



ESCUELA DE INGENIERÍA
FACULTAD DE INGENIERÍA

EDUCACIÓN
PROFESIONAL

Introducción a la Ciencia de Datos con R

Miguel Jorquera

Educación Profesional
Escuela de Ingeniería

El uso de apuntes de clases estará reservado para finalidades académicas. La reproducción total o parcial de los mismos por cualquier medio, así como su difusión y distribución a terceras personas no está permitida, salvo con autorización del autor.



ESCUELA DE INGENIERÍA
FACULTAD DE INGENIERÍA

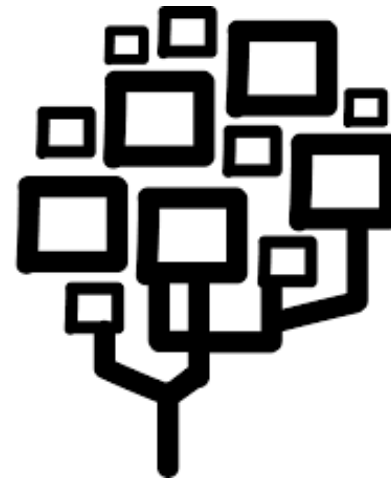
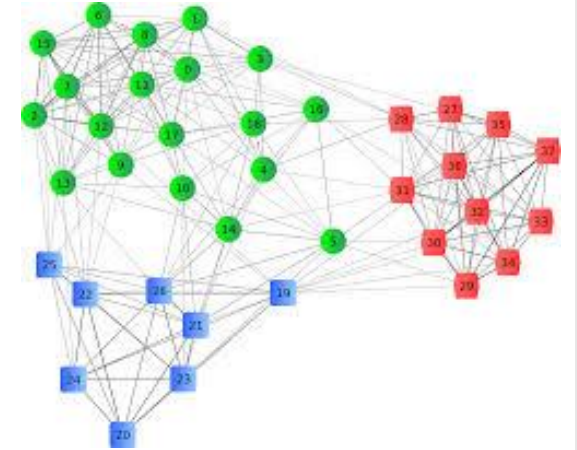
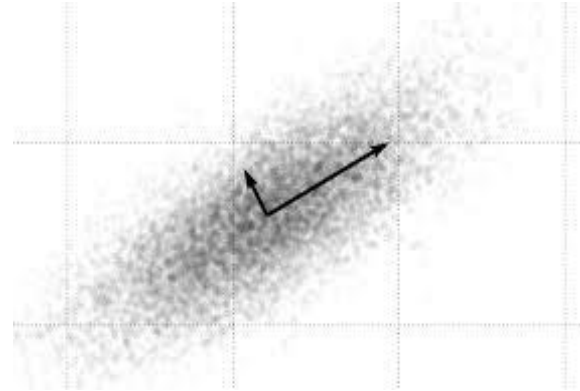
EDUCACIÓN
PROFESIONAL

RESUMEN

Aprendizaje no supervisado

Algunas tareas usuales:

- Reducción de la dimensionalidad
- Generación de clusters
 - Comenzamos con K-means
- Reglas de asociación
 - Estudiamos el algoritmo apriori





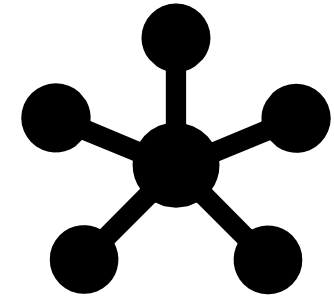
ESCUELA DE INGENIERÍA
FACULTAD DE INGENIERÍA

EDUCACIÓN
PROFESIONAL

TEMAS PARA HOY

Temas para hoy (y siguiente clase)

- Clustering (continuación)
- Introducción a modelos supervisados
 - Regresión Lineal
 - Validación y flexibilidad de métodos
 - Modelos de clasificación
 - Árboles de decisión
 - Métricas de evaluación
 - Métodos ensamblados
 - Regresión logística
 - Support Vector machines





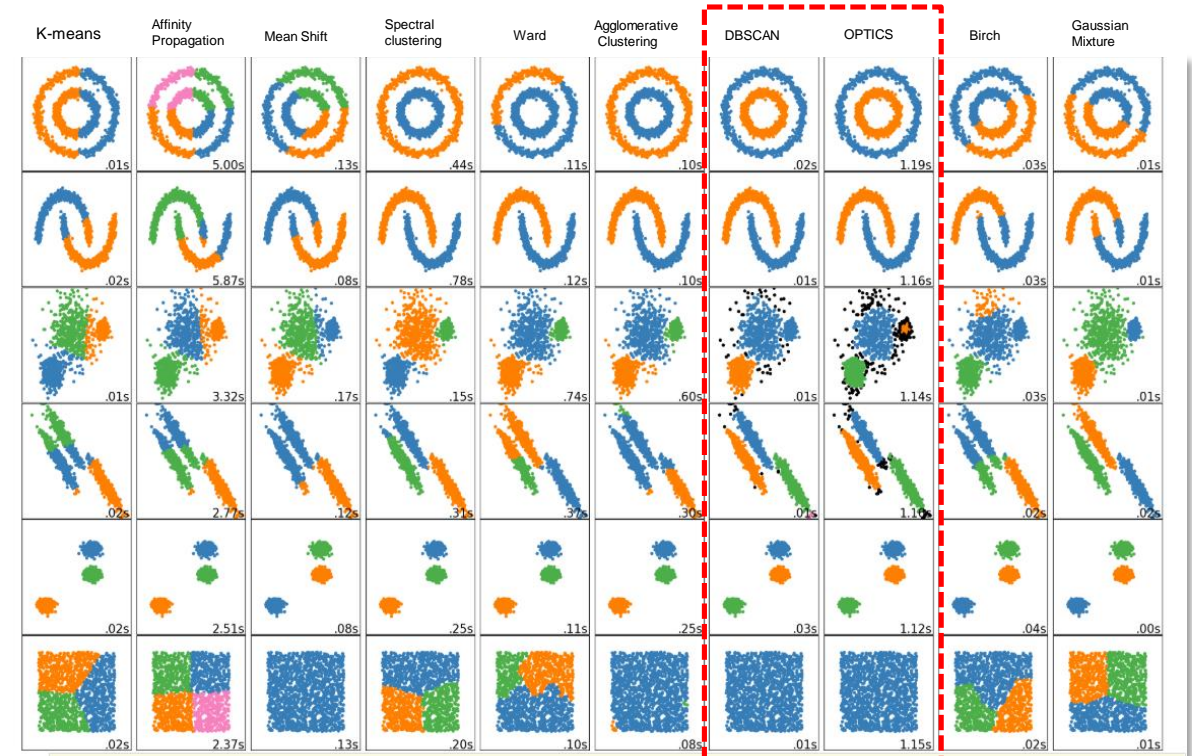
ESCUELA DE INGENIERÍA
FACULTAD DE INGENIERÍA

EDUCACIÓN
PROFESIONAL

CLUSTERING (CONTINUACIÓN)

Clustering jerárquico

- ▶ Los métodos de clustering son sensibles a la forma de los conjuntos que buscamos agrupar.
- ▶ La siguiente comparativa muestra la forma en que logran caracterizar los grupos los distintos métodos.
- ▶ Otro enfoque que permite aislar los puntos que están fuera de zonas “densas” son los algoritmos DBSCAN y OPTICS, los que a su vez pueden ser utilizados como un método de detección de anomalías.



En esta imagen de referencia se destacan en color naranja, azul y verde, los clústers generados por cada algoritmo, mientras que los puntos destacados en negro corresponden a puntos con baja densidad local, y por ende considerados como atípicos.

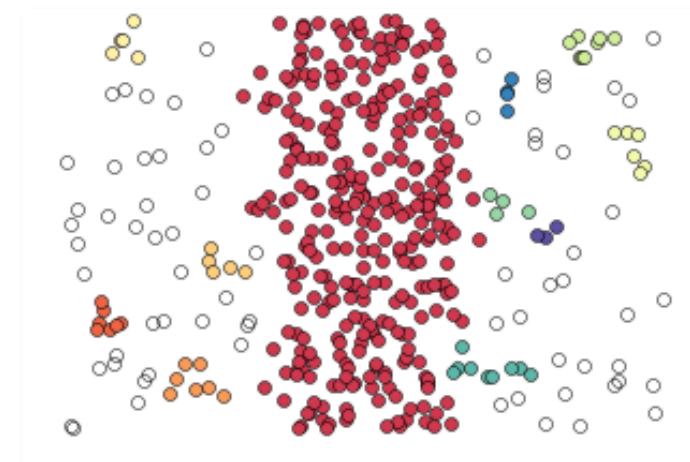
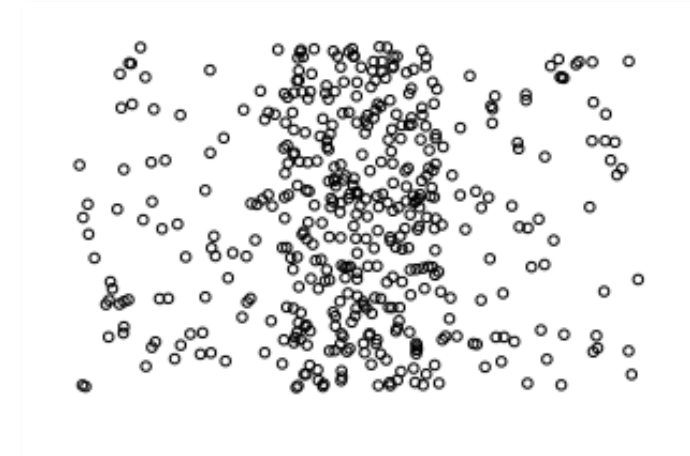


DBSCAN

Density-based spatial clustering of
applications with noise

Idea del algoritmo

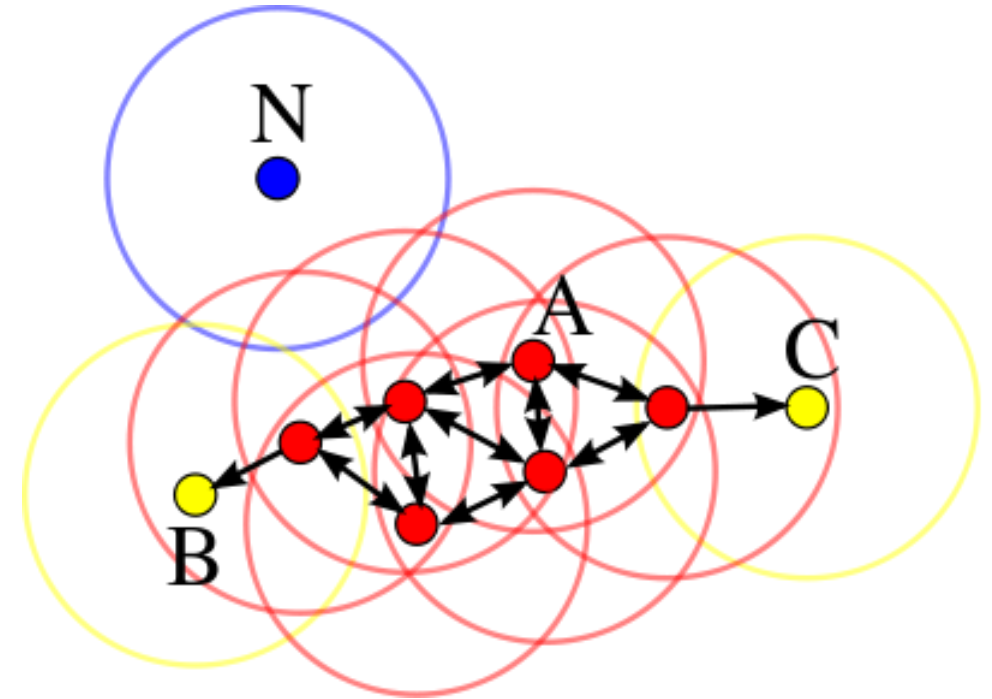
1. Se eligen 2 parámetros y un valor ϵ (eps) y un número de puntos (minPts).
2. Se elige un punto aleatorio del espacio
3. Se buscan todos aquellos puntos, que estén a una distancia igual o menor a ϵ del punto inicial.
 1. Si hay más de minPts puntos (incluyendo el inicial), se eligen sólo minPts
4. Se expande el cluster , revisando todos los nuevos puntos, y se ve si ellos forman un cluster, incrementando el cluster recursivamente.
5. Cuando se acaban los puntos, se toma un nuevo punto y se comienza de nuevo.
6. Aquellos puntos que no entran (por estar más distantes de e de los otros) se les considera "ruido" y quedan aislados.



DBSCAN Density-based spatial clustering of applications with noise

Conceptos claves

1. Core-point: Si al menos minPts están en la vecindad de radio eps de p
2. Directly Reachable-point: Un punto q es alcanzable desde p (core-point) si está a distancia menor que eps
3. Reachable-point: un punto q es densamente alcanzable desde p , si existe una ruta de puntos p_1, \dots, p_n , con $p_1 = p$ y $p_n = q$, donde $p_{(i+1)}$ es directamente alcanzable desde p_i . Notar que p_i es punto núcleo con la posible salvedad de q .
4. Outlier (noise point): Un punto que no sea directa o densamente alcanzable desde cualquier otro punto.



$\text{minPts} = 4$

A: Puntos núcleos

B, C: Puntos densamente alcanzables

N: Ruido

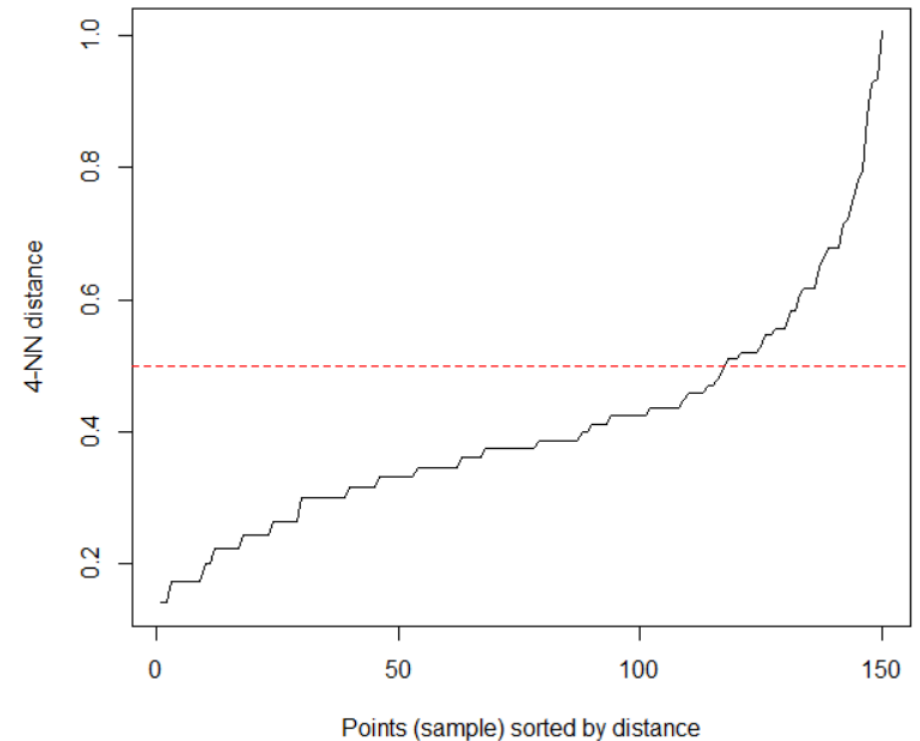


DBSCAN

Density-based spatial clustering of
applications with noise

Notar que los dos parámetros ϵ y minPts , deben ser definidos inicialmente. Estos determinarán la calidad de la partición generada.

- Usualmente se considera minPts como $p+1$, donde p es la dimensión del espacio de atributos de nuestro dataset.
- Una manera de escoger el valor de ϵ es mediante la visualización de las KNN distancias de cada punto.
 - Para cada punto se calcula la distancia al $k = \text{minPts}$ vecino más cercano.
 - Se ordenan las distancias de menor a mayor.
 - Se busca el punto de mayor crecimiento (regla del "codo")



OPTICS

Ordering Points To Identify Clustering Structure

Una extensión del algoritmo DBSCAN corresponde al algoritmo OPTICS

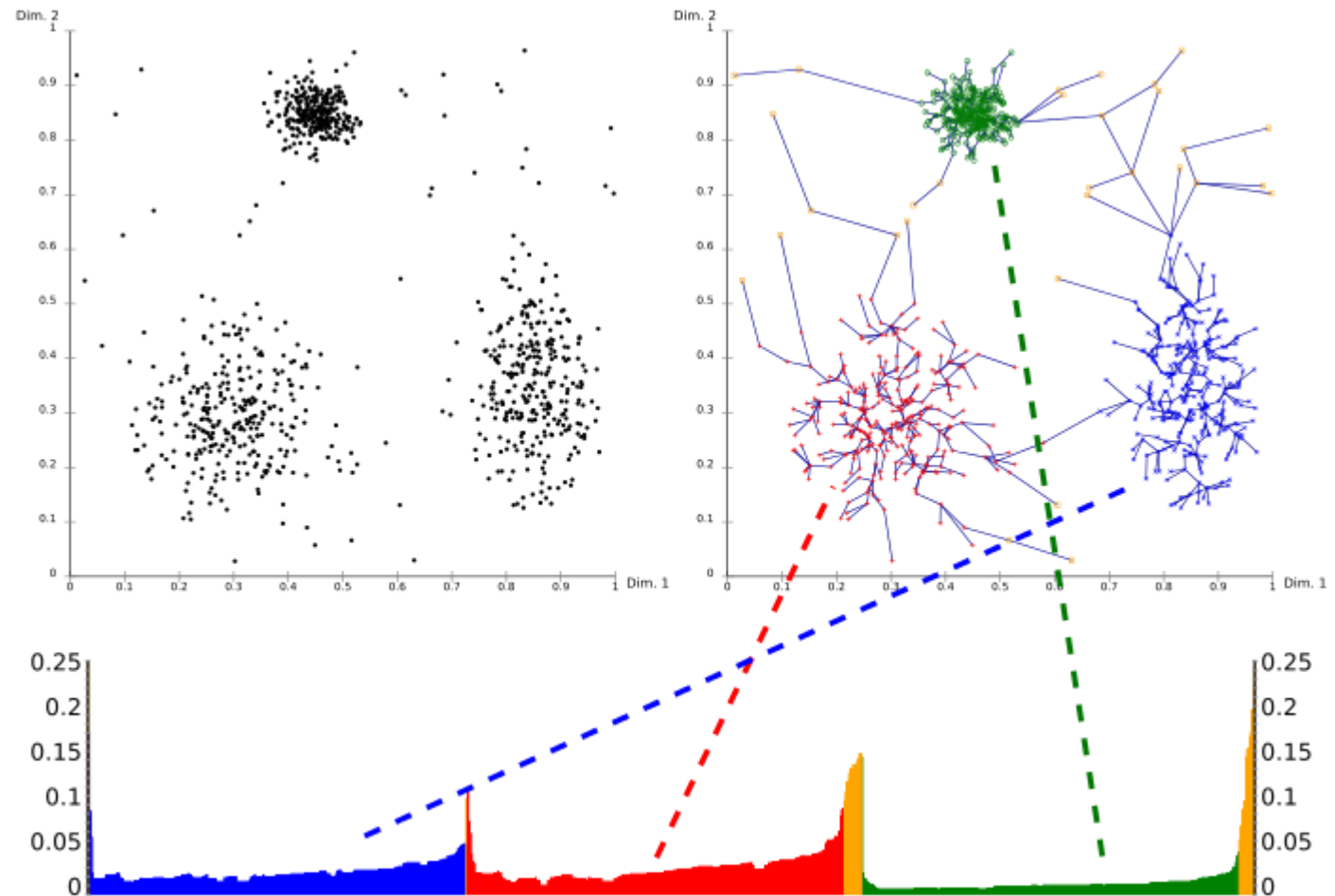
- Comparte los mismos conceptos que DBSCAN, pero a cada punto se le asignan nuevas distancias que lo caracterizan:
 - $Core-dist_{(\epsilon, minPts)}(p)$ = distancia al $minPts$ -ésimo punto más cercano dentro de $N_{\epsilon}(p)$
Sólo definida si p es punto núcleo.
 - $Reachability-dist_{(\epsilon, minPts)}(o, p) = \max(Core-dist_{(\epsilon, minPts)}(p), dist(p, o))$.
Sólo definida si p es un punto núcleo.
- El algoritmo propone un ordenamiento de la base de datos, a modo de poder calcular bajo dicha indexación la distancia de alcance de cada punto.
- Las distancias de alcance funcionan como una especie de dendograma, y permitirá establecer la cantidad de clusters a generar.
- A diferencia de DBSCAN, el parámetro **eps** no es requerido (gracias a la inclusión de la distancia de alcance), pero se recomienda su uso para efectos de costo computacional.



OPTICS

Ordering Points To Identify Clustering Structure

- ▶ El gráfico de las distancias de alcance (reachability plot), contiene los puntos bajos el ordenamiento propuesto por el algoritmo en el eje-x , mientras que en el eje-y la distancia de alcance respectiva.
- ▶ Puntos que pertenezcan al mismo cluster, tendrán una menor distancia de alcance a sus vecinos cercanos.
- ▶ Los valles representan los clusters.
- ▶ Mientras más pronunciado es el valle, mayor densidad del cluster



OPTICS

Ordering Points To Identify Clustering Structure

Trabajo original

- <https://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=C75424AECC04C36AC911CCCC7DB41238?doi=10.1.1.129.6542&rep=rep1&type=pdf>

Extensión OPTICS-OF para la detección de outliers

- <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.46.6586&rep=rep1&type=pdf>

Manual de referencia package dbscan

- <https://cran.r-project.org/web/packages/dbscan/dbscan.pdf>





ESCUELA DE INGENIERÍA
FACULTAD DE INGENIERÍA

EDUCACIÓN
PROFESIONAL

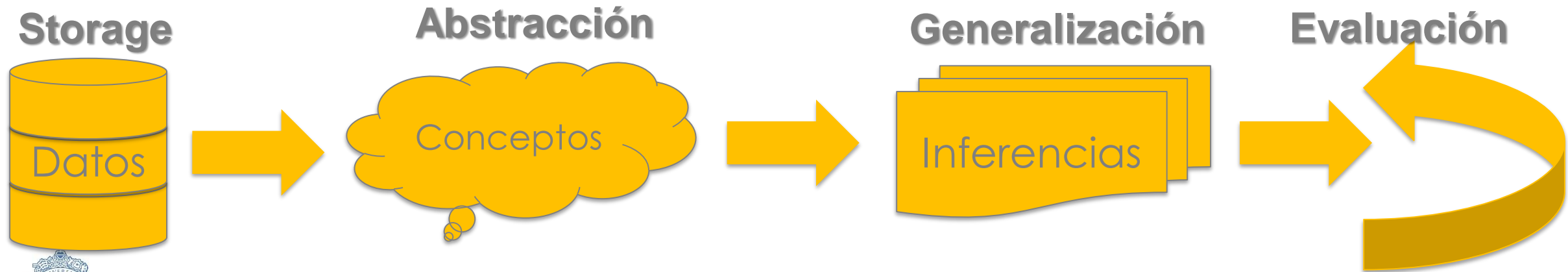
INTRODUCCIÓN MACHINE LEARNING

Modelos supervisados y regresión

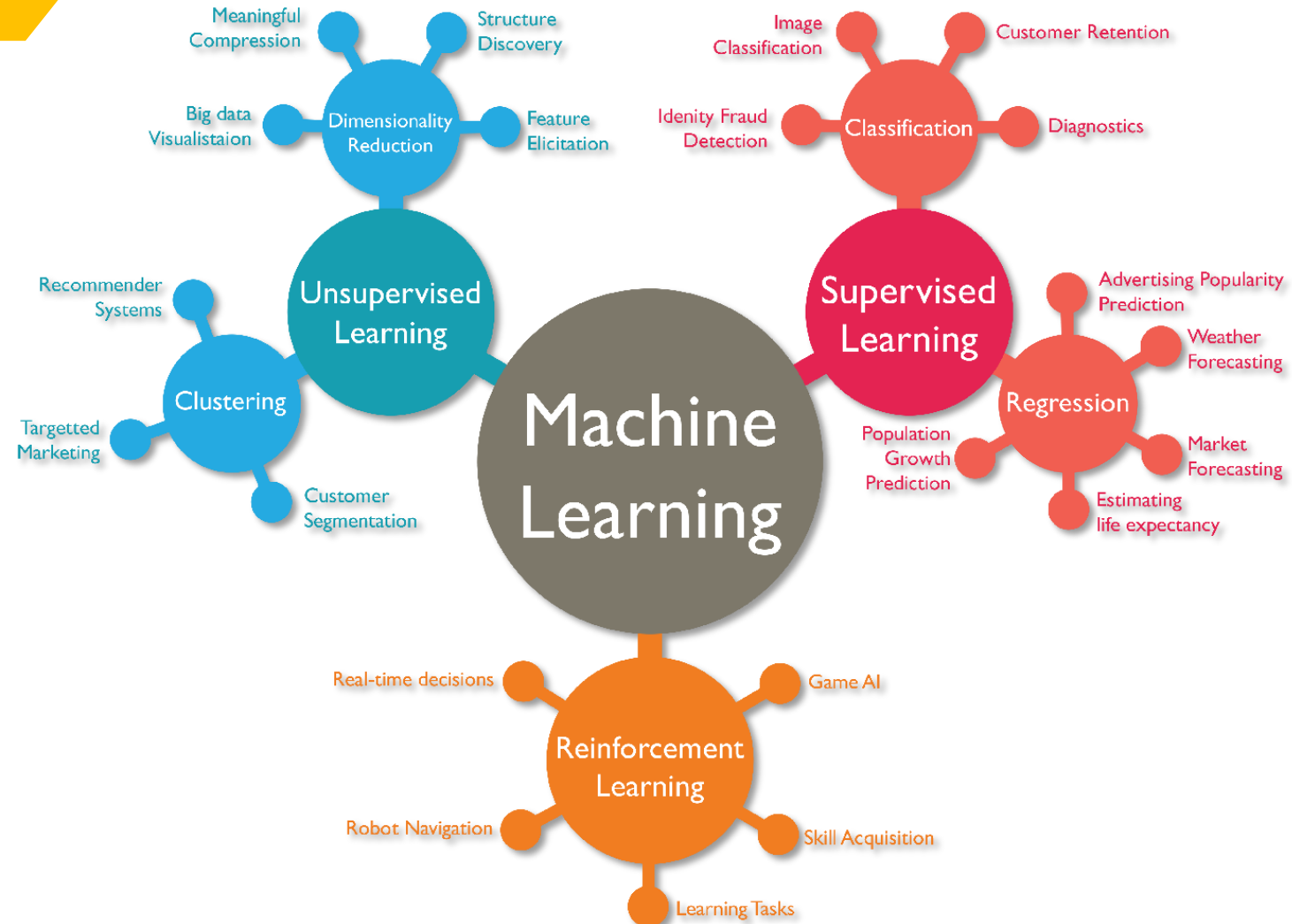
Machine Learning

“A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T , as measured by P , improves with experience E .”

Tom M.
Mitchell (1997)

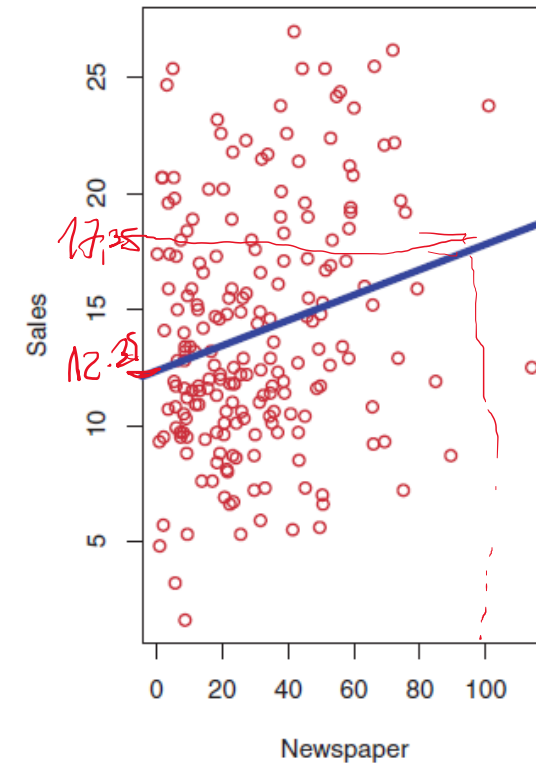
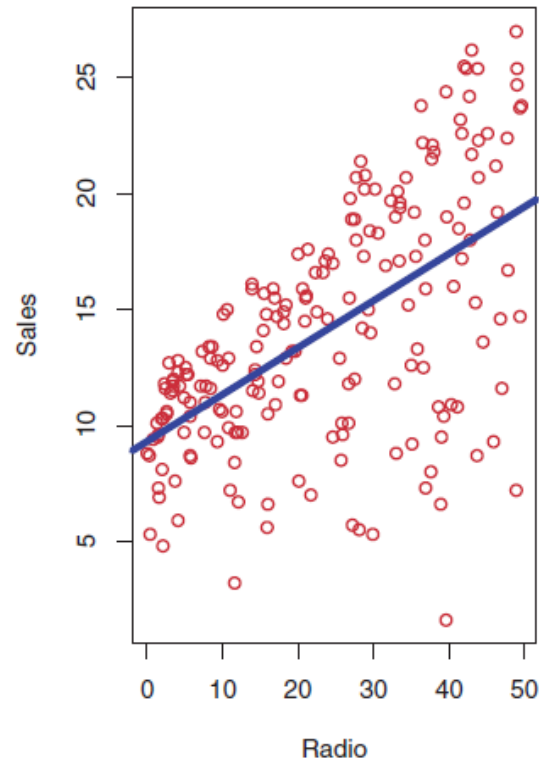
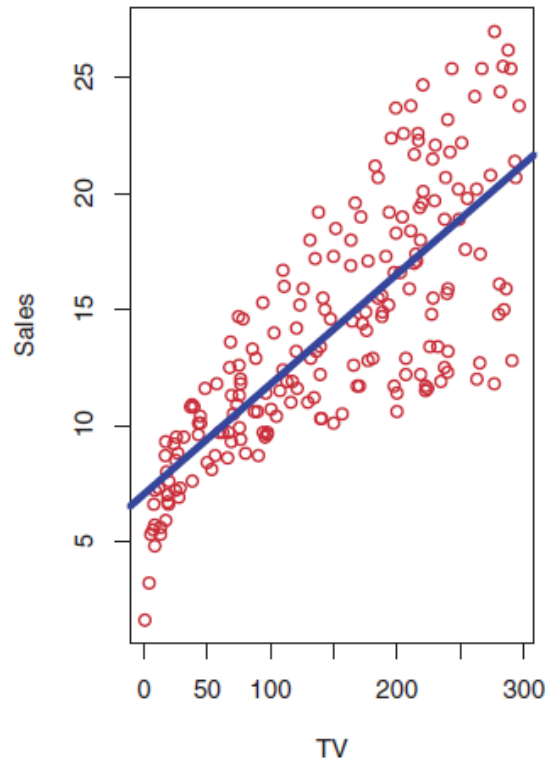


Machine Learning



Regresión lineal simple

Supongamos que interesa predecir el nivel de ventas de un determinado producto en función de los montos invertidos en distintos medios de publicidad (tv, radio, periódico).



$$y = 12.35 + 0.05 \cdot x$$



Regresión lineal simple

- En general, el problema podría expresarse matemáticamente de la siguiente manera:

$$Y = f(X) + \epsilon \quad (1)$$

- Donde X contiene a las variables explicativas (monto en tv, radio y periódico en el ejemplo), y ϵ es un error aleatorio no observable.
- En esta especificación, f es una función desconocida y que buscamos estimar.
- En términos simples, diremos que un modelo es de regresión, cuando en la expresión (1), la variable de interés a predecir, Y, es una variable numérica.



Regresión lineal simple

- Cuando el modelo f a estimar, se asume como una función lineal, diremos que (1) es un modelo de regresión lineal. En tal caso, el modelo matemático queda expresado de la siguiente manera:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon \quad (1)$$

- Donde $\beta_i, i = 0, \dots, p$ son los parámetros a estimar (estos parámetros definen al modelo), y ϵ es un error aleatorio no observable, típicamente siguiendo una distribución aleatoria normal $N(0, \sigma^2)$.



Regresión lineal simple

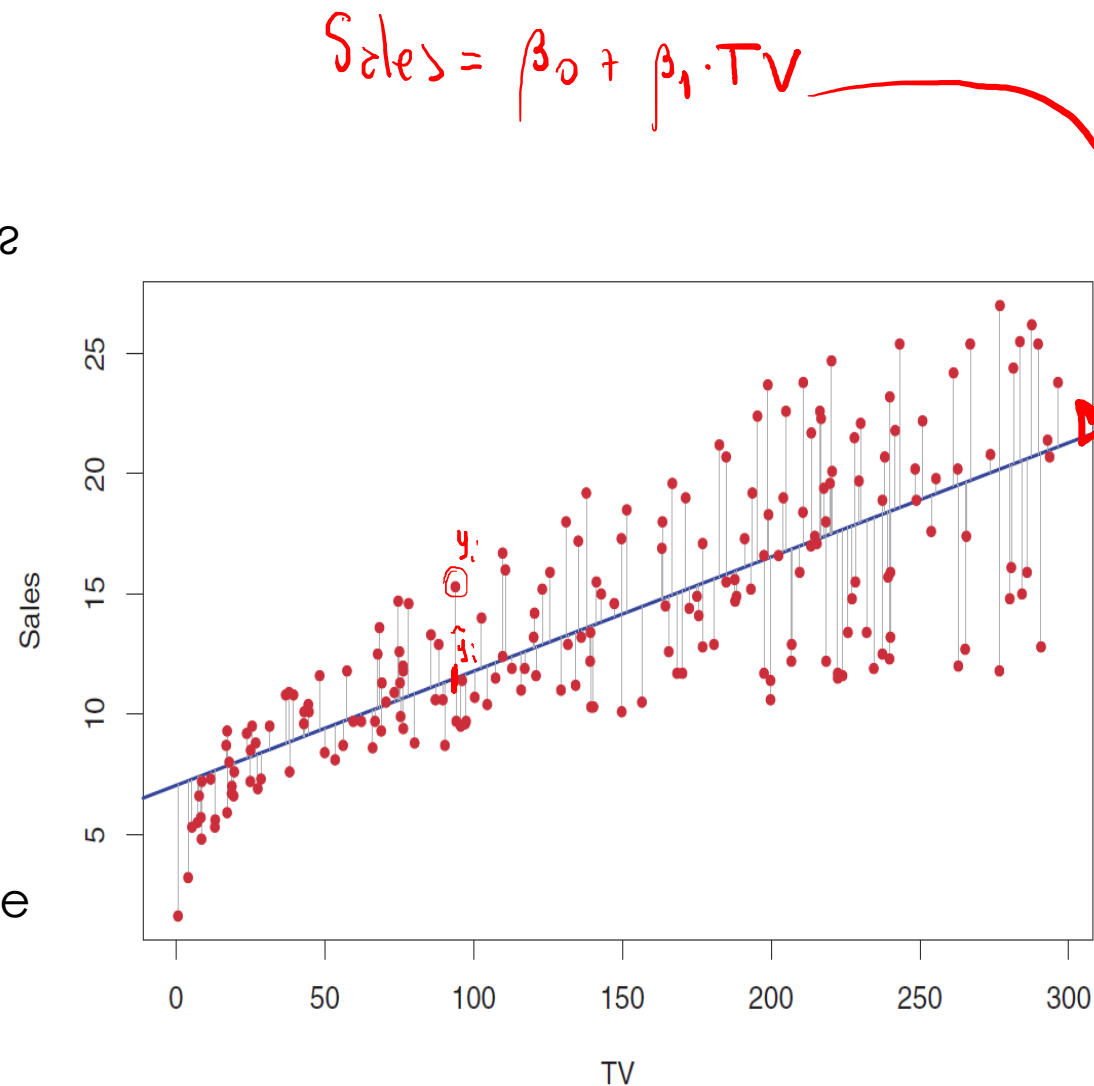
¿Cómo se estiman los parámetros en una regression lineal?

- Tanto en R como en la mayoría de los softwares, la manera estándar de estimar los coeficientes en un modelo de regresión, es mediante la estimación vía mínimos cuadrados, donde se busca minimizar la suma residual:

$$\sum_{i=1}^n (\hat{y}_i - y_i)^2$$

$$, \hat{y}_i = f(x_i)$$

- No entraremos en detalle, respecto de las bondades de esta estimación y que coincide con otros estimadores en el caso de la regresión lineal con errores normales.



Regresión lineal simple

- Sin entrar en más detalles técnicos, veamos como podemos ajustar una regresión lineal en R.
- Para ello podemos utilizar la función `lm()`, del paquete base. Esta recibe como argumento principal una formula y un datasets, del siguiente modo.

```
lm(formula = y ~ x1+x2+...+xp, data = dataset)
```

- Generemos nuestra primera regresión lineal con el dataset "Advertising". El cual contiene las ventas totales de un producto y los montos invertidos en tres tipos de publicidad (tv, radio, periódico). En esta primera iteración consideremos solamente la variable newspaper

```
lm(formula = sales ~ newspaper, data = Advertising)
```

- Hablaremos sobre los coeficientes estimados y la salida que genera R en el notebook



Regresión lineal simple

¿Cómo interpretamos los parámetros de una regresión lineal (simple)

$$\text{Sup. } Y = \beta_0 + \beta_1 X + \varepsilon \rightarrow \hat{Y}(X) = \beta_0 + \beta_1 X \quad (1)$$

¿Cómo varía Y , si X aumenta 1 unidad?

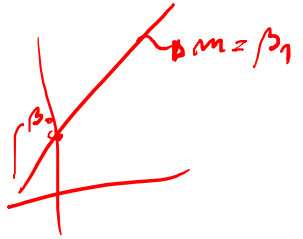
$$(X+1) \quad \hat{Y}(X+1) = \beta_0 + \beta_1 \cdot (X+1) \quad (2)$$

$$\hat{Y}(X) = \beta_0 + \beta_1 X$$

$$\hat{Y}(X+1) = \beta_0 + \beta_1 (X+1)$$

$$\Delta \hat{Y} = \beta_1$$

$\therefore \beta_1$ representa la variación en Y cuando X aumenta 1 unidad.





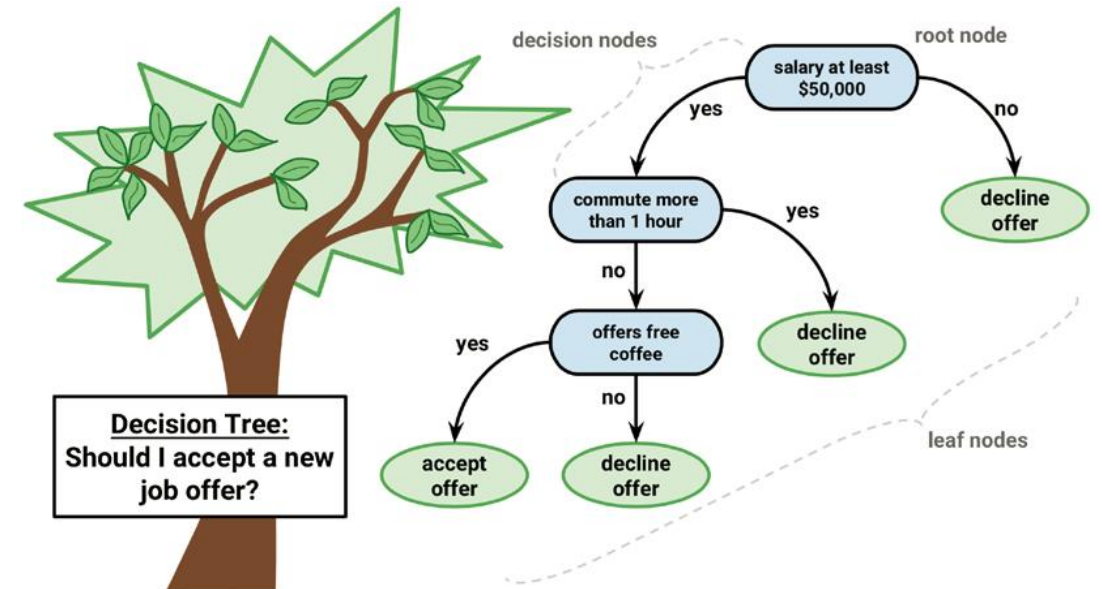
ESCUELA DE INGENIERÍA
FACULTAD DE INGENIERÍA

EDUCACIÓN
PROFESIONAL

MODELOS DE CLASIFICACIÓN

Árboles de clasificación

- Imaginemos que tenemos una serie de sentencias lógicas que nos permiten representar la data
- A través de una serie de sentencias del tipo "If" - "Then".
- Cada "if" divide la data en dos categorías.
- "then" asigna un valor.
- Cuando los "if" están anidados, la estructura se denomina árbol.



Árboles de clasificación

Preguntas interesantes:

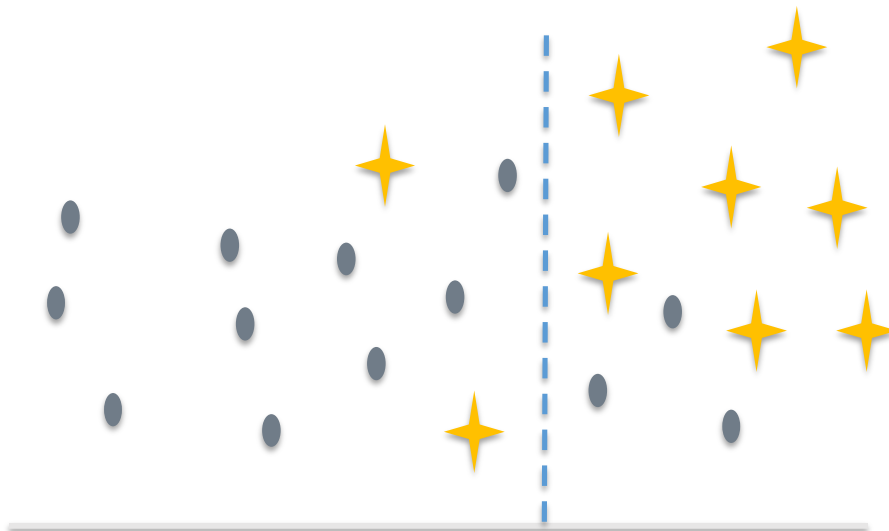
- ¿Como decidir el orden de las variables en que se dividen las ramas?
- ¿Como se asigna la clase a la que pertenecen las observaciones de un nodo?
- ¿Como decidir hasta donde hacer crecer un árbol?
- ¿Como evaluar el desempeño de este modelo?
- ¿Como determinar la precisión si se quiere realizar predicciones?



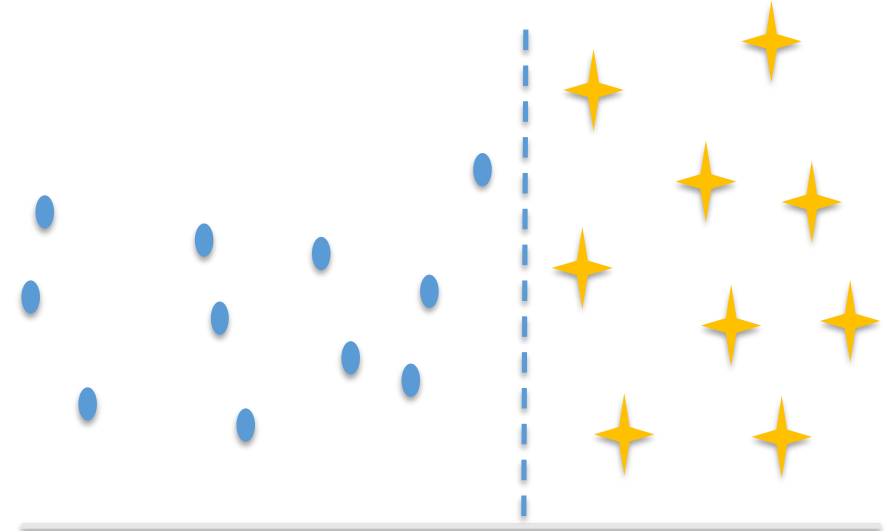
Árboles de clasificación

¿Como decidir el orden de las variables en que se dividen las ramas?

Supongamos que queremos decidir si dividir por V1 o por V2. ¿Qué atributo nos aporta mayor información?



V1



V2



Árboles de clasificación

Una manera de medir el nivel de impureza (o "desorden"), dentro de un nodo es la entropía, definida como:

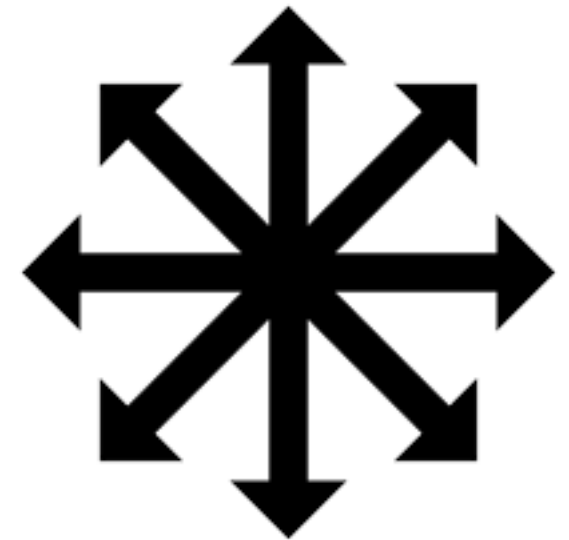
$$H(Y) = - \sum_y P(Y = y) \cdot \log P(Y = y) \quad (1)$$

Se define la entropía condicional a X_i como:

$$\begin{aligned} H(Y|X_i) &:= \sum_x P(X_i = x) H(Y|X_i = x) \\ &= - \sum_x P(X_i = x) \sum_y P(Y = y|X_i = x) \log P(Y = y|X_i = x) \end{aligned}$$

Así, la ganancia de información (o información mutua) es la diferencia entre las dos:

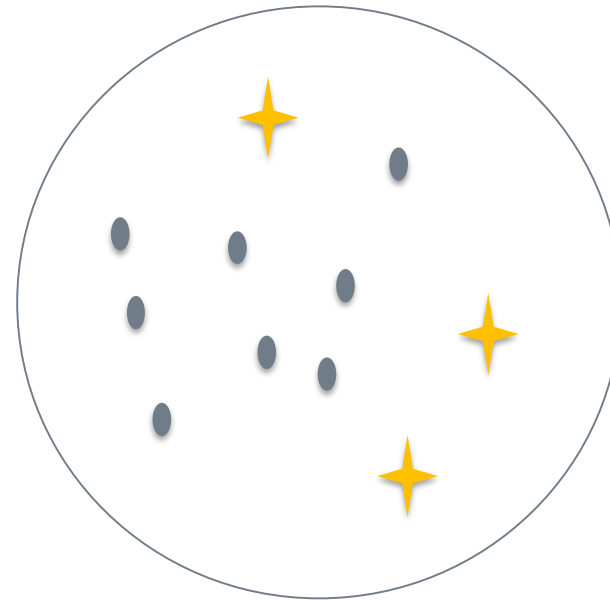
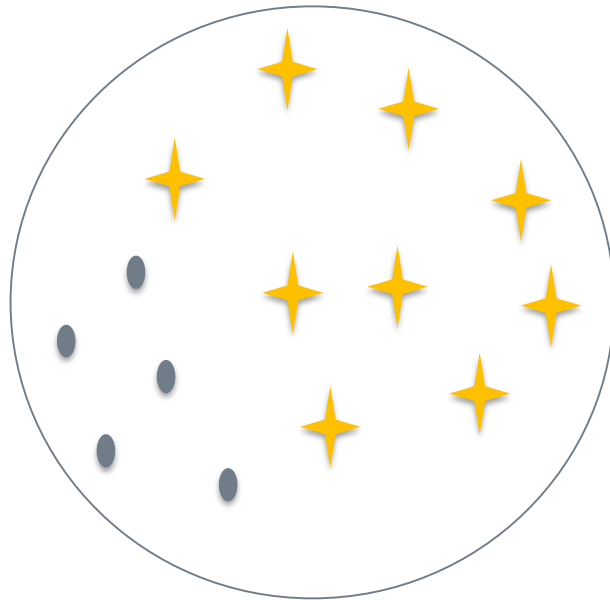
$$I(Y, X_i) = H(Y) - H(Y|X_i)$$



Árboles de clasificación

¿Como se asigna la clase a la que pertenecen las observaciones de un nodo terminal?

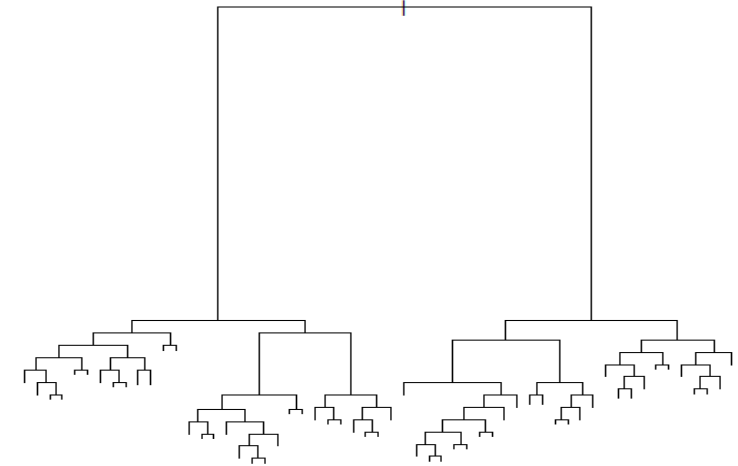
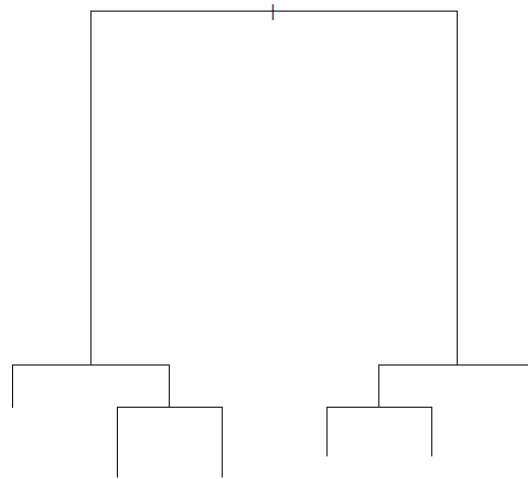
- Clasificación: Mayoría de votos.
- Regresión: Promedio



Árboles de clasificación



- ¿Como decidir hasta donde hacer crecer un árbol?
- ¿Como evaluar el desempeño de este modelo?
- ¿Como determinar la precisión si se quiere realizar predicciones?





ESCUELA DE INGENIERÍA
FACULTAD DE INGENIERÍA

EDUCACIÓN
PROFESIONAL

EVALUACIÓN DE MODELOS DE CLASIFICACIÓN

¿Por qué es importante validar el modelo en nuevos datos que no hayan formado parte del set de entrenamiento?

[Ref.regression](#)



Flexibilidad y performance del modelo

Retomemos por un momento el contexto de regresión

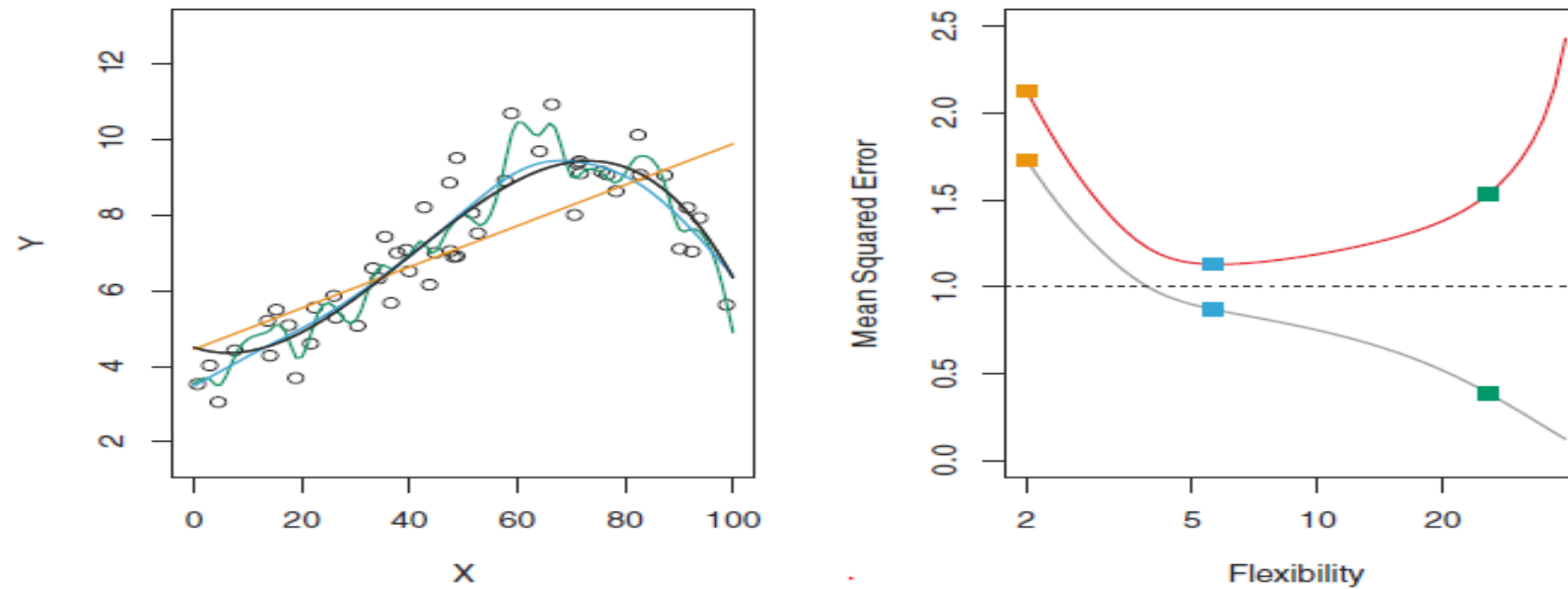


FIGURE 2.9. Left: Data simulated from f , shown in black. Three estimates of f are shown: the linear regression line (orange curve), and two smoothing spline fits (blue and green curves). Right: Training MSE (grey curve), test MSE (red curve), and minimum possible test MSE over all methods (dashed line). Squares represent the training and test MSEs for the three fits shown in the left-hand panel.



Error de Test

Asumiendo un punto de test fijo:

$$E \left(y_0 - \hat{f}(x_0) \right)^2 = \text{Var}(\hat{f}(x_0)) + [\text{Bias}(\hat{f}(x_0))]^2 + \text{Var}(\epsilon).$$

- ¿Cómo evaluamos el desempeño de un modelo de regresión?
- ¿Cómo estimamos entonces el error de test?
- ¿Y si el contexto es de clasificación?



Métricas de desempeño Clasificación

- ¿Qué tan bien clasifica nuestro árbol?

$$\begin{aligned} \textit{Accu} &:= \text{Tasa de clasificación correcta} \\ &= \frac{\text{Clasificaciones correctas}}{\text{Total de observaciones a predecir}} \end{aligned}$$

En general se utiliza la *Tasa de error*:

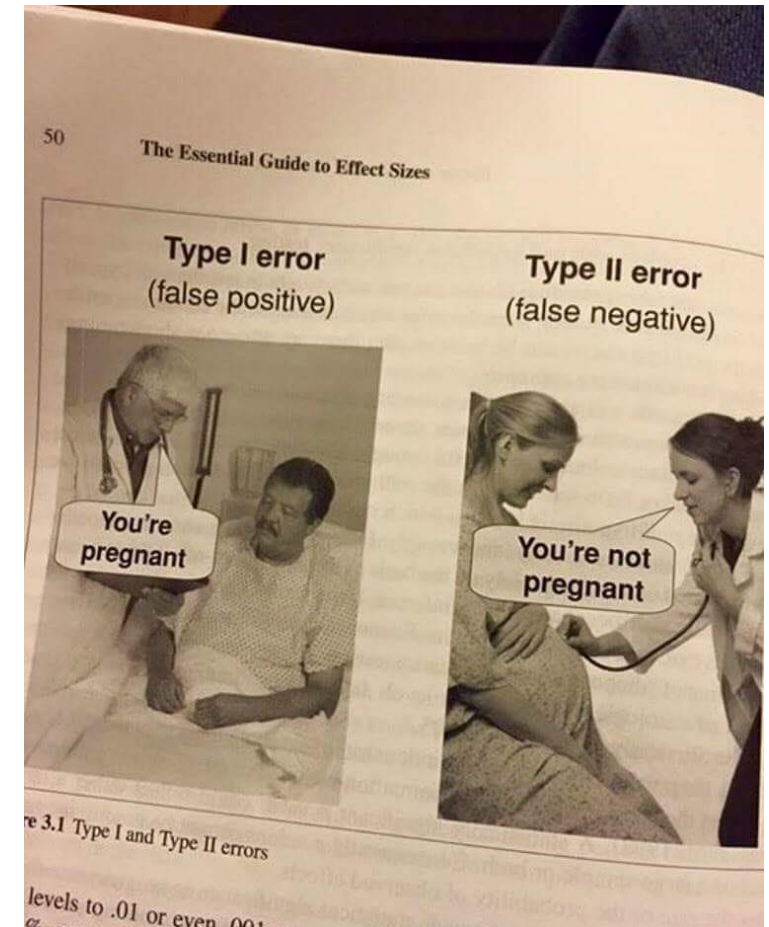
$$\textit{Err} = 1 - \textit{Accu}$$

En nuestro ejemplo...



Error de predicción

- Cuando la clasificación es binaria, existen otros indicadores que nos ayudarán a determinar el desempeño de un modelo de clasificación.
- Supongamos que nuestra variable dependiente Y toma los estados **Positivo (P)** y **Negativo (N)**.
- Bien sabemos que se pueden cometer dos tipos de errores:



Error de predicción



TP = "True positive"

FP = "False positive"

TN = "True negative"

FN = "False negative"

Nuestra medida de exactitud queda entonces como:

$$Accu = \frac{TP + TN}{TP + FP + FN + TN}$$



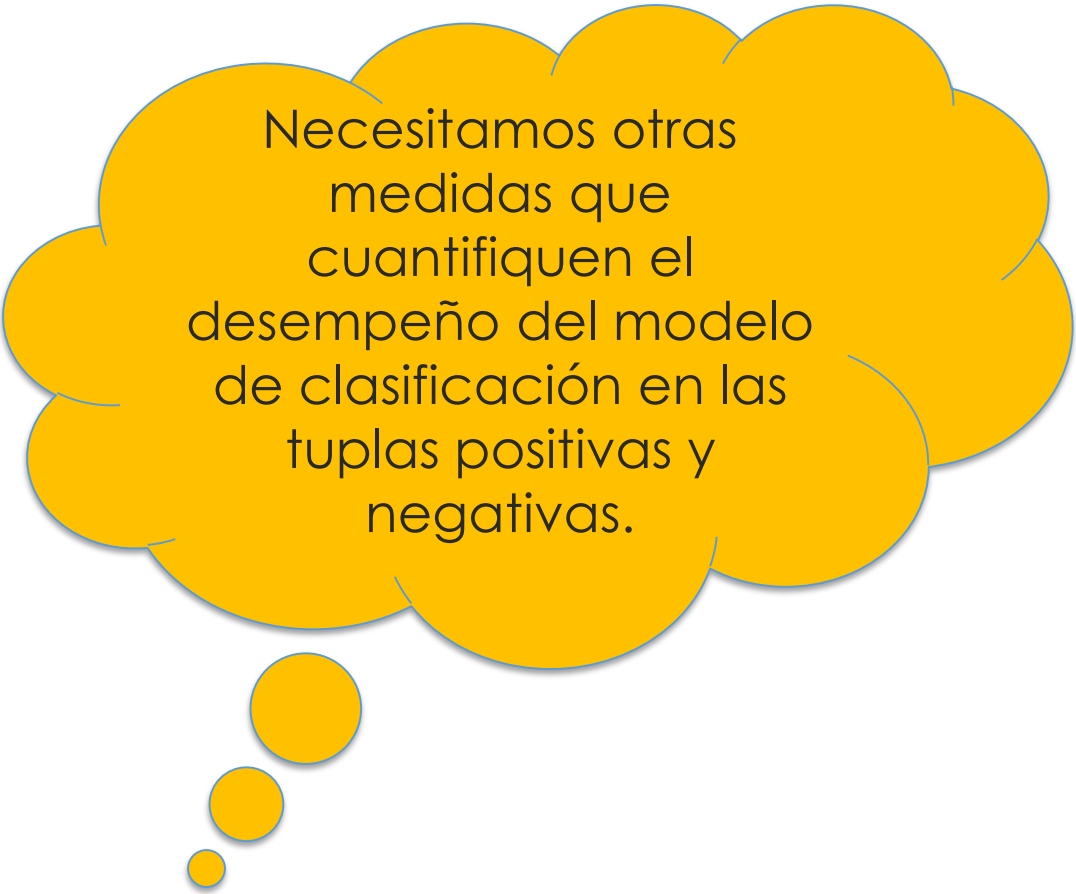
Matriz de confusión

		Clase Predicha		
		Si	No	Total
Clase Real	Si	TP	FN	P
	No	FP	TN	N
	Total	P'	N'	P+N



Error de predicción

- ¿Qué ocurre si la clase de interés es rara o poco frecuente?
- Ejemplo: detección de fraude, cáncer, fuga de clientes, etc
- Una tasa de reconocimiento del 97% (tasa de error de un 3%) puede ser muy alta, pero engañosa si sólo el 3% de las tuplas tenían cáncer.
- Puede ocurrir que el clasificador sea muy bueno reconociendo los registros libres de cáncer, pero incapaz de detectar las que sí lo padecen.



Necesitamos otras medidas que cuantifiquen el desempeño del modelo de clasificación en las tuplas positivas y negativas.



Ejemplo extremo



Otras métricas

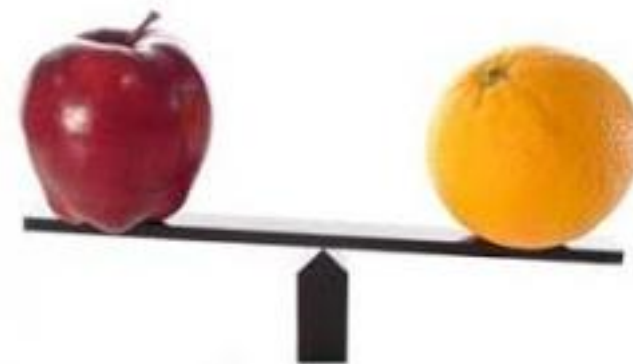
- **Recall** (true positive rate o *sensibilidad*)
 $TPR = TP/P = TP/(TP + FN)$
- **True negative rate** (*Especificidad*)
 $TNR = TN/N = TN/(TN + FP)$
- **Precision**
 $PREC = TP/(TP + FP)$
- **False Positive rate** $FPR = FP/N = FP/(FP + TN) = 1 - TNR$



Comparación de dos clasificadores

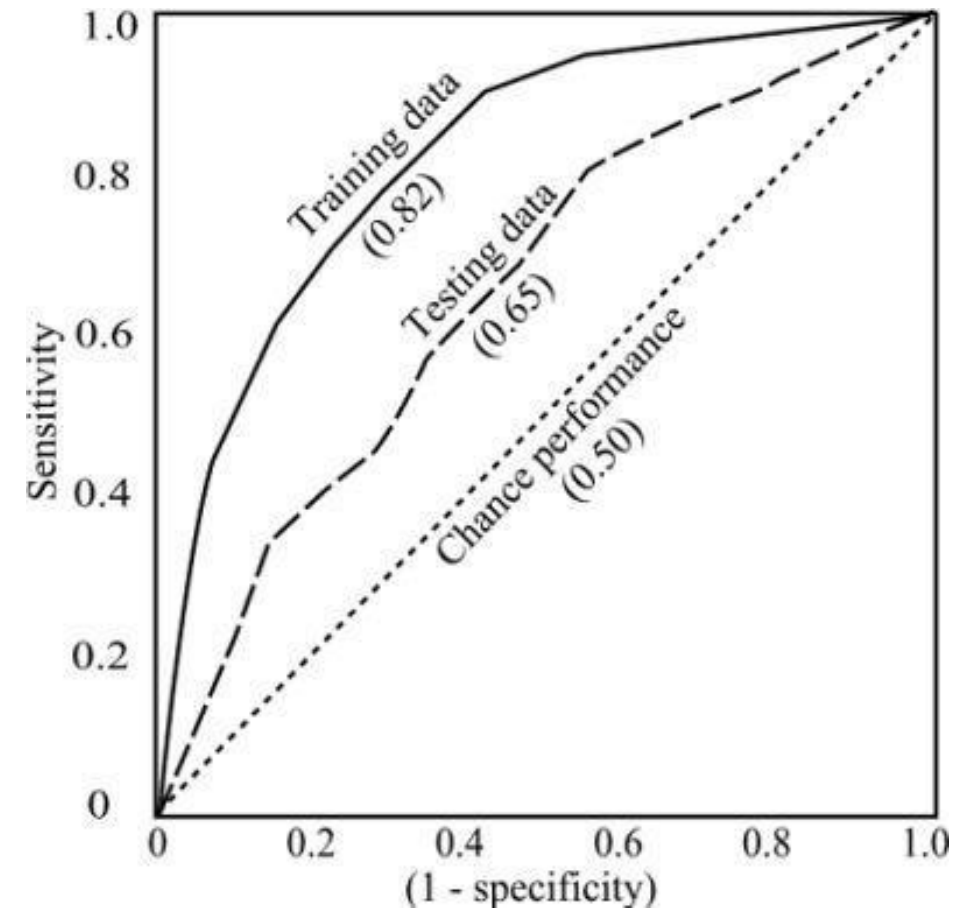
Cómo comparar dos clasificadores

- TP, TN, FP, FN también son útiles para cuantificar los costos y beneficios asociados con un modelo de clasificación.
- Los costos de un FN (predecir que está sano un paciente con cáncer) es mucho mayor que el de un FP (predecir que está con cáncer alguien sano).
- Se puede pesar más uno tipo de error que el otro, asignándole un costo más alto.
- Estos costos consideran el riesgo de la mala clasificación, costos económicos, etc.



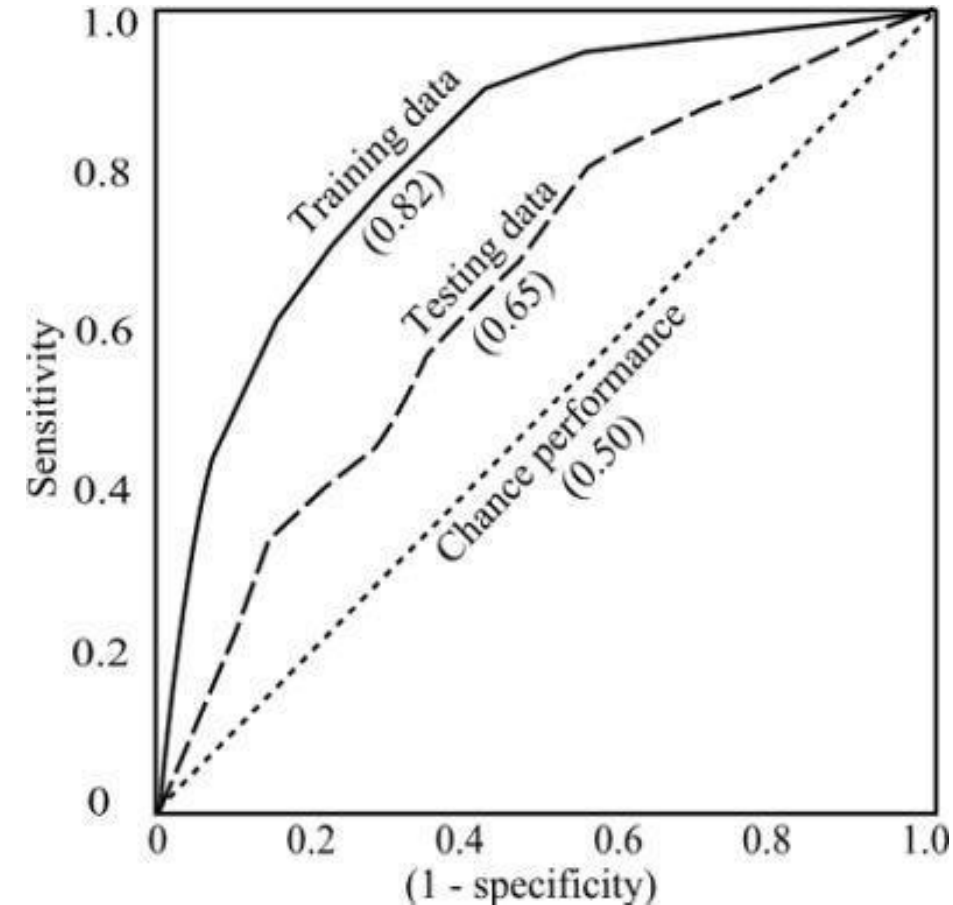
Curvas de ROC

- Para un problema con 2 clases, la curva ROC permite visualizar el compromiso entre la tasa en que el modelo puede reconocer con precisión los casos positivos versus la tasa de falsos positivos para diferentes porciones del conjunto donde se está validando el modelo.
- Eje Y: Tasa de verdaderos positivos (TPR o sensibilidad)
- Eje X: Tasa de falsos positivos (FPR o 1 - Especificidad)



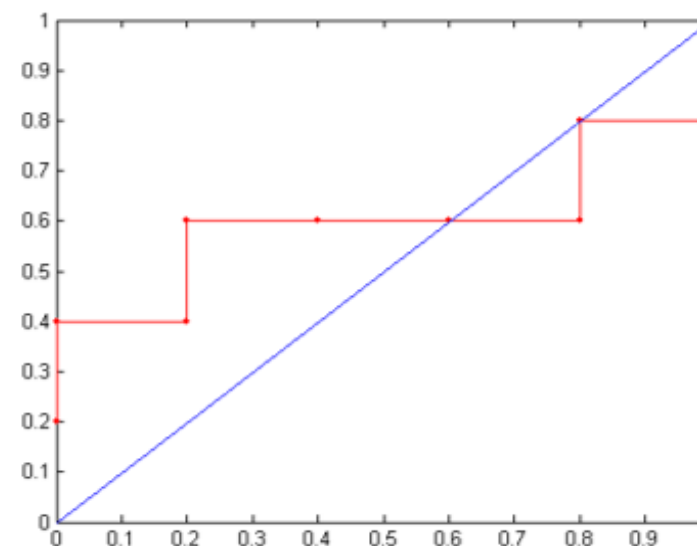
Curvas de ROC

- Permiten comparar visualmente distintos modelos de clasificación.
 - El área que queda bajo la curva es una medida de la precisión del clasificador (AUC)
 - Más cerca de la diagonal (área = 0.5), menos preciso será el modelo
 - Por tanto, un modelo perfecto tendrá área = 1.



Curvas de ROC

Ejemplo	P(+E)	Clase
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+



Clase	+	-	+	-	-	-	+	-	+	+	
Probabilidad	0.25	0.43	0.53	0.76	0.85	0.85	0.85	0.87	0.93	0.95	1.00
TP	5	4	4	3	3	3	3	2	2	1	0
FP	5	5	4	4	3	2	1	1	0	0	0
TN	0	0	1	1	2	3	4	4	5	5	5
FN	0	1	1	2	2	2	2	3	3	4	5
TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0



Curvas de ROC



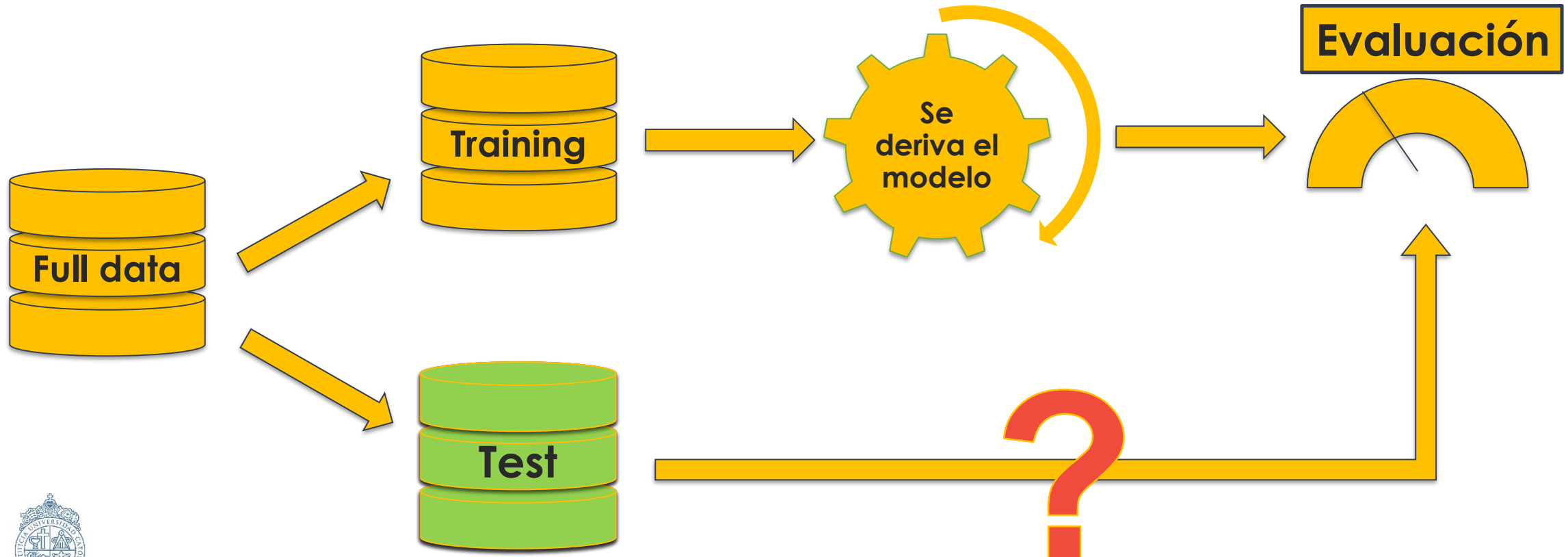


ESCUELA DE INGENIERÍA
FACULTAD DE INGENIERÍA

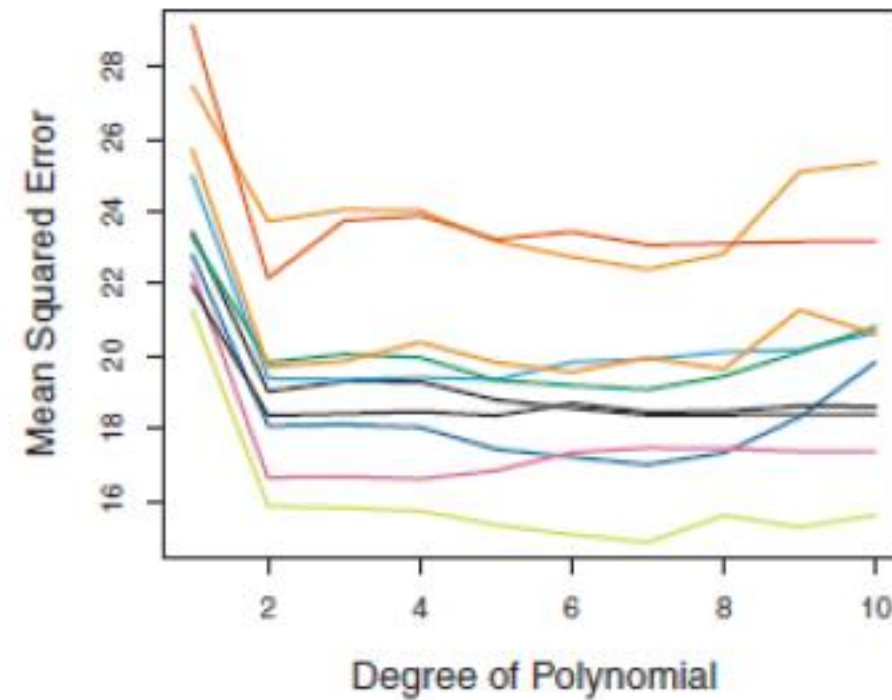
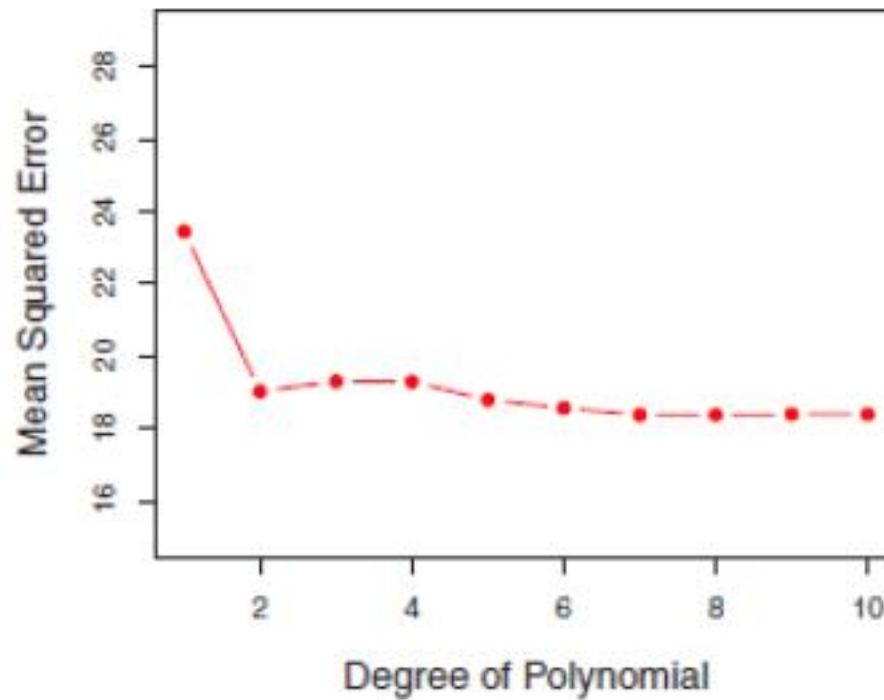
EDUCACIÓN
PROFESIONAL

EVALUACIÓN DE MODELOS DE CLASIFICACIÓN PARTE II

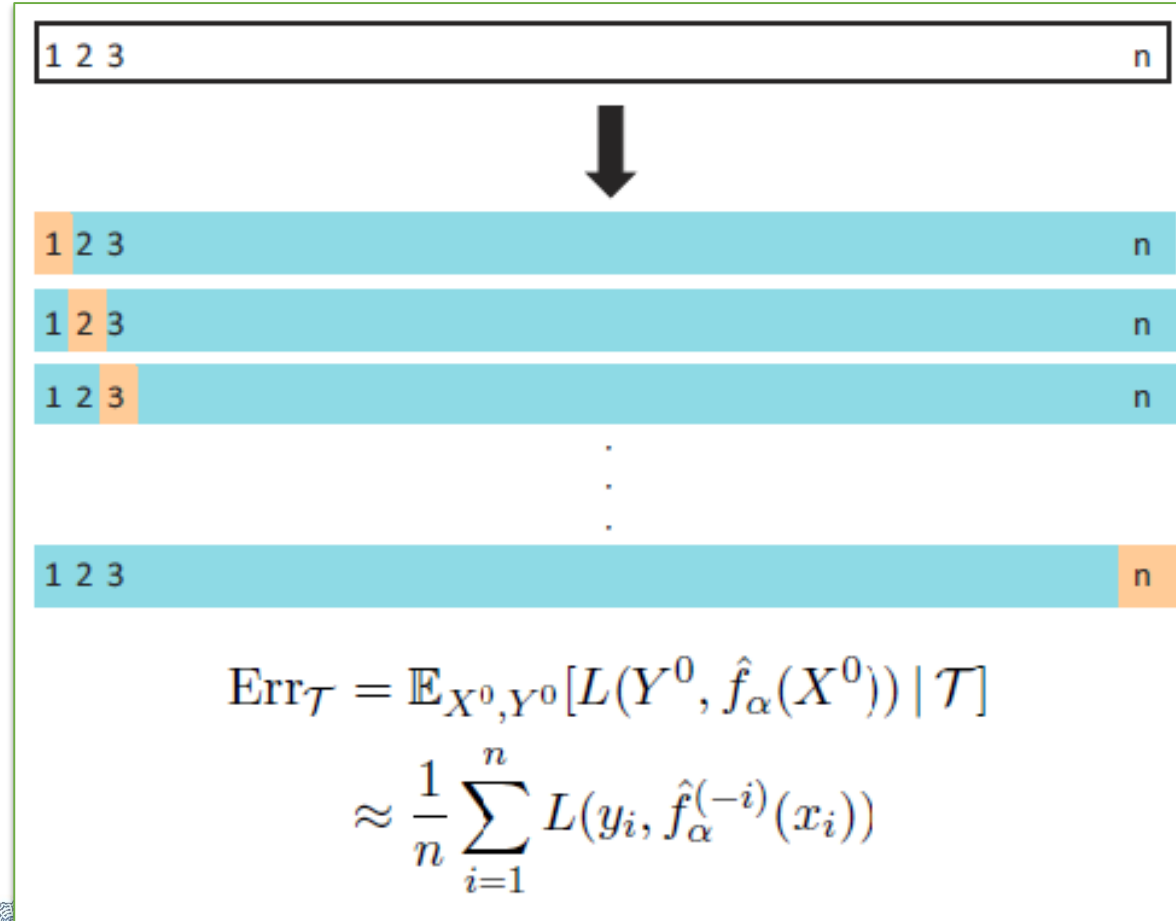
Hasta ahora hemos seguido este enfoque, pero...



Auto data set. izquierda: tasa de error para una sola data de test. Derecha: tasa de error repetida 10 veces (diferentes divisiones aleatorias)



Validación cruzada LOOCV

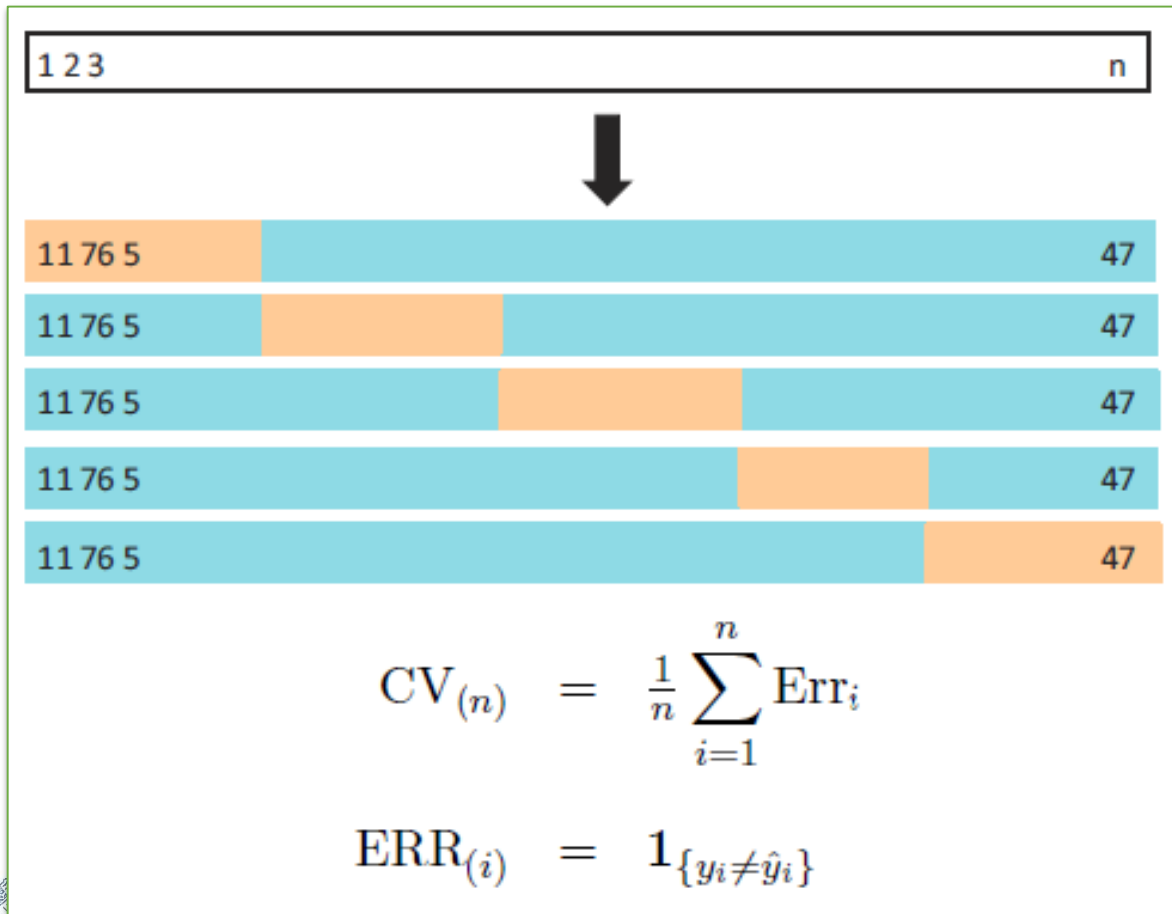


Intuitivamente

- ▶ ¿Qué inconveniente es evidente?



Validación cruzada K-FOLD

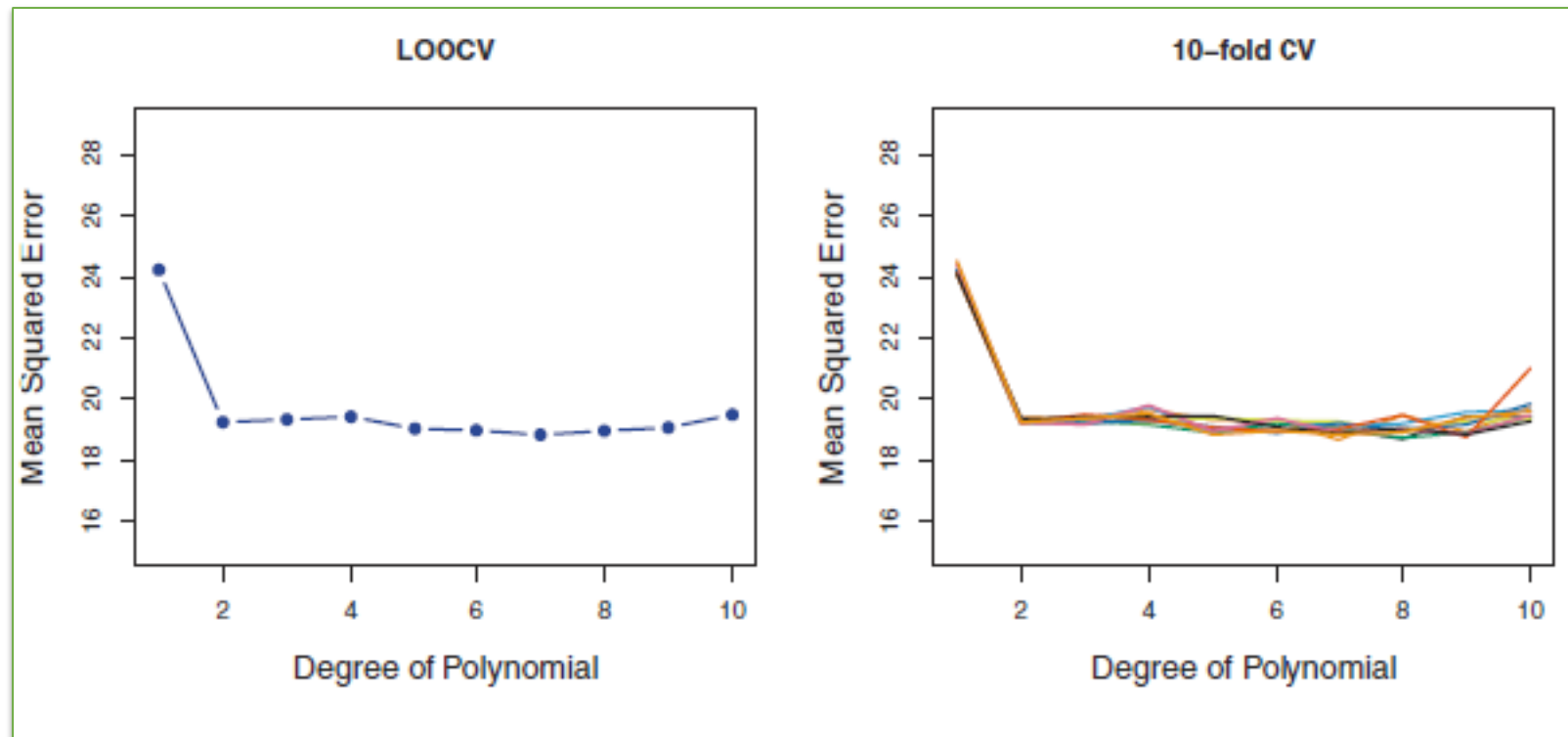


Similar a LOOCV pero de menor costo computacional



Validación cruzada K-FOLD

Se puede demostrar que con $k=10$, se obtiene una estimación similar a la obtenida a través de LOOCV del error de test.





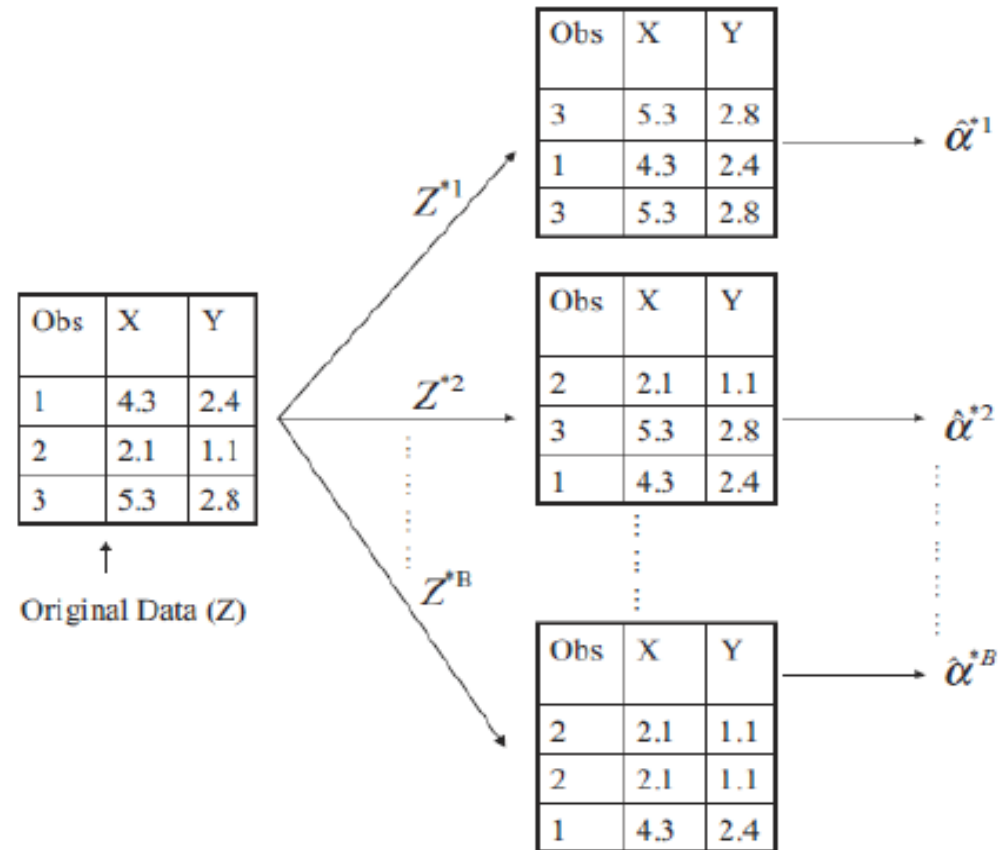
ESCUELA DE INGENIERÍA
FACULTAD DE INGENIERÍA

EDUCACIÓN
PROFESIONAL

MÉTODOS ENSAMBLADOS

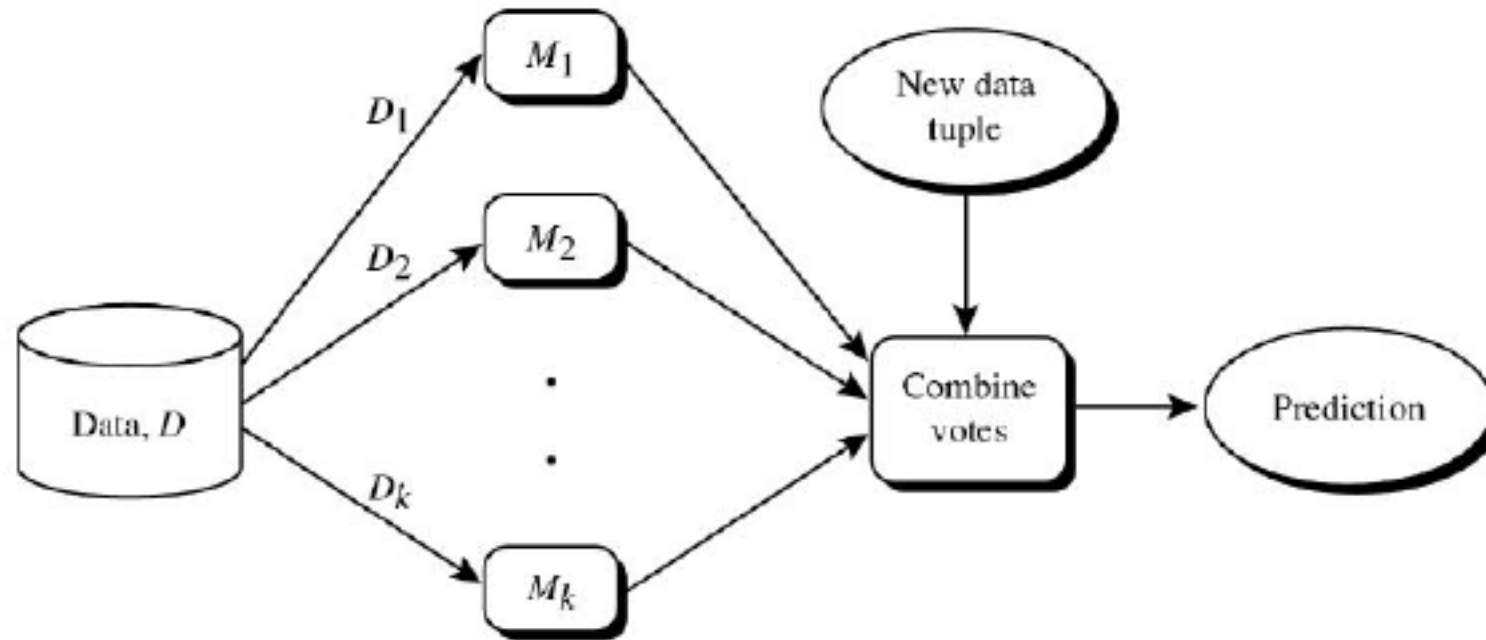
Bootstrap aggregation

- Intuición
...



Bootstrap aggregation

Con la idea de Bootstrap podemos generar métodos ensamblados de la siguiente manera

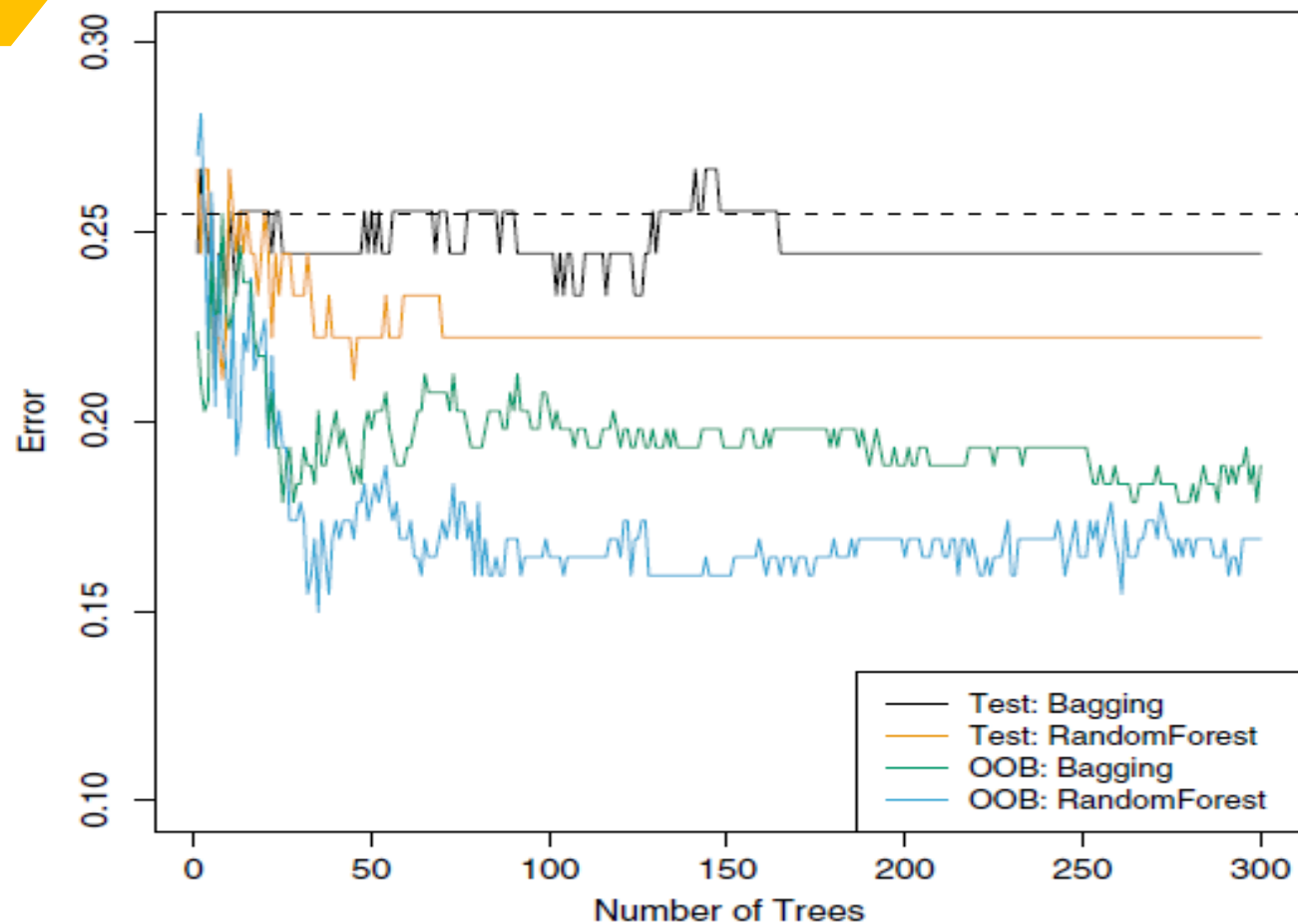


OOB error y LOOCV error

- Al generar muestras bootstrap y ensamblar un método de clasificación (o regresión) mediante Bagging, es posible estimar el error de test sin la necesidad de llevar a cabo validación.
- Aproximadamente un 63% de los datos serán parte de cada muestra bootstrap.
- Se puede predecir y promediar el error en el “tercio” restante y estimar el error.
- Se puede demostrar que para un alto número de muestras bootstrap OOB converge a LOOCV error.



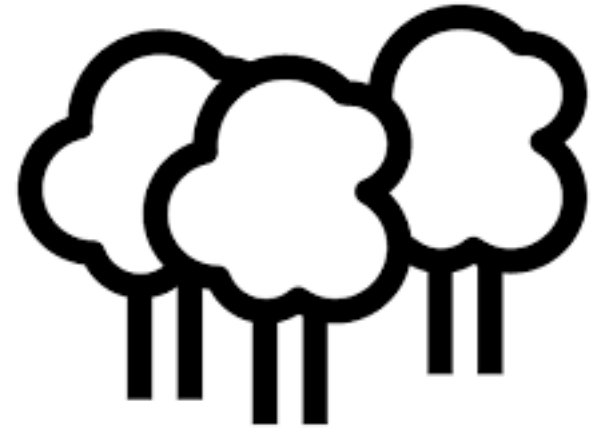
Bootstrap aggregation



Modelos usuales

Algunos de los algoritmos ensamblados más populares, son modelos basados en Árboles de decisión:

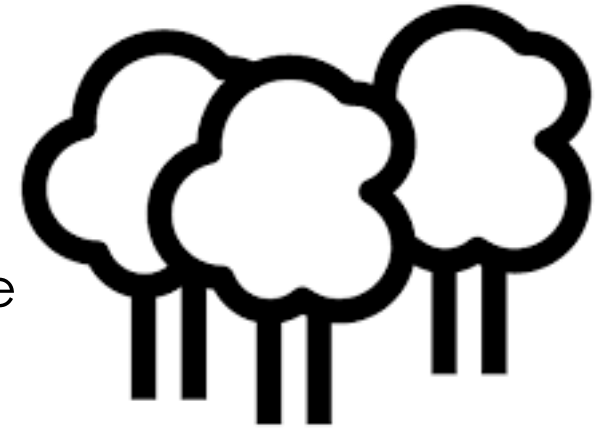
- Bagging
- Random Forest
- Adaboost



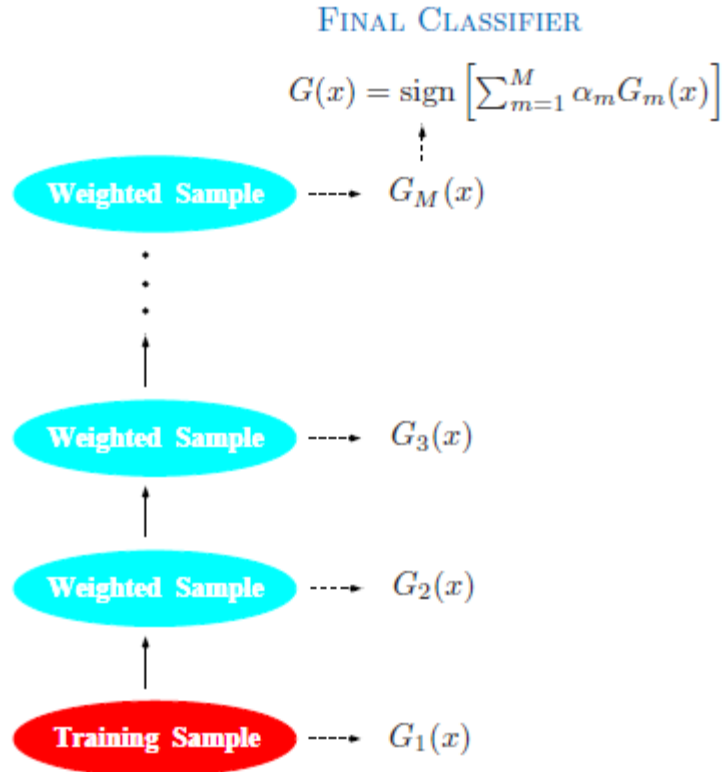
Random Forest

Un Random Forest es esencia un Bootstrap Aggregation, pero en cada iteración (generación de cada árbol) se tienen las siguientes consieraciones:

- Se escogen $k < p$ variables por las que realizar los splits.
- Estas k variables explicativas se escogen al azar.
- Esto impacata en la calidad predictiva del modelo, ya que se disminuye el nivel de “correlación” de las respuestas de cada modelo (iteración)



Adaboost



Algorithm 10.1 *AdaBoost.M1*.

1. Initialize the observation weights $w_i = 1/N$, $i = 1, 2, \dots, N$.
2. For $m = 1$ to M :
 - (a) Fit a classifier $G_m(x)$ to the training data using weights w_i .
 - (b) Compute

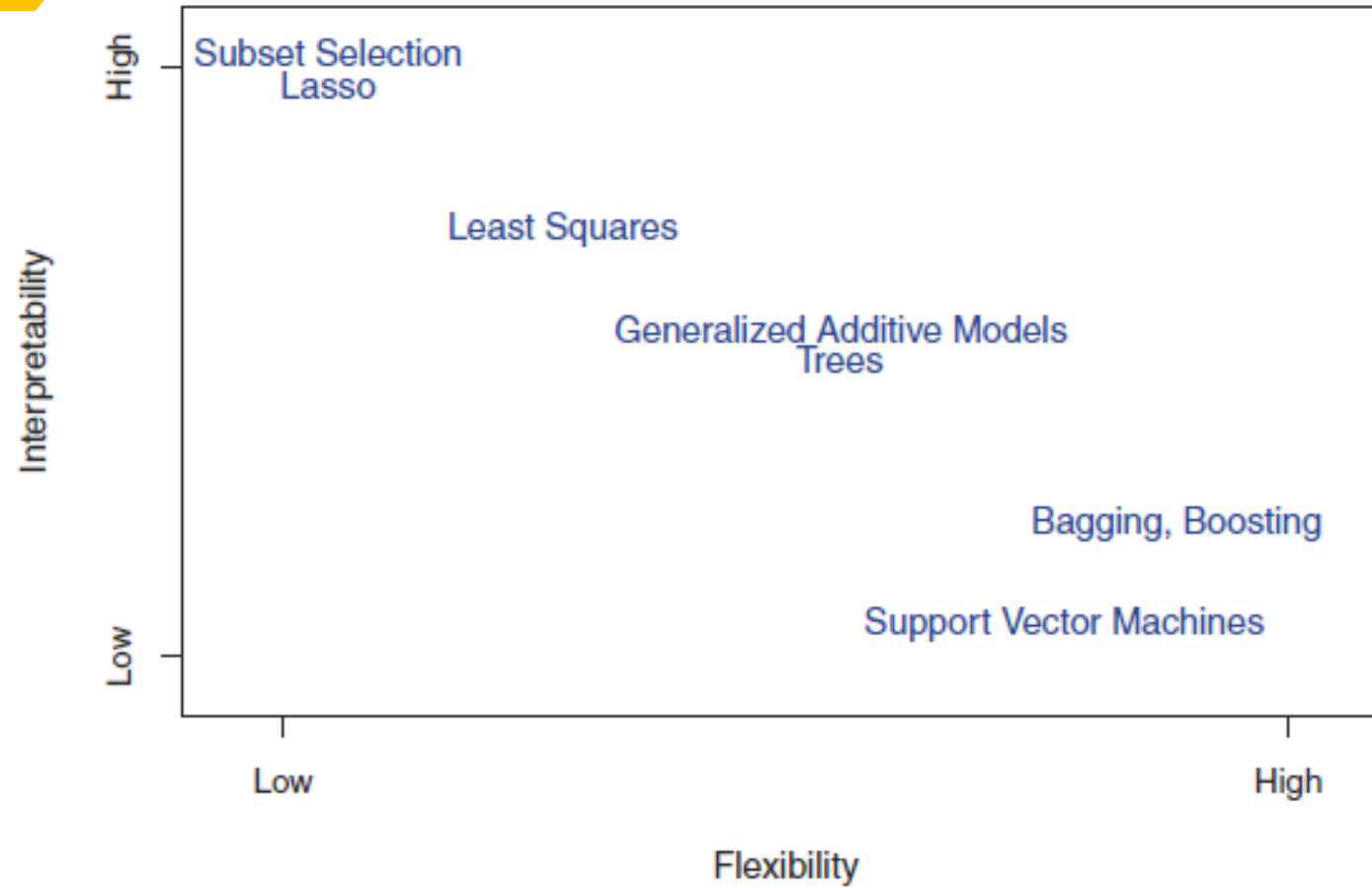
$$\text{err}_m = \frac{\sum_{i=1}^N w_i I(y_i \neq G_m(x_i))}{\sum_{i=1}^N w_i}.$$

- (c) Compute $\alpha_m = \log((1 - \text{err}_m)/\text{err}_m)$.
- (d) Set $w_i \leftarrow w_i \cdot \exp[\alpha_m \cdot I(y_i \neq G_m(x_i))]$, $i = 1, 2, \dots, N$.

3. Output $G(x) = \text{sign} \left[\sum_{m=1}^M \alpha_m G_m(x) \right]$.
-



Flexibilidad e interpretación



Implementación





ESCUELA DE INGENIERÍA
FACULTAD DE INGENIERÍA

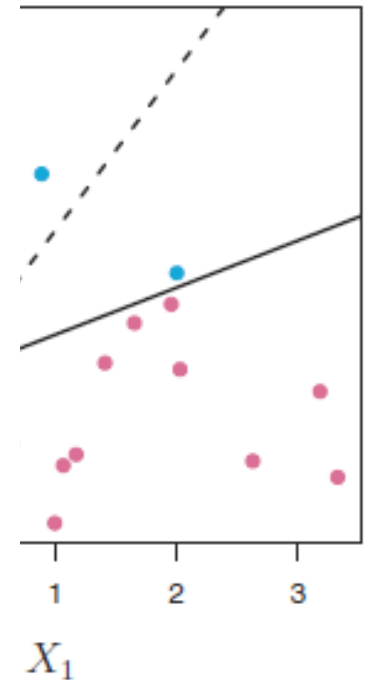
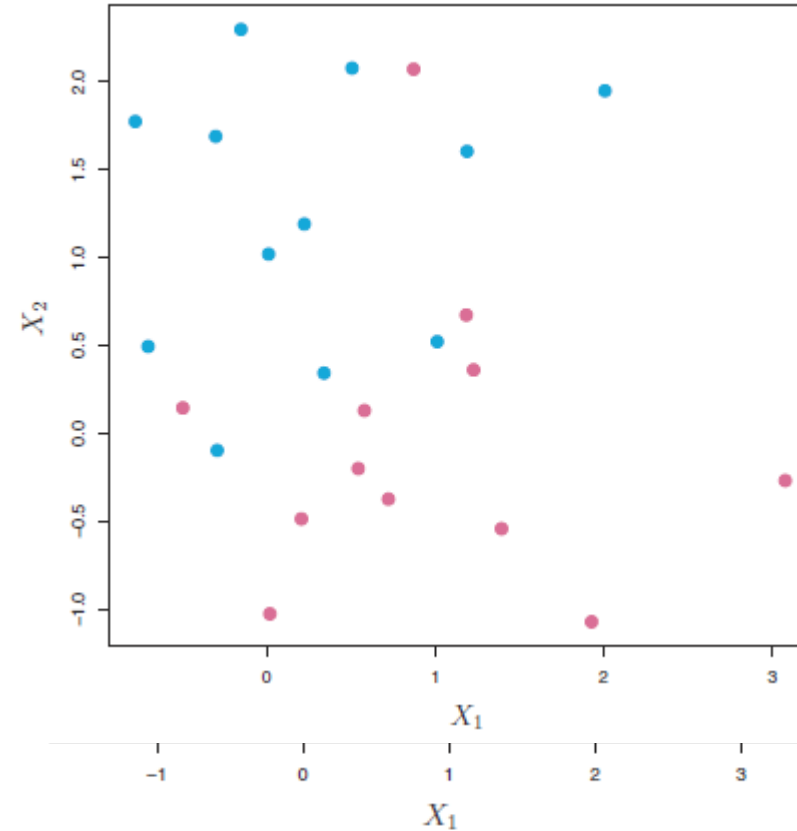
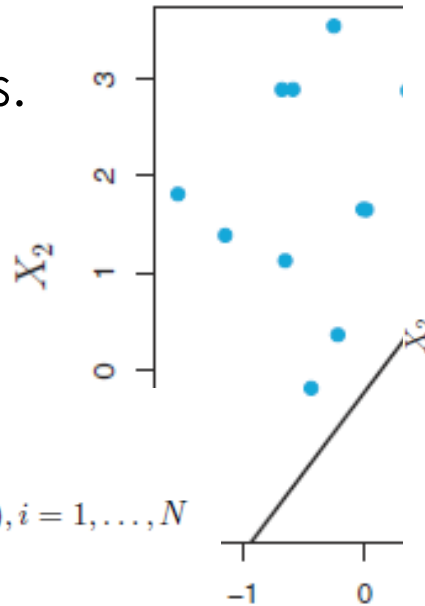
EDUCACIÓN
PROFESIONAL

OTROS MODELOS

Support Vector Machines (SVM)

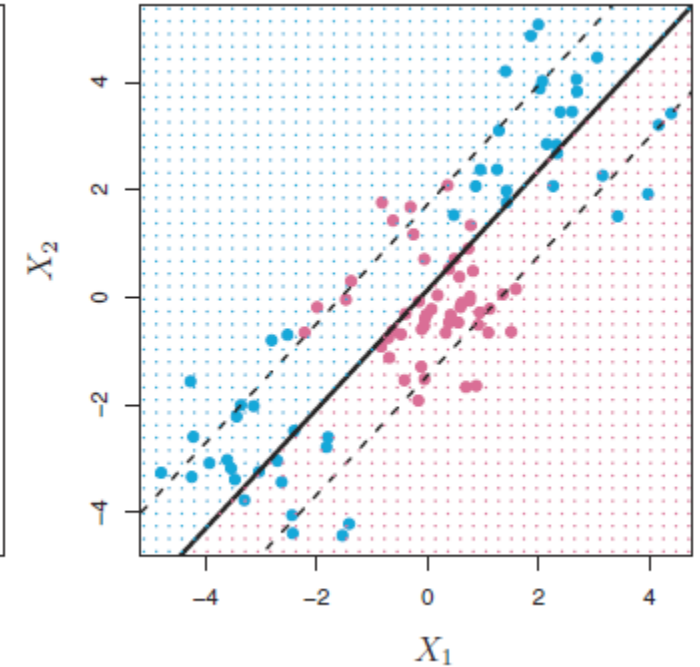
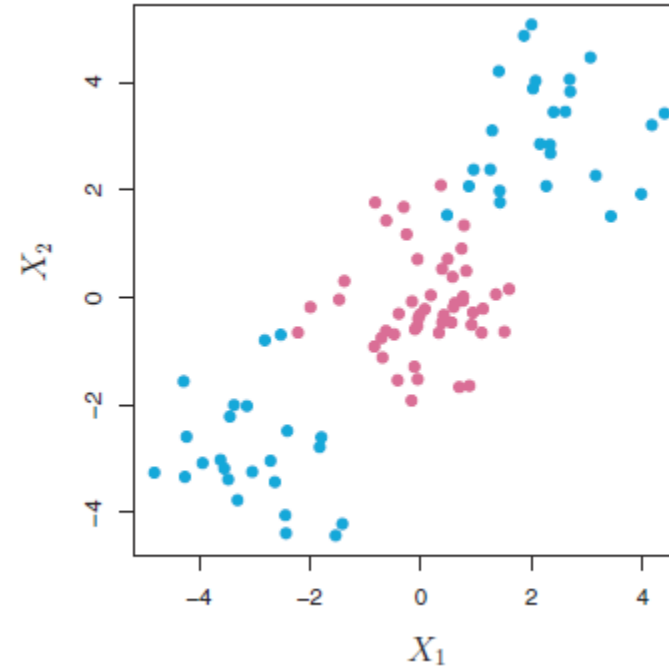
- Separación por hiperplanos. (maximal margin).
- Caso no separable. (Soft margin)

$$\left\{ \begin{array}{l} \max_{\beta, \beta_0, ||\beta||=1} M \\ \text{sujeto a } \xi \geq 0, \sum_{i=1}^N \xi_i \leq K, y_i(x_i^T \beta + \beta_0) \geq M(1 - \xi_i), i = 1, \dots, N \end{array} \right.$$



Support Vector Machines (SVM)

- Fronteras de decisión no lineales.
- Se introduce flexibilidad mediante la utilización de Kernels.



Support Vector Machines (SVM)

Considérese el set de entrenamiento $\mathbf{x}_1, \dots, \mathbf{x}_N \in \mathcal{X}$, con $N \in \mathbb{N}$ y $\mathcal{X} \subset \mathbb{R}^n$ compacto. De acuerdo a la formulación tradicional de un clasificador SVM (véase por ejemplo [1]), supóngase un kernel del tipo

$$k(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle,$$

donde ϕ es una función que mapea $\mathcal{X} \rightarrow \mathcal{H}$, con \mathcal{H} un espacio completo y con un producto interno $\langle \cdot, \cdot \rangle$ definido. Para efectos de esta aplicación $\mathcal{H} = \mathbb{R}^p, p \in \mathbb{N}$, parámetro que variará con cada uno de los modelos a implementar.

El Kernel a utilizar en este modelo es el radial, dado por

$$k(\mathbf{x}, \mathbf{y}) := \exp(-\gamma \|\mathbf{x} - \mathbf{y}\|^2)$$



B. Schölkopf and A.J. Smola, *Learning with Kernels*. MIT Press, 2002.

Support Vector Machines (SVM)

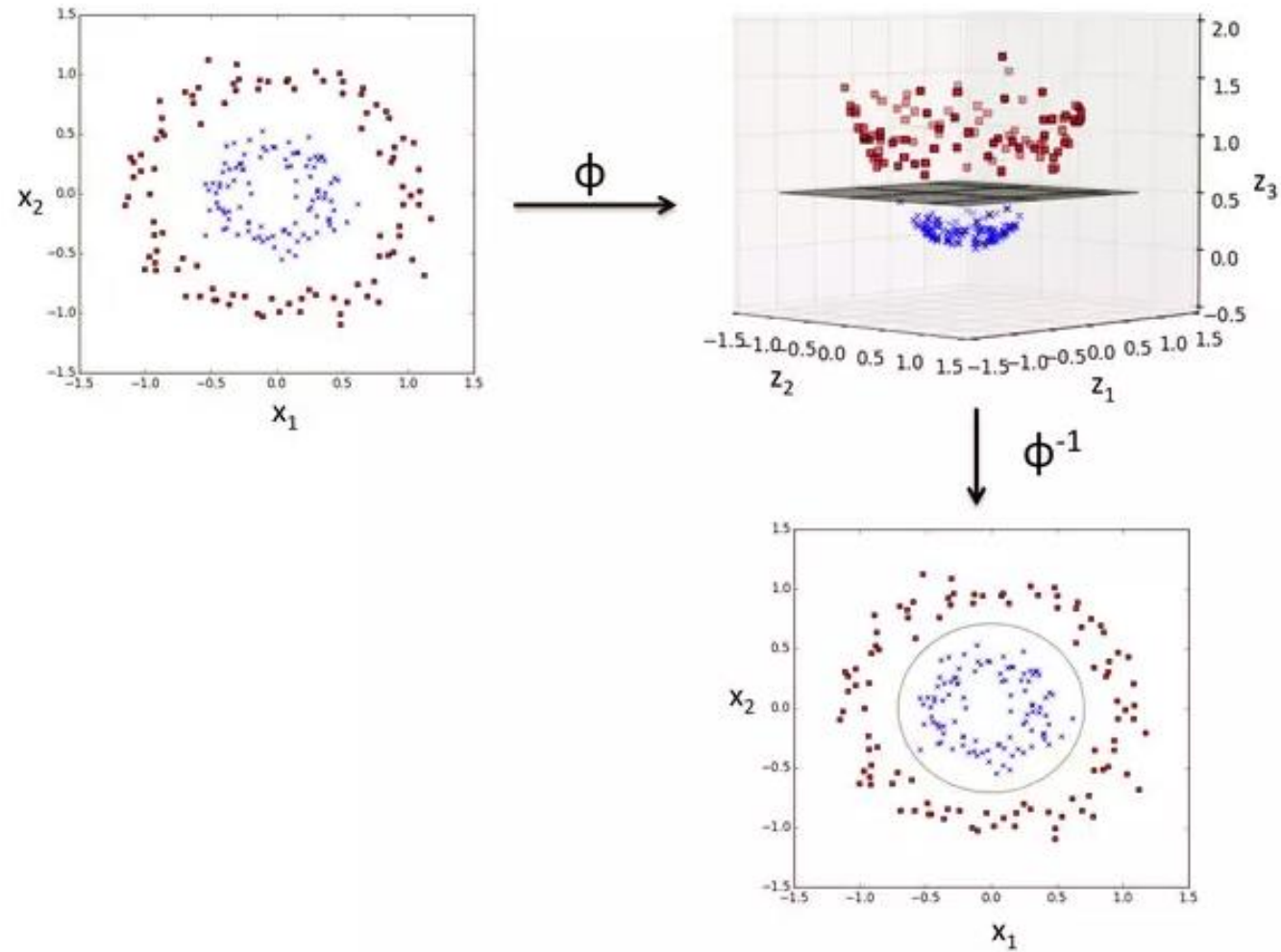
$$\left\{ \begin{array}{l} \max_{\beta, \beta_0, ||\beta||=1} M \\ \text{sujeto a } \xi \geq 0, \sum_{i=1}^N \xi_i \leq K, y_i(x_i^T \beta + \beta_0) \geq M(1 - \xi_i), i = 1, \dots, N \end{array} \right.$$



$$\left\{ \begin{array}{l} \min_{\beta, \beta_0} \frac{1}{2} ||\beta||^2 + C \sum_{i=1}^N \xi_i \\ \text{sujeto a } \xi \geq 0, y_i(\langle \phi(x_i), \beta \rangle + \beta_0) \geq (1 - \xi_i), i = 1, \dots, N \end{array} \right.$$



Support Vector Machines (SVM)



Support Vector Machines (SVM)



Regresión Logística

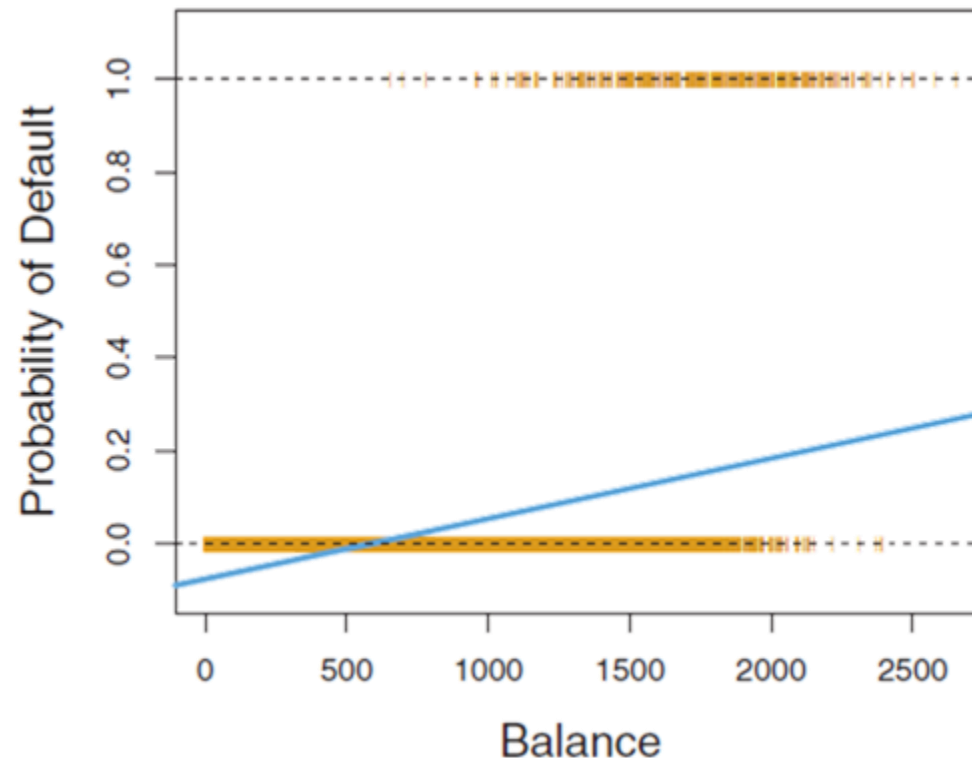
Un poco de contexto:

- Por ejemplo, supongamos que nos interesa saber si una persona quedará en “default” (no terminará de pagar su tarjeta de crédito), en función de atributos como su ingreso, nivel de deuda y si es estudiante o no.
- Para tal efecto nos interesa modelar entonces la probabilidad de caer en “default”.



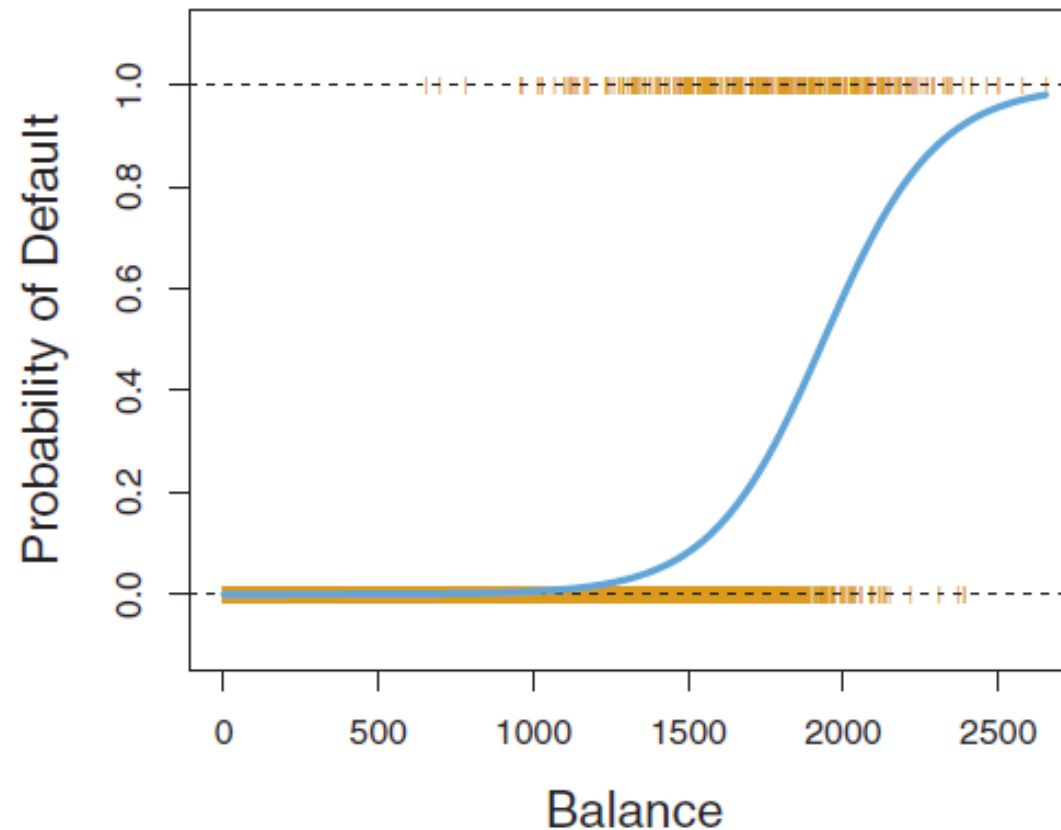
Regresión Logística

- Supongamos que en una primera aproximación sólo modelamos “default” en función del balance de la cuenta.
- ¿Que problema podríamos tener si modelamos la probabilidad de “default” mediante una regresión lineal?



Regresión Logística

- Dado lo anterior, nos gustaría poder establecer un modelo que nos permita modelar la probabilidad de “default”, por ejemplo, de la siguiente manera:



Regresión Logística

- Un posible modelo, corresponde a la regresión logística, el cual se especifica de la siguiente manera:
- Supongamos que Y representa el estado del cliente, el cual se codifica como “Yes” si el cliente cayó en “default” y “No”, en caso contrario.

$$\log\left(\frac{P(Y = \text{Yes}|\text{income})}{1 - P(Y = \text{Yes}|\text{income})}\right) = \beta_0 + \beta_1 \text{income}$$

- Dada la ecuación anterior, es posible estimar los parámetros β_0 y β_1 , mediante un procedimiento “similar” al de una regresión lineal.
- De manera más general, cuando incluimos más de una variable explicativa, el modelo se especifica de la siguiente manera.

$$\log\left(\frac{P(Y = 1|\mathbf{X} = \mathbf{x})}{1 - P(Y = 1|\mathbf{X} = \mathbf{x})}\right) = \beta_0 + \beta_1 X_1 + \dots + \beta_p X_p$$



Regresión Logística

- En R, es posible ajustar un modelo logístico mediante la función `glm` la cual recibe como principales argumentos:
- `formula`: Fórmula correspondiente al modelo planteado.
- `family`: Argumento que especifica la distribución de los errores así como la función de enlace al modelo lineal generalizado que se busca ajustar. Para ajustar una regresión logística este parámetro por defecto está seteado a modo de ajustar una reg. logística cuando la variable respuesta es binaria.

`family = binomial(link = "logit")`

- `data`: Dataset desde donde se mapen las variables indicadas en formula.



Regresión Logística



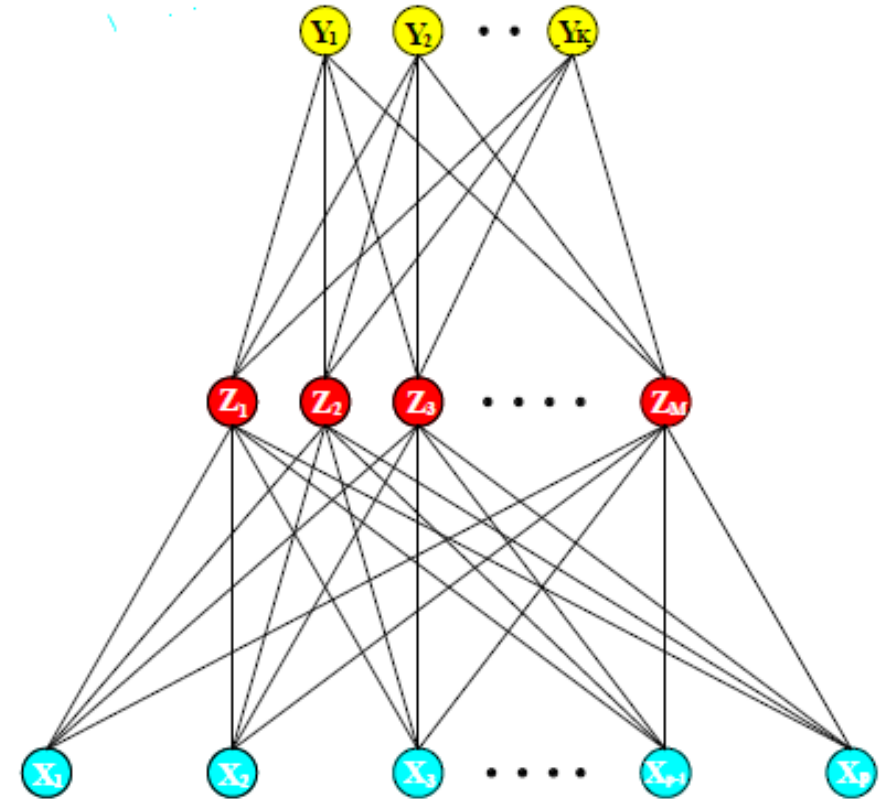
Redes neuronales

Single hidden layer
back-propagation

- Discusión: single hidden layer back-propagation network, or single layer perceptron.
- Objetivo: Aprendizaje supervisado.
- Regresión y clasificación.
- Sólo otra versión de modelos no lineales.
- Mayor precisión a costo de menor interpretabilidad.

Referencias de interés

- The Elements of Statistical Learning. Trevor Hastie, Robert Tibshirani, Jerome Friedman (ESLII de nuestra carpeta)
- Deep Learning with Python. François Chollet.
- Deep Learning. Ian Goodfellow, Yoshua Bengio & Aaron Courville.



Redes neuronales

Single hidden layer
back-propagation

- Función de activación: $\sigma(v)$
- Units (en la capa oculta): T_k
- Pesos:

$$\begin{aligned} \{\alpha_{0m}, \alpha_m; m = 1, 2, \dots, M\} & M(p+1) \\ \{\beta_{0k}, \beta_k; k = 1, 2, \dots, K\} & K(M+1) \end{aligned}$$

- $g_k(\mathbf{T})$ en el caso de regresión suele ser la identidad mientras que en regresión la función softmax:

$$g_k(\mathbf{T}) = \frac{e^{T_k}}{\sum_{\ell=1}^K e^{T_\ell}}.$$

For K -class classification, there are K units at the top, with the k th unit modeling the probability of class k . There are K target measurements Y_k , $k = 1, \dots, K$, each being coded as a 0 – 1 variable for the k th class.

Derived features Z_m are created from linear combinations of the inputs, and then the target Y_k is modeled as a function of linear combinations of the Z_m ,

$$\begin{aligned} Z_m &= \sigma(\alpha_{0m} + \alpha_m^T X), \quad m = 1, \dots, M, \\ T_k &= \beta_{0k} + \beta_k^T Z, \quad k = 1, \dots, K, \\ f_k(X) &= g_k(T), \quad k = 1, \dots, K, \end{aligned} \tag{11.5}$$

where $Z = (Z_1, Z_2, \dots, Z_M)$, and $T = (T_1, T_2, \dots, T_K)$.

The activation function $\sigma(v)$ is usually chosen to be the *sigmoid* $\sigma(v) = 1/(1 + e^{-v})$; see Figure 11.3 for a plot of $1/(1 + e^{-v})$. Sometimes Gaussian radial basis functions (Chapter 6) are used for the $\sigma(v)$, producing what is known as a *radial basis function network*.

Neural network diagrams like Figure 11.2 are sometimes drawn with an additional *bias* unit feeding into every unit in the hidden and output layers.



Redes neuronales

Single hidden layer
back-propagation

Algunas consideraciones al ajustar un red neuronal:

- Valores iniciales de los pesos (velocidad de la convergencia).
- Sobre-ajuste (número de iteraciones). Hoy en día se aplican técnicas de regularización similares a LASSO, o bien detener el proceso iterativo hasta cierta estabilidad (sobreajuste en mínimo global de la función de pérdida) (weight decay).
- Escalar los inputs, controla escala pesos en primera capa.
- Número de unidades tuneable de las capas ocultas.

