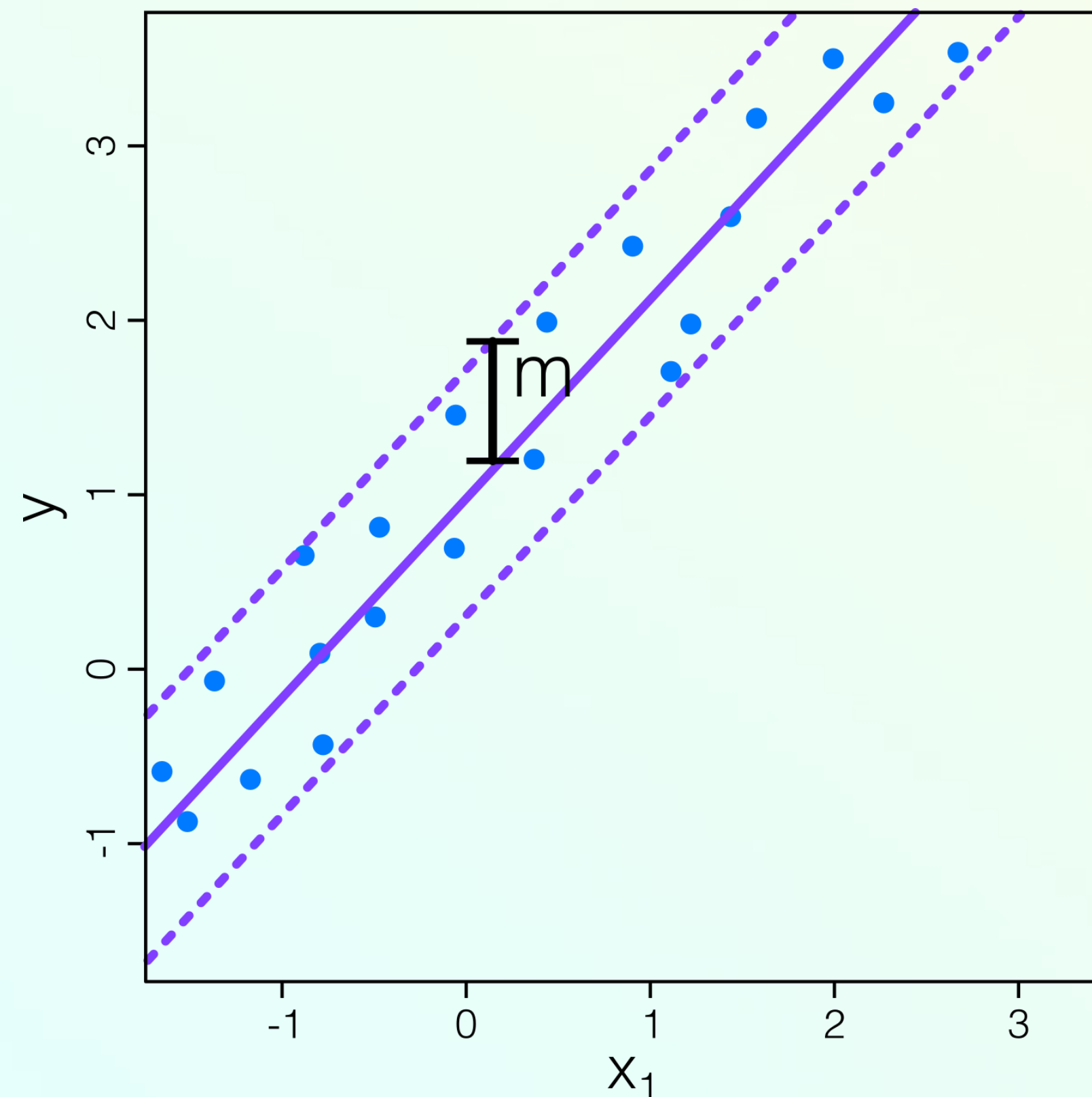
Kernel radial

SUPPORT VECTOR MACHINE



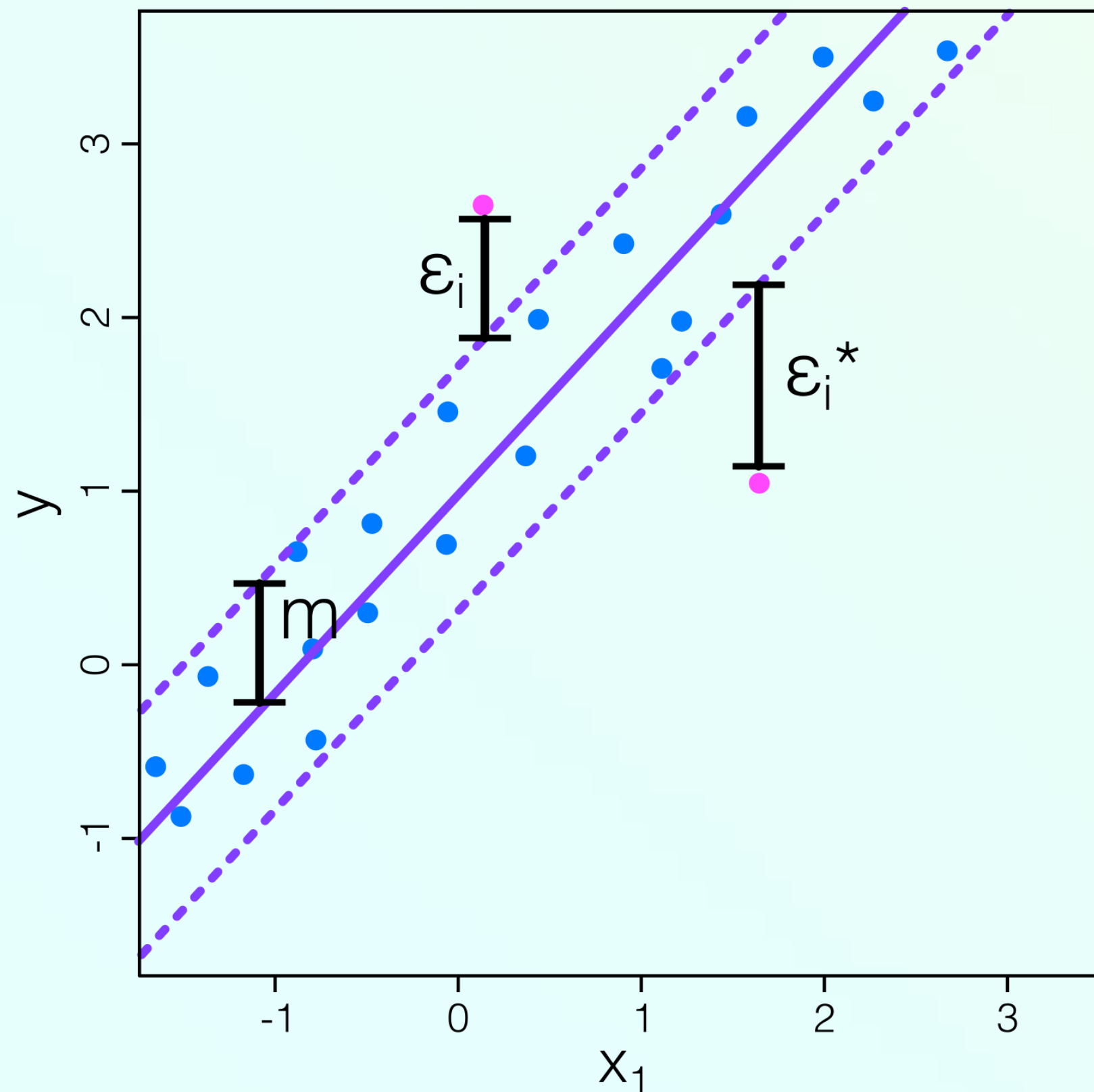
SUPPORT VECTOR MACHINE EN REGRESIÓN

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.



SUPPORT VECTOR MACHINE EN REGRESIÓN

SVR también es sensible a valores atípicos, por lo que se introduce el concepto de margen de error.



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

La restricción es:

$$y_i - (b_0 + \langle \vec{b}, X \rangle) \leq m + \epsilon_i \quad \forall i = 1, \dots, n$$

$$(b_0 + \langle \vec{b}, X \rangle) - y_i \leq m + \epsilon_i^*$$

ÁRBOLES DE DECISIÓN

ÁRBOLES 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.

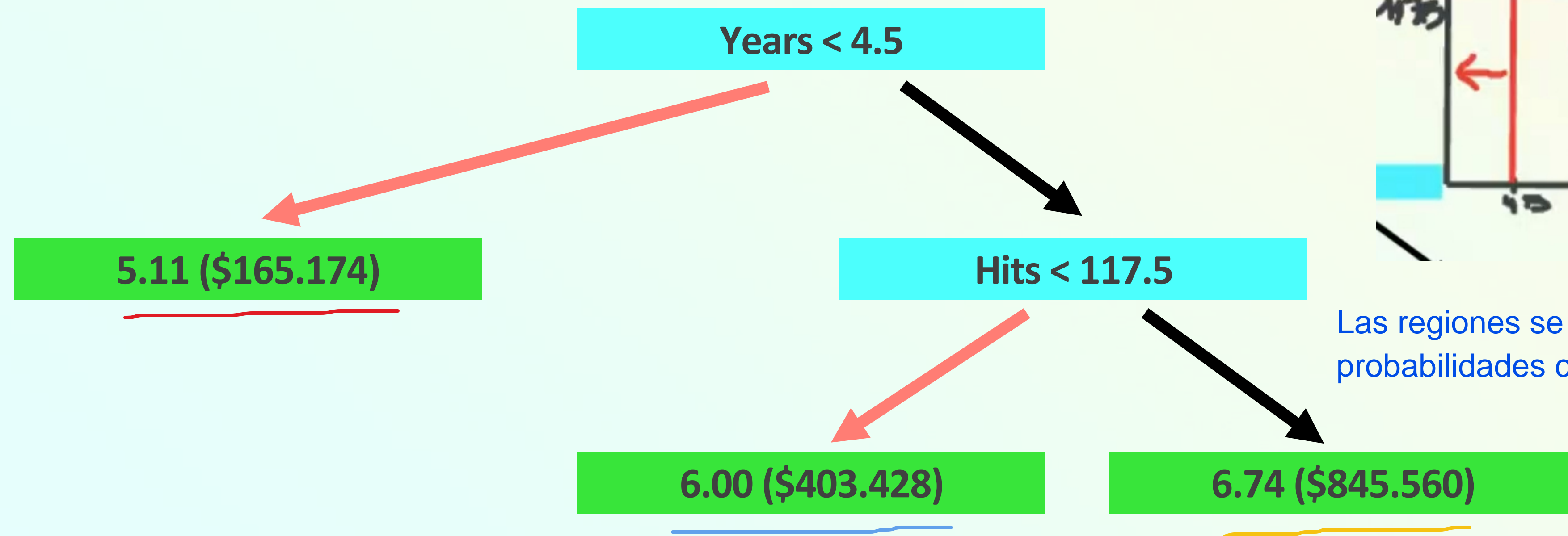
Arboles de clasificación

Arboles de regresión

- * En KNN se dio un enfoque radial para clasificacion
- * En SVM (original) se establece fronteras de decision lineales
- * En Arboles de decision, lo que se busca es establecer semi espacios y fronteras lineales. De aqui se definen cuadrículas donde el valor predicho toma el valor de la moda/media.

ÁRBOLES DE REGRESIÓN

Empecemos con un ejemplo sencillo, usando el dataset [Hitters](#) (Dato de salarios de jugadores de Beisbol de 1987 y estadísticas deportivas de 1986)

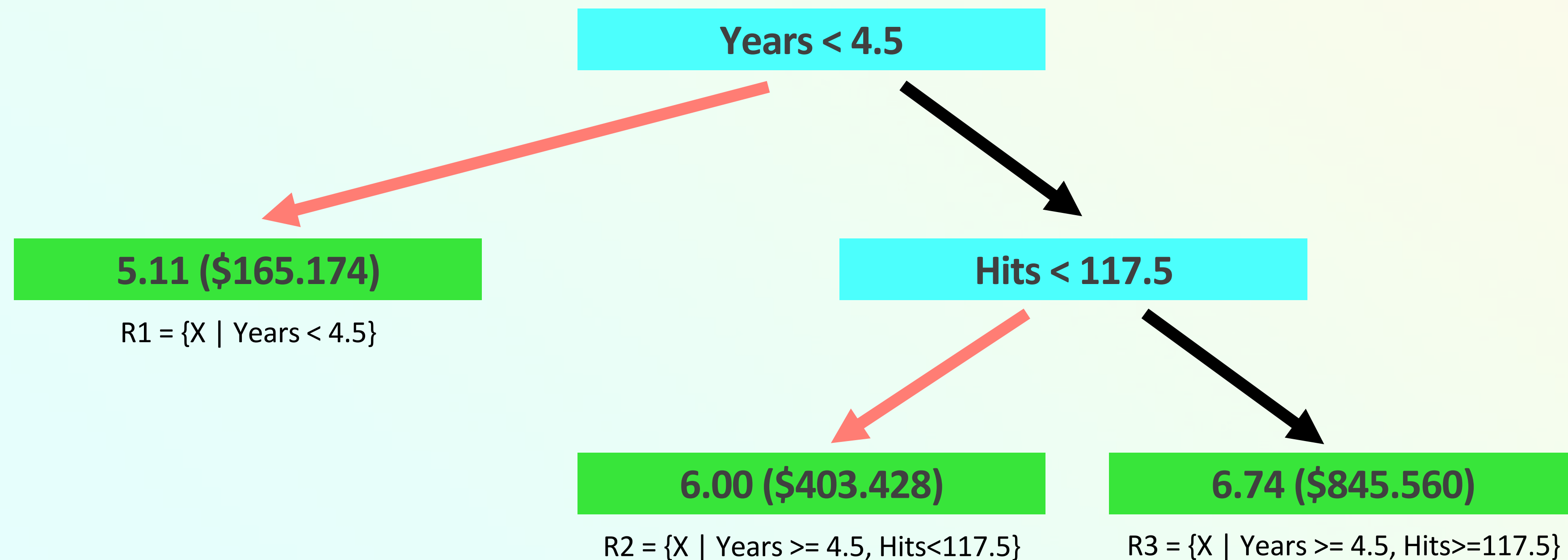


Las regiones se puede ver como probabilidades condicionadas

Se predice el logaritmo del salario (tiene una distribución más de campana). Years es años jugando en las ligas. Hits es el número de hits que realizó el año pasado.

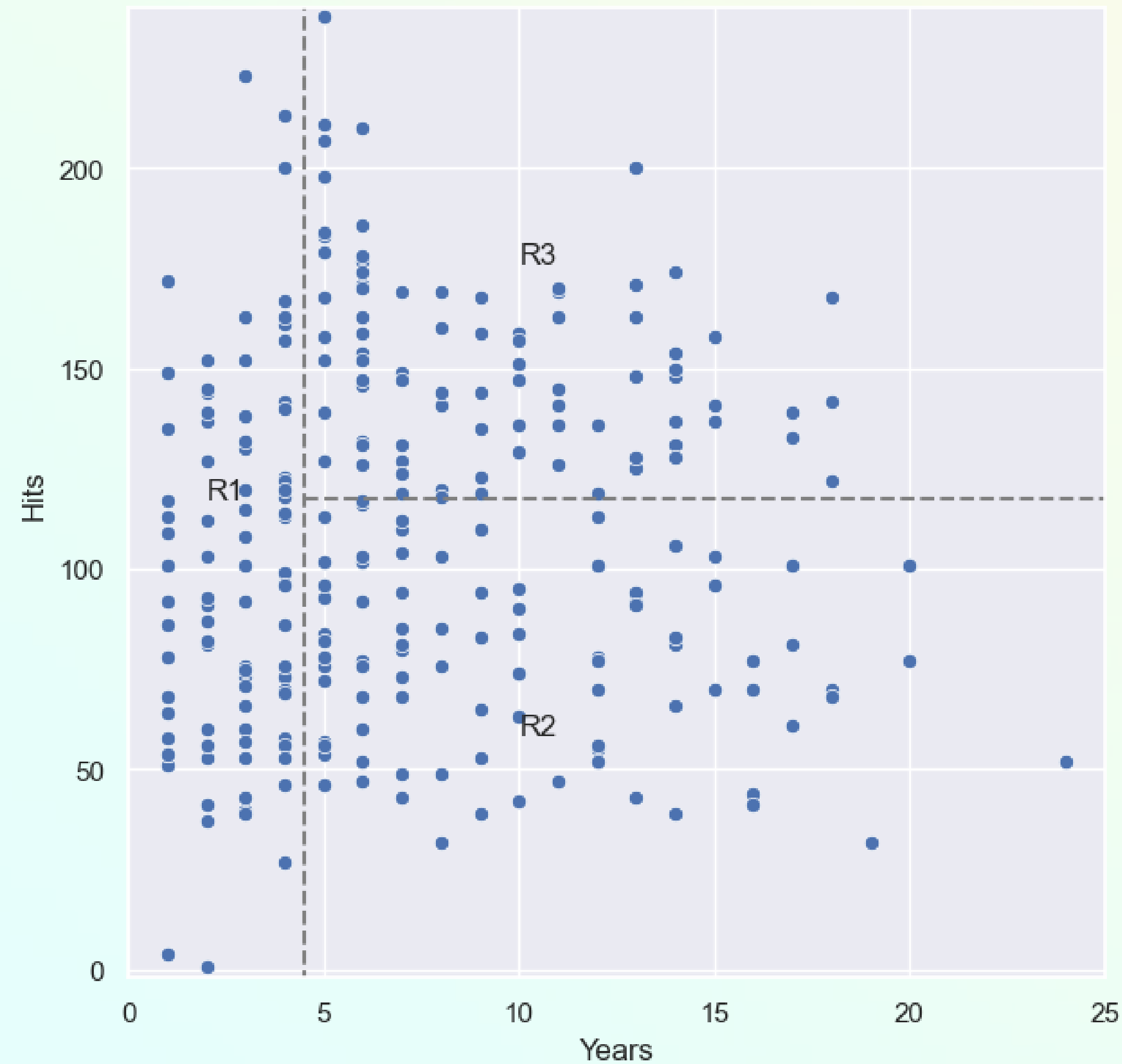
ÁRBOLES DE REGRESIÓN

Empecemos con un ejemplo sencillo, usando el dataset [Hitters](#) (Dato de salarios de jugadores de Beisbol de 1987 y estadísticas deportivas de 1986)



Se predice el logaritmo del salario (tiene una distribución más de campana). Years es años jugando en las ligas. Hits es el número de hits que realizó el año pasado.

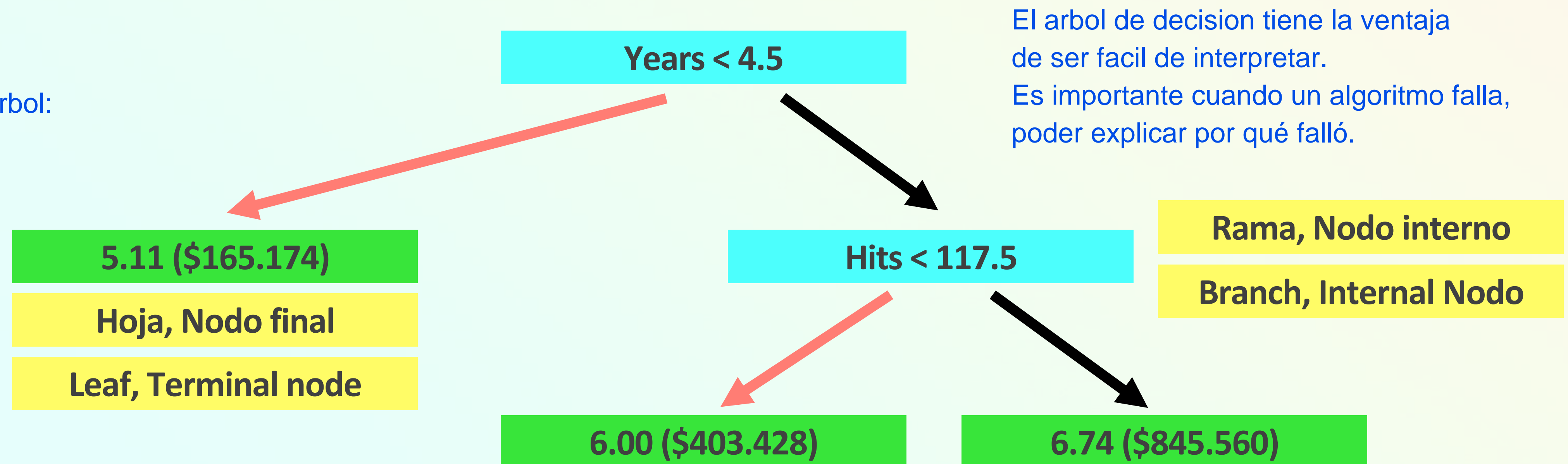
ÁRBOLES DE REGRESIÓN



ÁRBOLES DE REGRESIÓN

Empecemos con un ejemplo sencillo, usando el dataset [Hitters](#) (Dato de salarios de jugadores de Beisbol de 1987 y estadísticas deportivas de 1986)

Elementos de un arbol:



Este árbol de regresión es una sobre simplificación del verdadero valor de regresión entre Salary, Years e Hits. Sin embargo, tiene sus ventajas porque es más fácil entender y tienen mejor representación gráfica.

ÁRBOLES DE REGRESIÓN

Como construimos el proceso de construcción del árbol de regresión: El factor critico es donde establecemos el corte

1. Dividimos el espacio de observaciones, que son el set de los valores posibles X_1, X_2, \dots, X_p , en J regiones distintas y que no se solapan R_1, R_2, \dots, R_J .
2. Para cada observación que cae en una región R_j , hacemos la misma predicción, la cual es simplemente la media de la respuesta de los valores de entrenamiento que están en R_j .

Podemos usar otra métrica de medición de posición central.

ÁRBOLES DE REGRESIÓN

¿Cómo dividimos el espacio de observaciones?

En teoría, el espacio lo podríamos dividir en cualquier tipo de regiones, pero se elige espacios *rectangulares* para simplificar el modelo.

El objetivo es encontrar cajas R_1, \dots, R_J que minimice la suma al cuadrado de los residuos, (RSS) dado por:

Funcion de costo:

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

Media en R_j

Distancia al cuadrado de todos los puntos, de una region, respecto a su centro de masa/media

Error cuadratico medio de la region

Suma de todas las regiones

El inconveniente de esta funcion de costo, es que la suma siempre va a ser menor a medida que las regiones crecen. Cuantas mas regiones, las distancias de los puntos respecto a su media, es menor

ÁRBOLES DE REGRESIÓN

¿Cómo dividimos el espacio de observaciones?

En teoría, el espacio lo podríamos dividir en cualquier tipo de regiones, pero se elige espacios *rectangulares* para simplificar el modelo.

El objetivo es encontrar cajas R_1, \dots, R_J que minimice la suma al cuadrado de los residuos, dado por:

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

Es la media de y en la región R_j

ÁRBOLES DE REGRESIÓN

¿Cómo dividimos el espacio de observaciones?

Es imposible buscar todas las combinaciones posibles de valores para encontrar la minimizar a RSS.

Tenemos que usar algún algoritmo de optimización. Para ello tomamos un algoritmo top-down greedy que es conocido como Recursive binary splitting.

- Top-down: Arrancamos desde el tronco del árbol y vamos bajando. (Iterativo)
- Greedy: En cada paso, se busca la mejor bifurcación en ese paso particular. (Exahustivo)

ÁRBOLES DE REGRESIÓN

Recursive binary splitting

Se elije un X_j y el punto de corte s de tal forma que bifurca el espacio de features en dos regiones $\{X|X_j < s\}$ y $\{X|X_j \geq s\}$ que lleve a la mayor reducción de RSS.

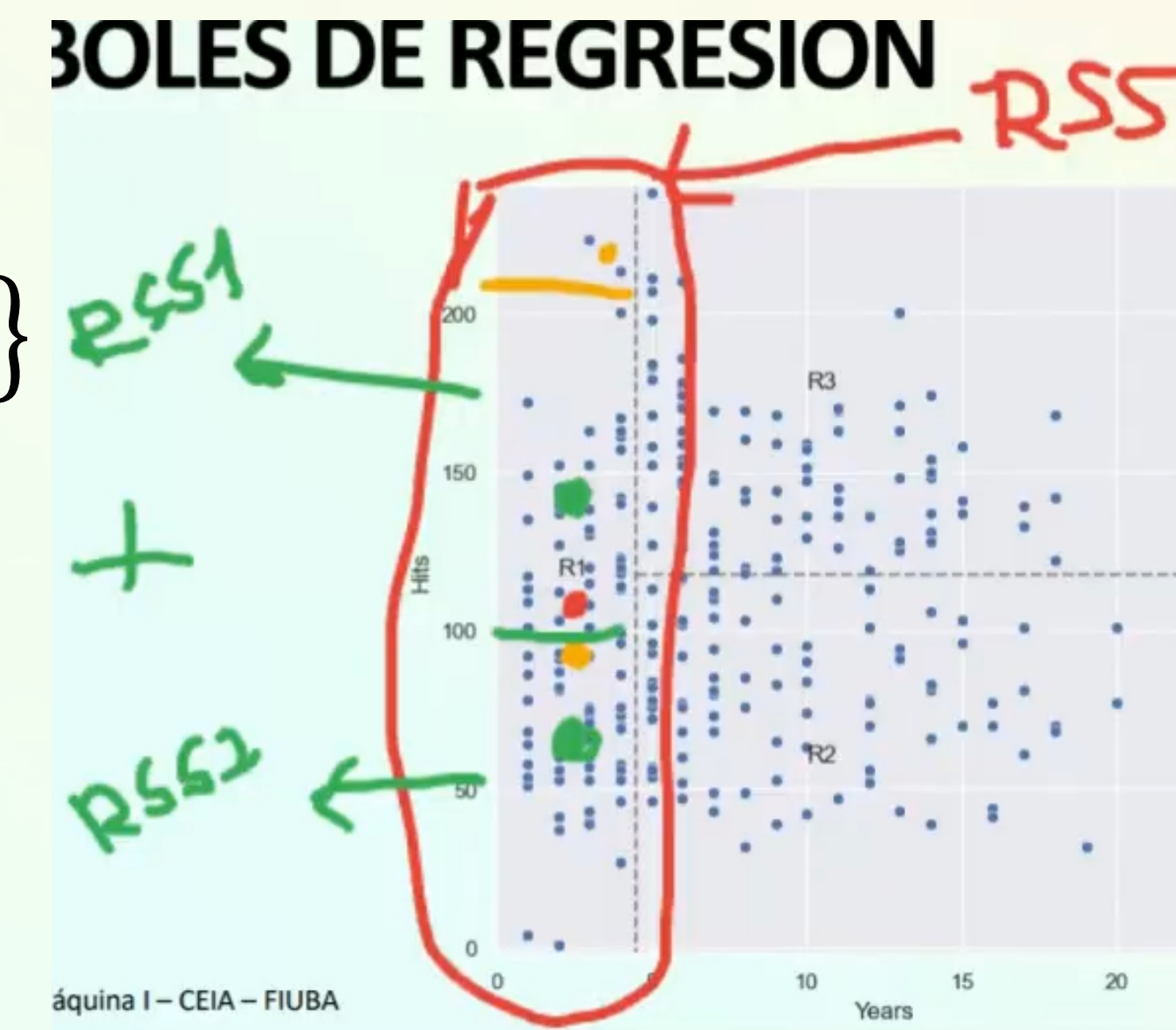
Es decir, para cada valor de j y cada valor de s :

$$R_1(j, s) = \{X|X_j < s\} \quad R_2(j, s) = \{X|X_j \geq s\}$$

Y buscamos el valor de j y s que minimice esta ecuación:

Es decir se busca minimizar los residuos de ambas mitades

$$\sum_{i:x_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i:x_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2$$



$RSS1 + RSS2$ siempre va a ser menor que RSS , pero lo que importa es cuanta distancia hay entre RSS y $RSS1+RSS2$. Esa distancia se llama ganancia de informacion.

ÁRBOLES DE REGRESIÓN

Recursive binary splitting

Y recursivamente repetimos esto para los segmentos que se generan, pero ahora tomando a las regiones formadas y aplicando este proceso, partimos en nuevas regiones.

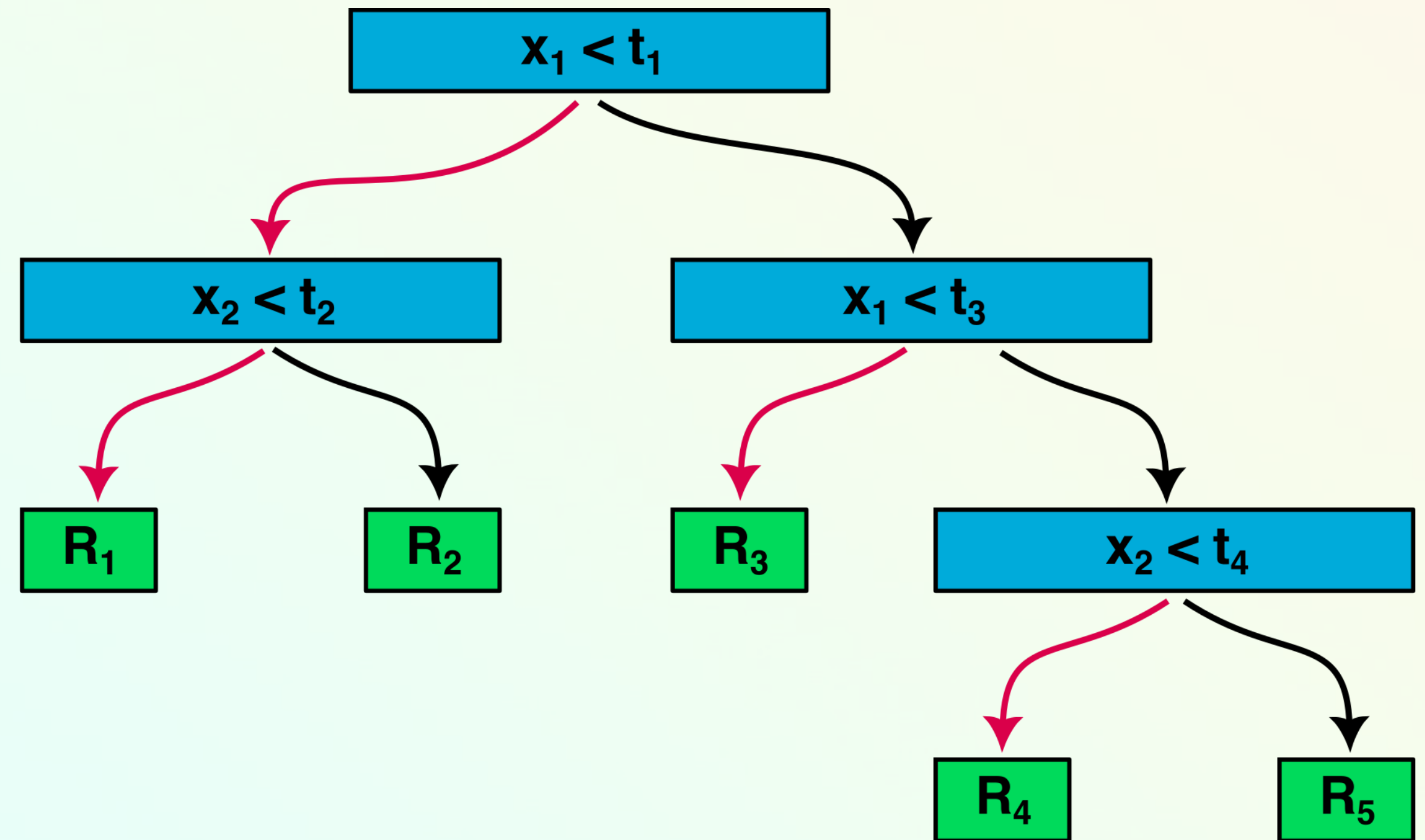
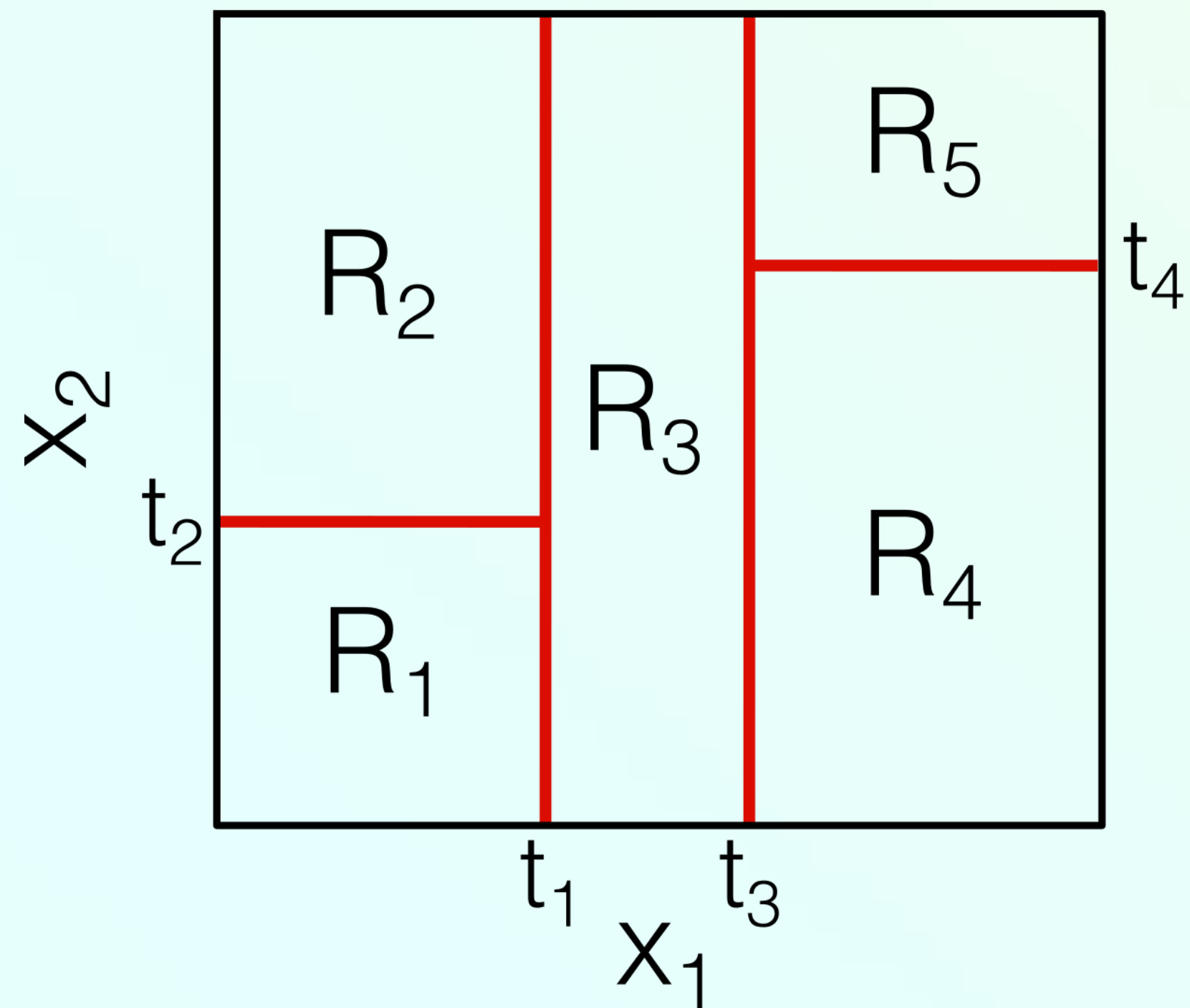
Este proceso continua hasta que llegamos a un criterio de corte.

Ejemplos de corte:

- Hasta que el arbol alcance una profundidad de 30.
- La diferencia entre la RSS ("padre") y sus particiones, sean menor que cierto umbral.

ÁRBOLES DE REGRESIÓN

Recursive binary splitting



ÁRBOLES DE REGRESIÓN

Podando los arboles

El proceso que se describió puede producir buenas predicciones del set de entrenamiento, pero muy fácilmente puede generar overfitting, haciendo que se desempeñe muy mal en el set de validación.

El caso más extremo es un árbol con una hoja por cada punto del set de entrenamiento.

Esto se debe a que el árbol es muy complejo. Un árbol más pequeño con menor regiones puede llevar a menos **varianza** y mejor interpretación a expensa de un poco de **sesgo**.

ÁRBOLES DE REGRESIÓN

Podando los arboles

La estrategia más obvia es construir el árbol sólo mientras la disminución en el **RSS** sea mayor a un valor umbral (relativamente alto). Esta estrategia dará como resultado árboles más pequeños, pero es demasiado cortoplacista,

Una división aparentemente sin valor en las primeras etapas del árbol podría ser seguida por una división muy buena, es decir, una división que conduzca a una gran reducción del RSS más adelante.

ÁRBOLES DE REGRESIÓN

Podando los arboles

Una mejor estrategia es llevar un paso de eliminación hacia atrás. Construimos un árbol enorme T_0 , y luego vamos podando para obtener un sub-árbol.

¿Como hacemos? Intuitivamente es elegir un sub-árbol que disminuya el error de validación.

Pero de nuevo, **estamos ante un problema demasiado complejo de iterar.**

ÁRBOLES DE REGRESIÓN

Podando los arboles

Vamos a introducir 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:

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

Ahora esta funcion ya no es siempre decreciente,
la sumatoria baja, pero el termino de alfa puede subir

...es mínimo.

L indica el número de hojas del sub-árbol.

α ($\alpha \geq 0$) presenta un trade-off entre complejidad del árbol y su capacidad de ajustar a los datos (Regularización). Si $\alpha=0$ el árbol elegido es T_0 . Cuando más grande es α , el precio a pagar por el tamaño de árbol cada vez es mayor.

ÁRBOLES DE REGRESIÓN

Podando los arboles

Algo interesante es que a medida que aumentamos α desde cero, las ramas se podan del árbol de una manera anidada y predecible, por lo que es fácil obtener la secuencia completa de subárboles en función de α .

Podemos seleccionar un valor de α usando búsqueda de hiper-parámetros.

ÁRBOLES DE REGRESIÓN

Algoritmo total

1. Usando Recursive binary splitting se crea el árbol más grande con el dataset de entrenamiento, terminando el entrenamiento cuando cada hoja tenga menos de un número determinado de observaciones.
2. Se aplica la técnica de podado de árbol para obtener un set de sub-arboles como función de α . Usando validación cruzada para elegir α .
3. Elija el sub-árbol del paso 2 que corresponde al valor de α que disminuya el error.

VAMOS A PRÁCTICAR UN POCO...

ÁRBOL DE CLASIFICACIÓN

ÁRBOL DE CLASIFICACIÓN

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

En el caso de regresión, al llegar la hoja, obteníamos el valor con el promedio de los valores en la hoja. Ahora, 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.

ÁRBOL DE CLASIFICACIÓN

La forma más simple en que se crea un árbol de clasificación es muy parecida al árbol de regresión con la estrategia top-down y greedy, pero no contamos con el error cuadrático.

Una primera métrica que podemos usar es la tasa de error de clasificación:

$$E = 1 - \max_k(\hat{p}_{mk})$$

El inconveniente es que la probabilidad depende linealmente de la clase mayoritaria.

Es la fracción de las observaciones de entrenamiento en esa región que no pertenecen a la clase más común.

\hat{p}_{mk} representa la proporción de las observaciones de entrenamiento en la región m que son de la clase k .

Ojo, el error de clasificación **no es suficientemente sensible para crecer a los árboles** y, en la práctica, son preferibles otras dos medidas.

ÁRBOL DE CLASIFICACIÓN

El índice de Gini, el cual es una medida de la desigualdad usada inicialmente para medir la desigualdad de los países:

$$G = \sum_{k=1}^K \hat{p}_{mk} (1 - \hat{p}_{mk})$$

Aca se tiene en cuenta TODAS las probabilidades que estan en juego.

Como una medida de la varianza a través de todas las clases K. El cual se observa que el índice de Gini toma un valor pequeño si todas las \hat{p}_{mk} son cercanas a cero o uno.

Por esto, se dice que el índice de Gini es una medida de la pureza de un nodo terminal

El indice de Gini penaliza mas las probabilidades intermedias y favorece los valores en los extremos (pertenece o no a una clase).

ÁRBOL DE CLASIFICACIÓN

En un árbol de clasificación, queremos encontrar una rama de decisión que de mucha información.

Si en una rama, hace que todas las observaciones de una clase vayan para un lado y todas las otras observaciones de otra clase vayan para la otra, va a ser una excelente rama para elegir.

En cambio, el caso que no afecta a como se mueven las clases, probablemente no es una buena opción.

La forma que capturamos esta noción de cuanta información transmite es con Entropía. O también como medida de desorden.

ÁRBOL DE CLASIFICACIÓN

Si en una hoja, todas las observaciones de entrenamiento son de una sola clase, la entropía es cero.

En cambio, si las clases están desperdigadas de forma uniforme entre la clase, la entropía es grande.

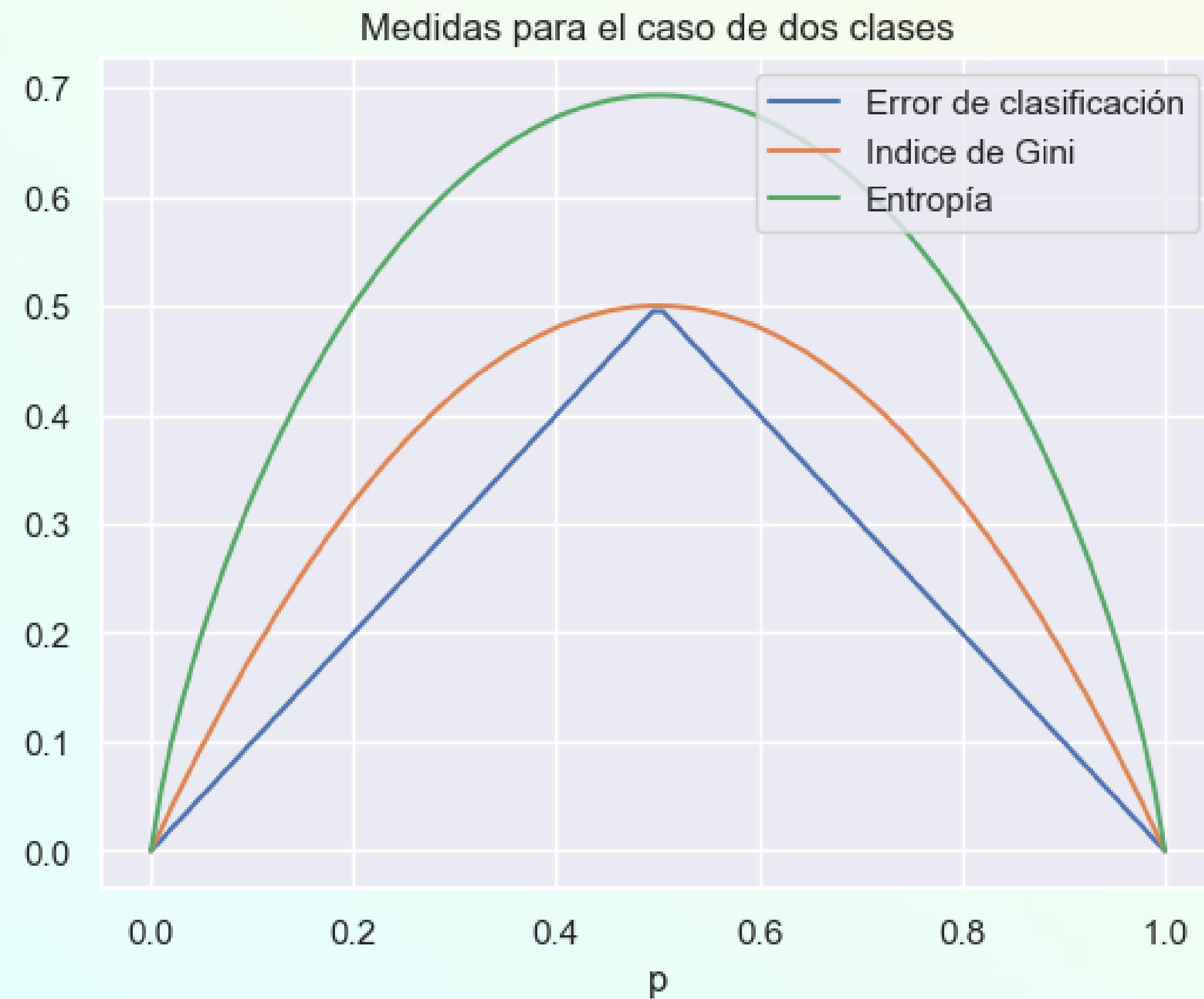
La definición de entropía es:

$$D = - \sum_{k=1}^K \hat{p}_{mk} \log(\hat{p}_{mk})$$

La entropía viene de la teoría de información.

Vemos que si \hat{p}_{mk} son cercanos a cero o a uno, la entropía es un valor pequeño. Si las proporciones son similares entre sí, este valor va a ser grande.

ÁRBOL DE CLASIFICACIÓN



Como se menciona, la entropía penaliza mucho mas a las probabilidades que no estan en los extremos.

ARBOL DE CLASIFICACIÓN

Todo el resto es igual al árbol de regresión

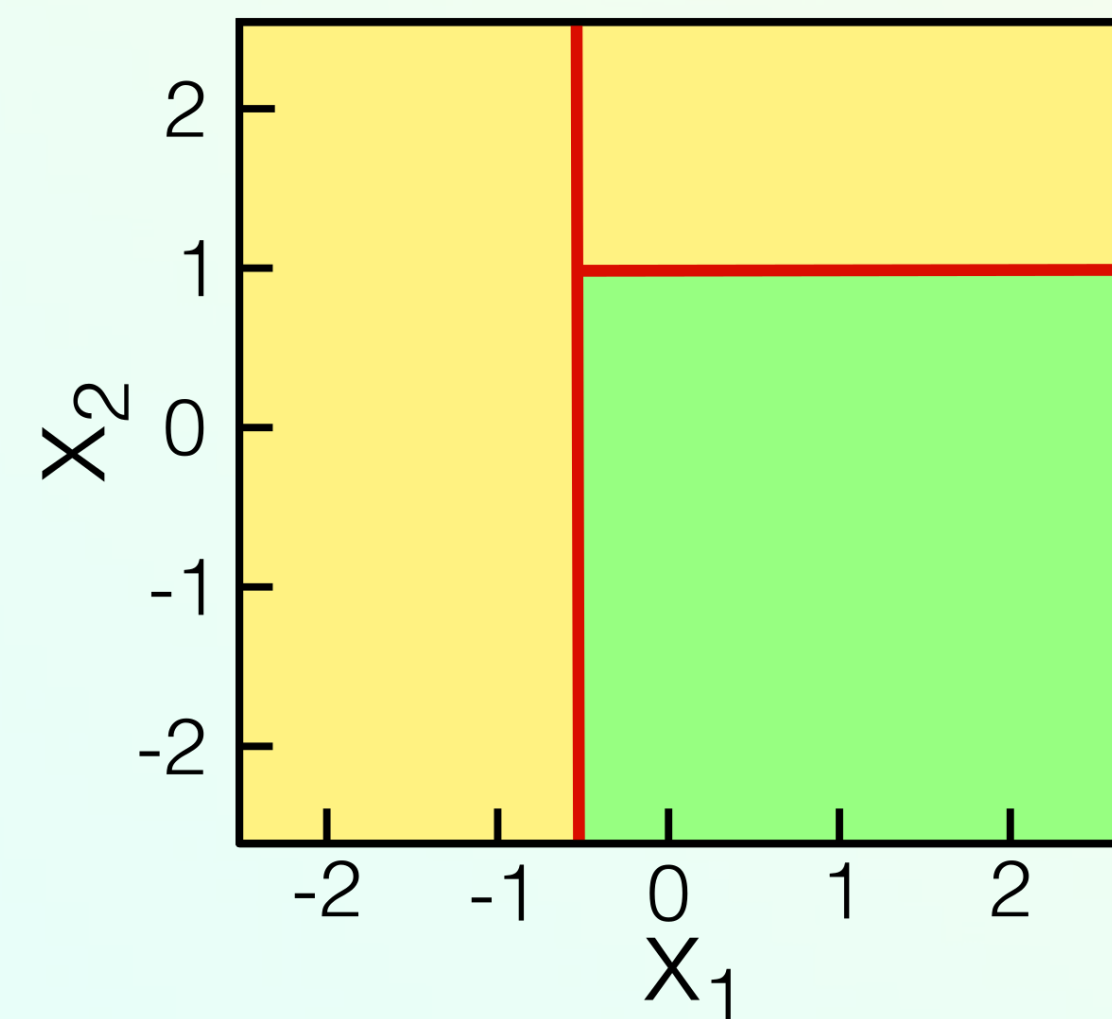
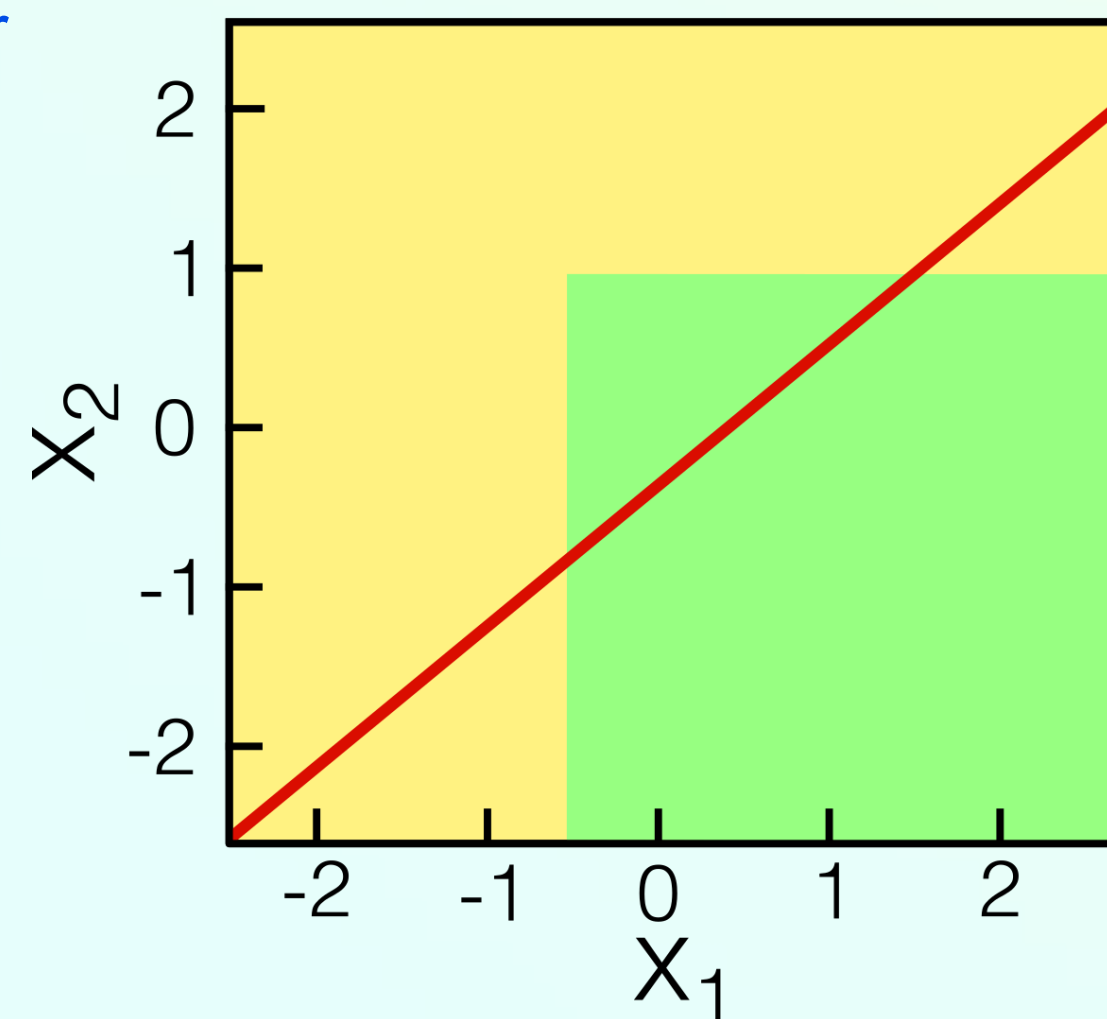
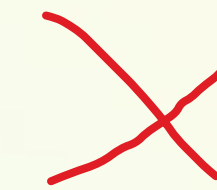
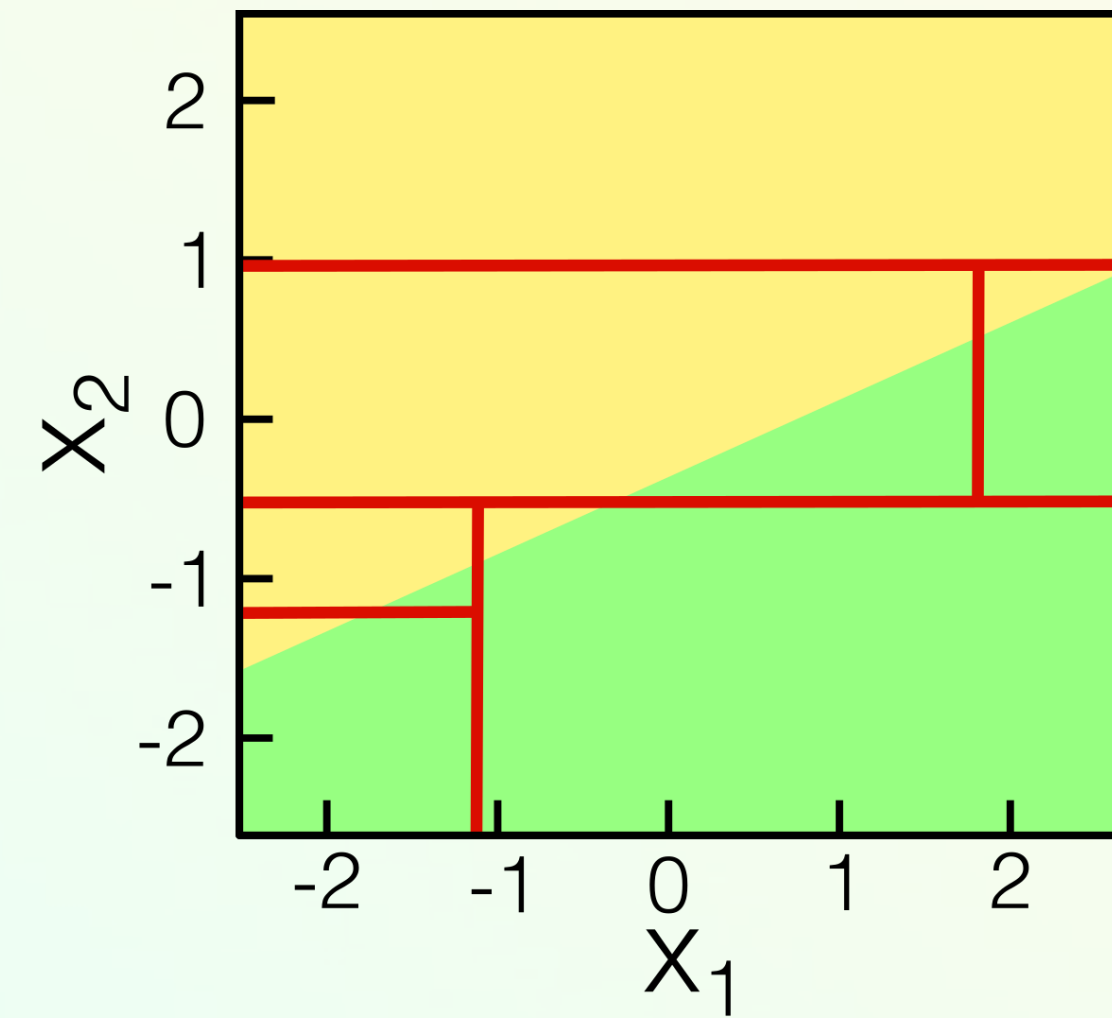
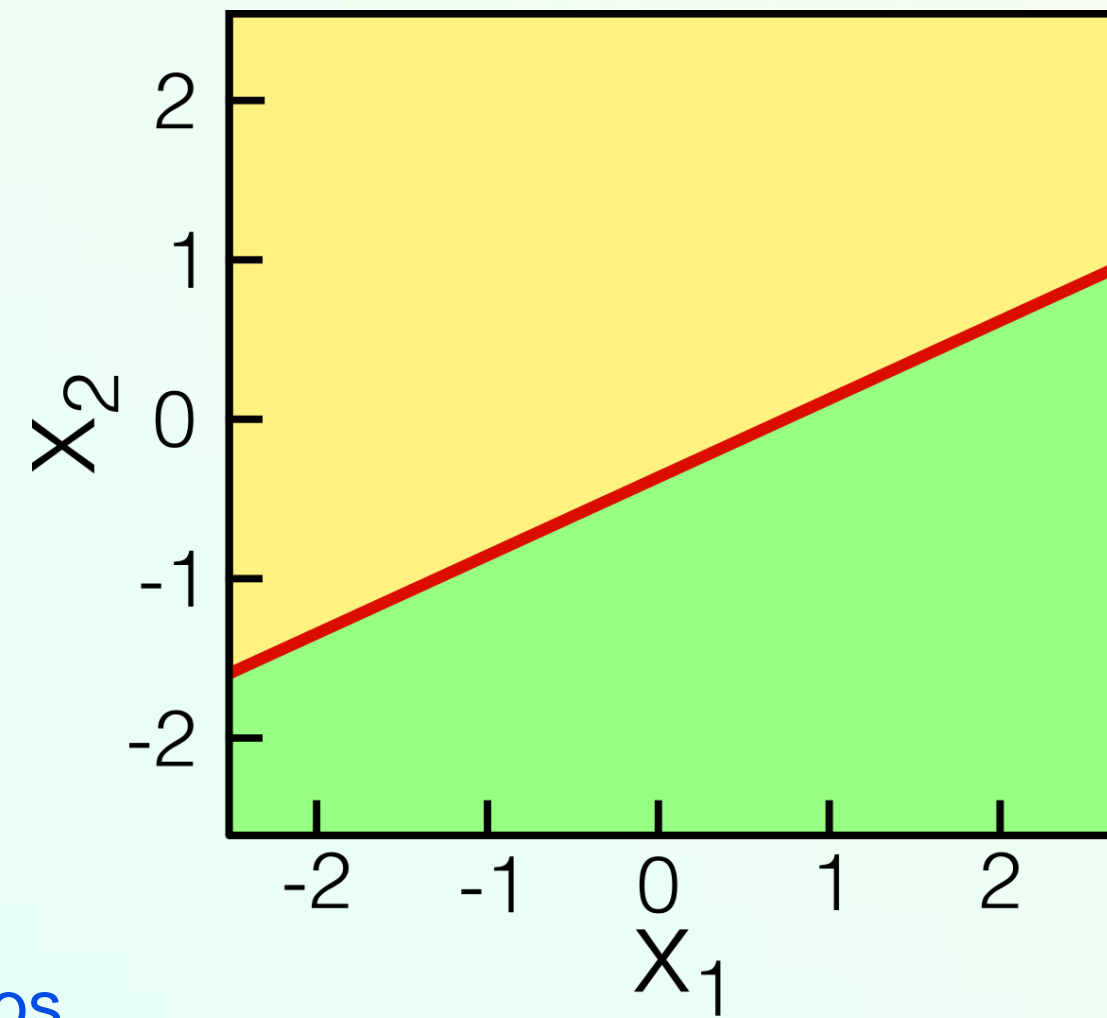
VENTAJAS Y DESVENTAJAS DE LOS ÁRBOLES

- ✓ Son fáciles de explicar a las personas, más inclusive que a la regresión lineal.
- ✓ Se hipotetiza que el árbol de decisión se acerca más a la forma que un humano piensa.
- ✓ Se puede representar gráficamente.
- ✓ Los árboles puede manejar fácilmente variables cualitativas sin necesidad de crear variables dummy.
- ⊘ No tienen el mismo nivel de exactitud de predicción que otros modelos
- ⊘ No son robustos, pequeños cambios en los datos pueden cambiar grandes cambios en la predicción.

Pueden tener mucho overfitting, si cada particion tiene pocas muestras o la profundidad es muy grande.
Pero este algoritmo es la base de otros mas poderosos.

VENTAJAS Y DESVENTAJAS DE LOS ÁRBOLES

Por lo tanto, el uso del algoritmo depende mucho del dominio, como esten distribuidos los datos, la densidad de los puntos, y por ende es conveniente probar distintos algoritmos para un mismo caso.



mejor fit

VAMOS A PRÁCTICAR UN POCO...