

# Introducción a la minería de datos

Universidad Nacional de Colombia, sede Medellín.

[ciencias.medellin.unal.edu.co](http://ciencias.medellin.unal.edu.co)

Facultad de Ciencias  
Sede Medellín



UNIVERSIDAD  
**NACIONAL**  
DE COLOMBIA

# Objetivos de la sesión

- Introducción a herramientas de Python para machine learning.
- Aprender acerca de principios generales del aprendizaje supervisado.

# Librerías Python: Scikit Learn

- **Scikit Learn homepage:**  
<https://scikit-learn.org/stable/>
- **Scikit Learn Guía de Usuario:**  
[https://scikit-learn.org/stable/user\\_guide.html](https://scikit-learn.org/stable/user_guide.html)
- **Scikit Learn API reference:**  
<https://scikit-learn.org/stable/modules/classes.html>



Generalmente en python importamos clases y funciones como:

```
from sklearn.model_selection import  
train_test_split  
from sklearn.tree import DecisionTreeClassifier
```

# Librerías Python: SciPy

<https://www.scipy.org/>



- Proporciona una variedad de herramientas computacionales científicas que incluyen distribuciones estadísticas, optimización de funciones, álgebra lineal, etc.

Ejemplo:

```
import scipy as sp
```

# Librerías Python: Numpy

<https://numpy.org/>

- Proporciona estructuras de datos fundamentales utilizadas por scikit-learn, en particular matrices multidimensionales.
- En ocasiones los datos que se ingresan en scikit learn estarán en la forma de una matriz Numpy.



Ejemplo:

```
import numpy as np
```

# Librerías Python: pandas

<https://pandas.pydata.org/>



- Proporciona estructuras de datos fundamentales como **DataFrame**.
- Soporte para lectura/escritura de datos en diferentes formatos.

Ejemplo:

```
import pandas as pd
```

# Librerías Python: Visualización



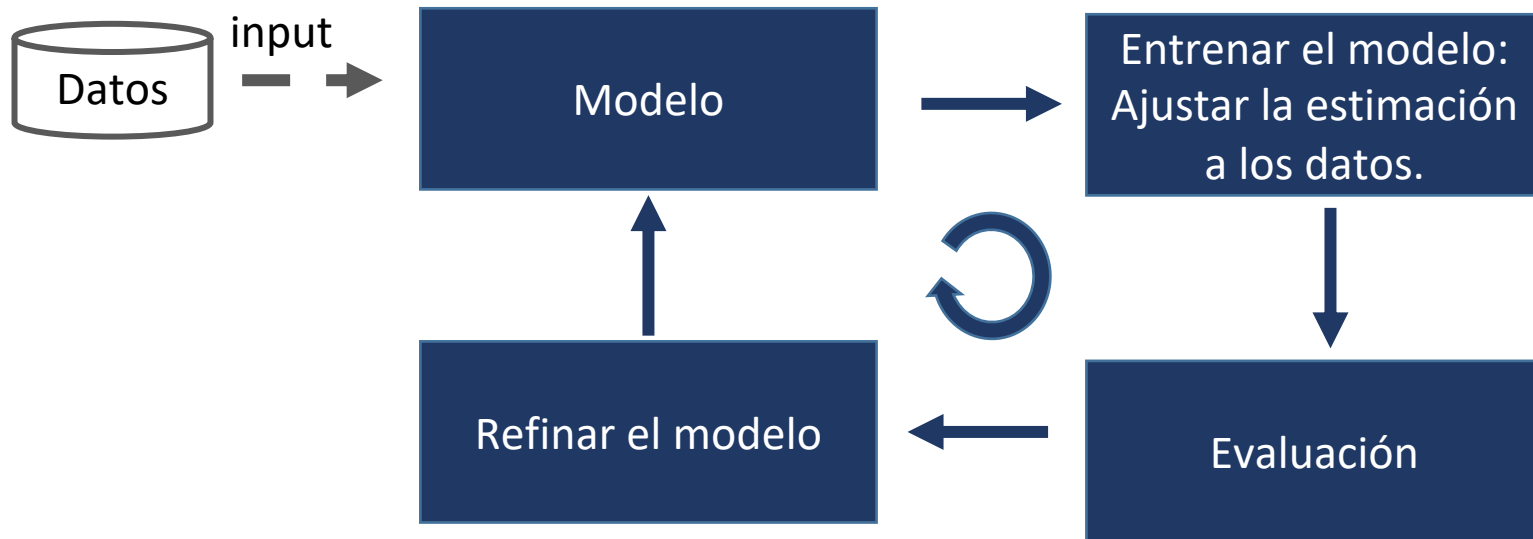
- Típicamente se usa el módulo pyplot de matplotlib (<https://matplotlib.org/>):

```
import matplotlib.pyplot as plt
```

- Una librería de visualización común es seaborn (<https://seaborn.pydata.org/>):

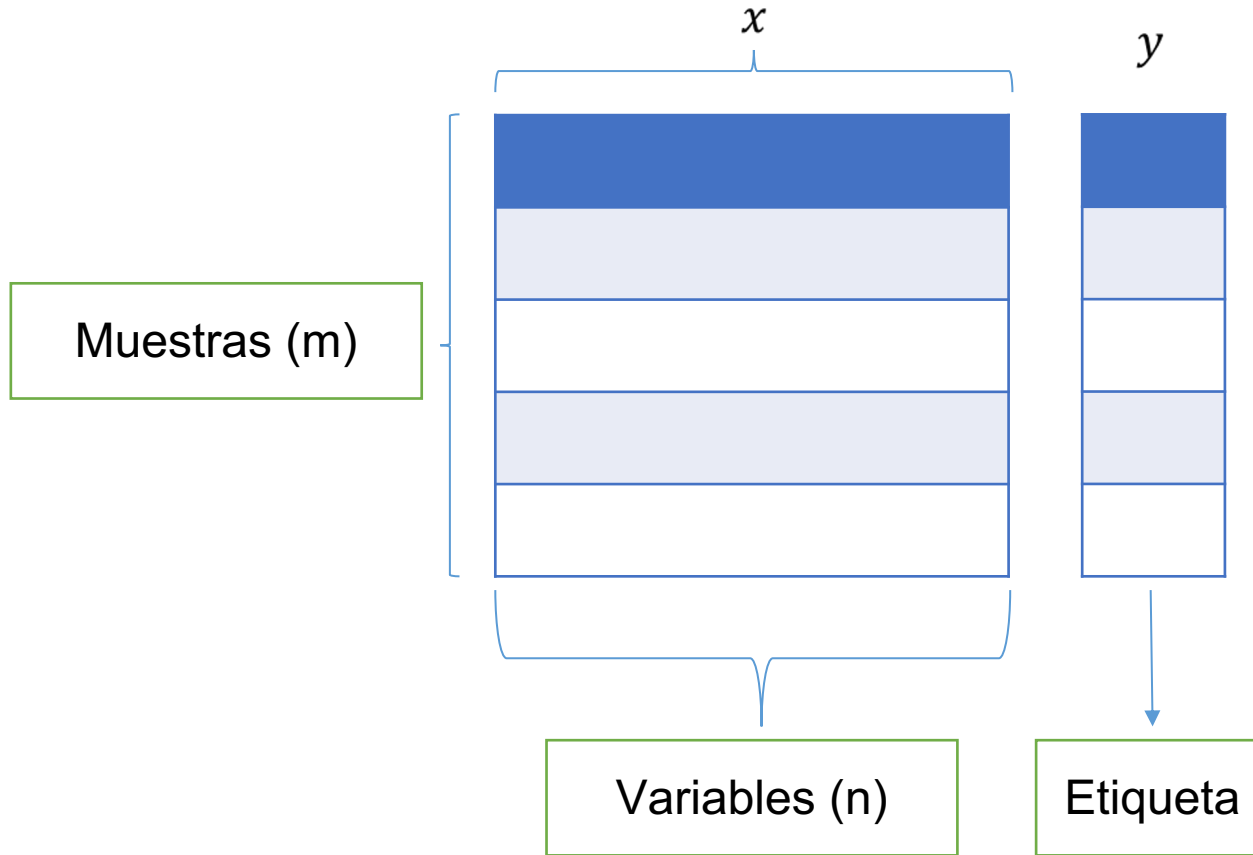
```
import seaborn as sn
```

# Representar/Entrenar/Evaluar/Refinar





# Generalidades: Los datos...



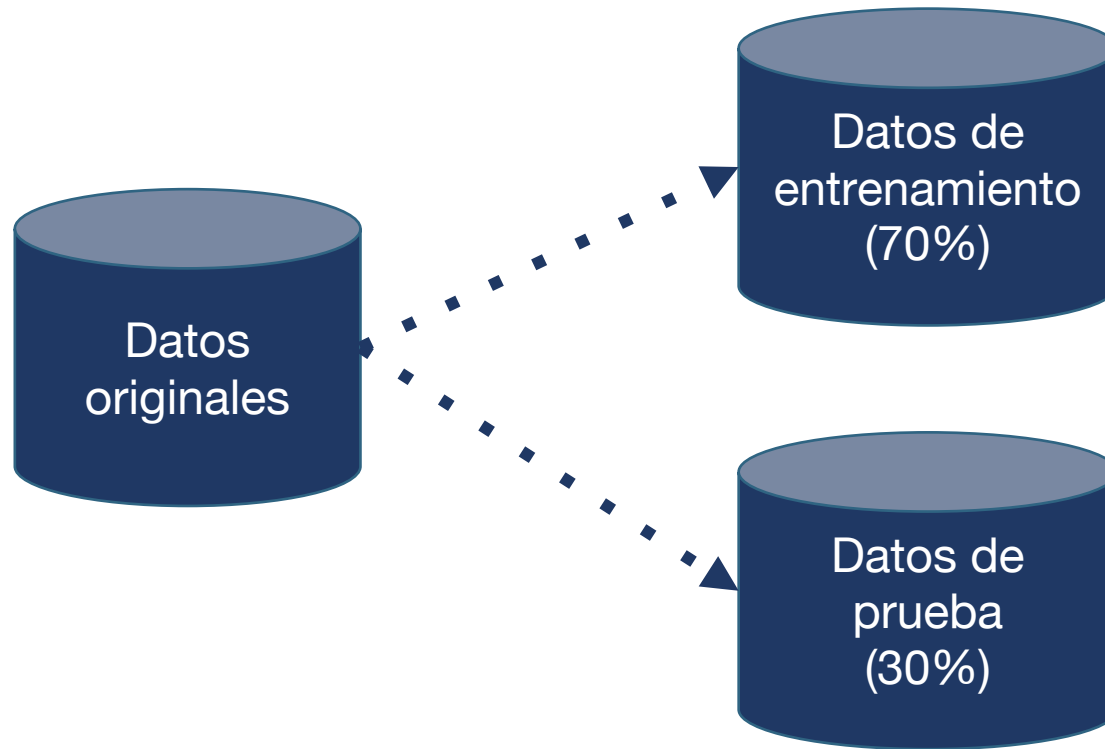
# Generalidades: El modelo

El modelo es una función matemática que mapea un conjunto de características (input) en una salida (output). La salida puede ser un vector, una variable **categorica** o una variable **numérica**.

$$y = f(x)$$

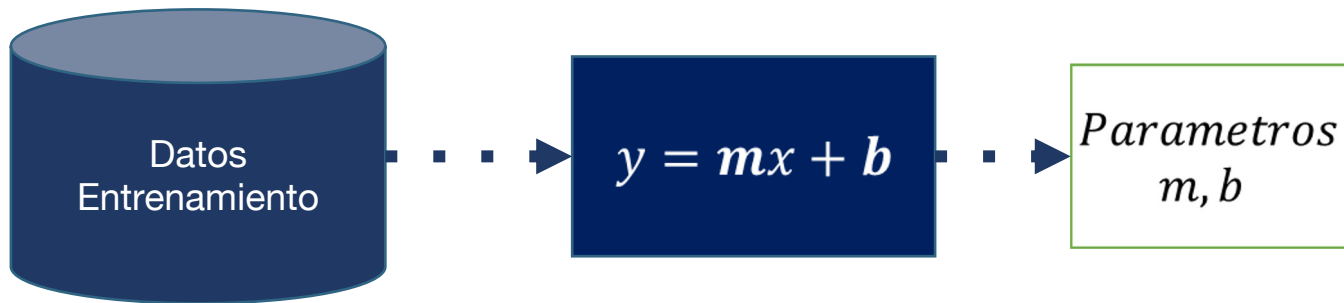
$$y = mx + b$$

# Generalidades: Entrenamiento y prueba



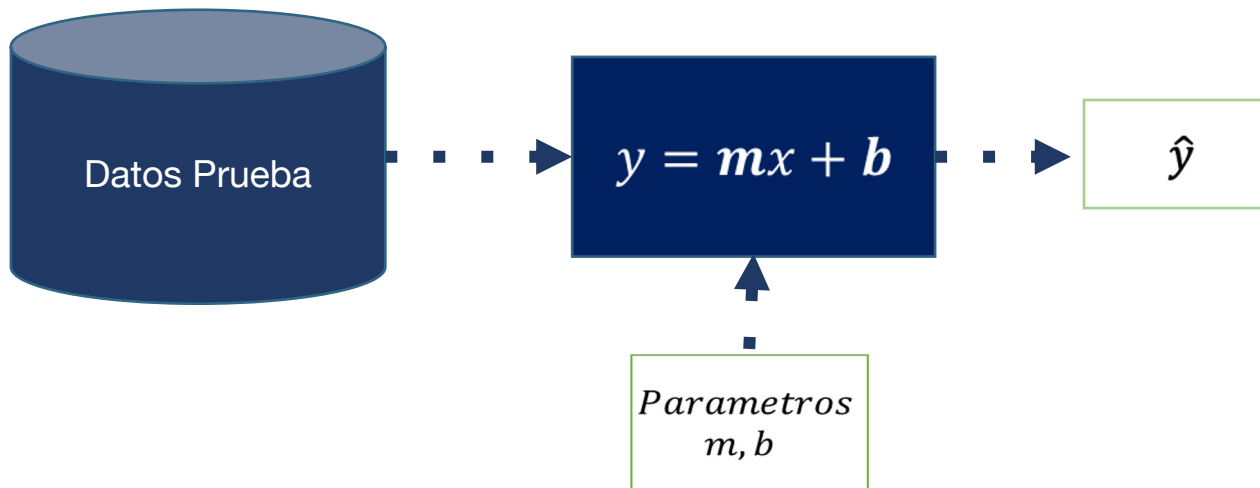
# Generalidades: Entrenamiento (training)

Usar datos para aprender o encontrar los mejores parámetros de un modelo.




# Generalidades: Prueba (test)

Usar los parámetros usados en el entrenamiento para hacer predicciones.



# Generalidades: El modelo y sus hiperparámetros

Los hiperparámetros son valores que son seleccionados previo al modelo y cambian la forma de este.

$$y = f(x|m, b)$$

$$y = mx^2 + b$$
$$y = mx + b$$

# Generalidades: El modelo y sus hiperparámetros

- Los hiperparámetros se pueden seleccionar a priori o usar diferentes técnicas de validación para encontrar los mejores.
- Algunos ejemplos hiperparámetros:
  - Cantidad de capas en una red neuronal
  - Cantidad de neuronas en una capa
  - Cantidad de árboles en un bosque aleatorio
  - Número de clústers en K-Means

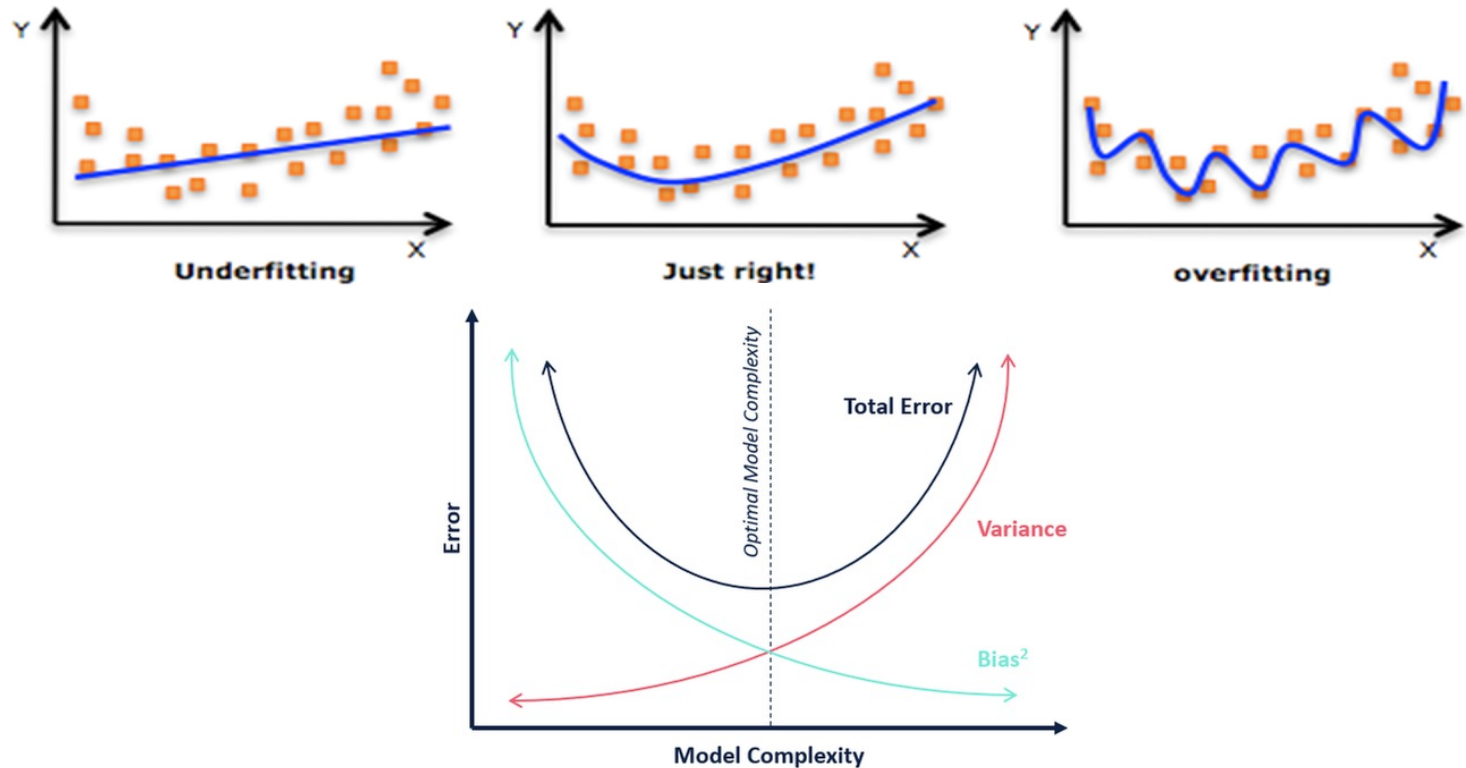
# Generalidades: Optimización

¡Es el aprendizaje en sí! Consiste en encontrar el conjunto de parámetros que minimizan (o maximizan una función de costo).

- Greedy search
- **Gradient descent (gradiente descendiente)**
- Programación lineal
- Otros



# Generalidades: Generalización- Overfitting - Underfitting



# Generalidades: Generalización- Overfitting - Underfitting

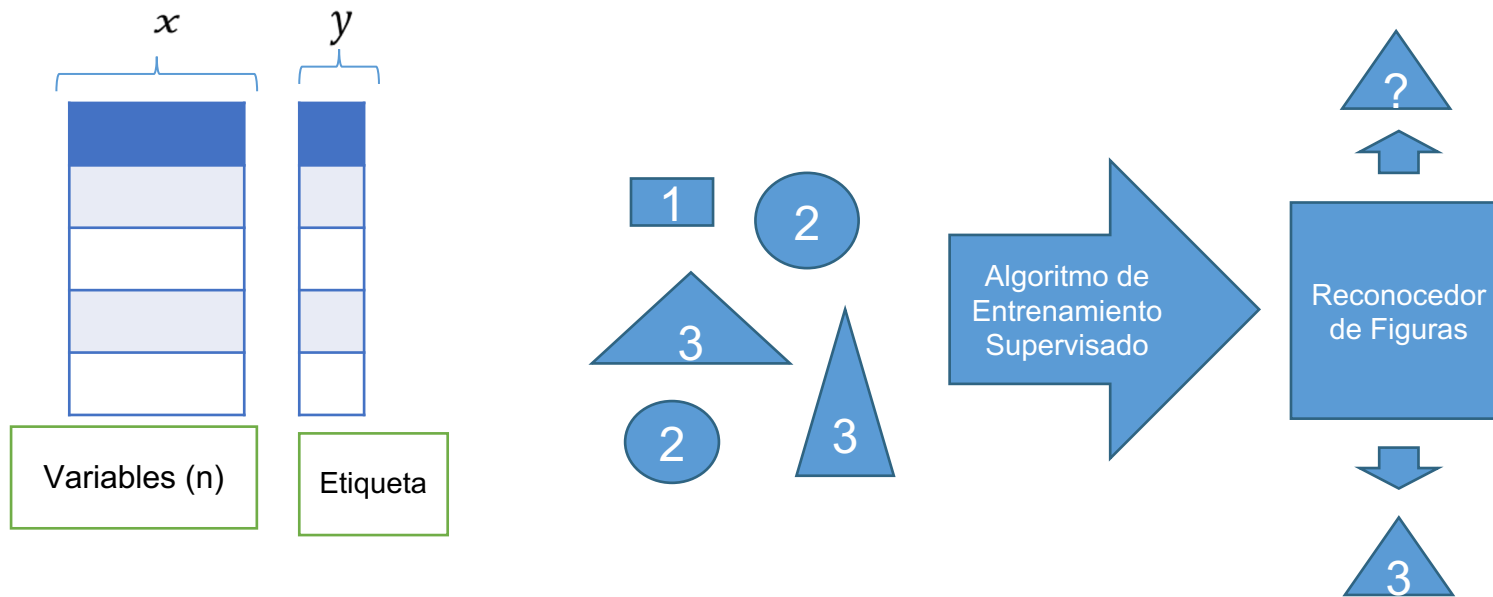
## Overfitting

- Aumentar la cantidad de datos
- Seleccionar menos características
- Seleccionar mejores características
- Reducir la complejidad del modelo
- Crear “ensemble models”
- Validación cruzada
- Regularización

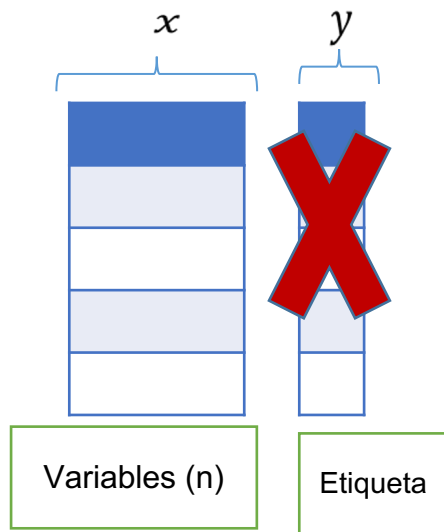
## Underfitting

- Aumentar la complejidad del modelo
- Seleccionar mejores características

# Generalidades: Aprendizaje supervisado y no supervisado



# Generalidades: Aprendizaje supervisado y no supervisado



# Generalidades: tipos de modelos

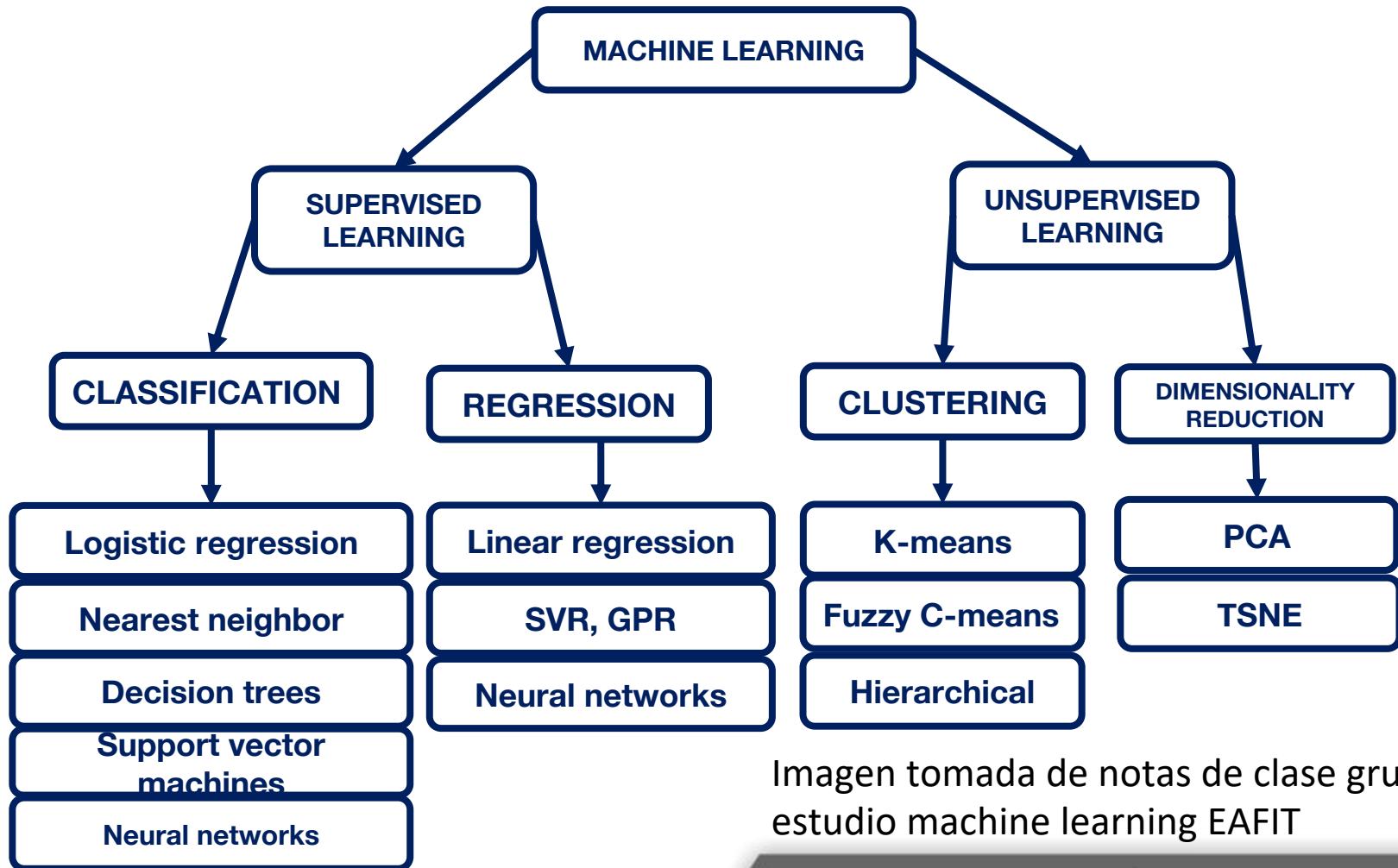


Imagen tomada de notas de clase grupo de estudio machine learning EAFIT

# Regresión Lineal

Un modelo lineal es una suma de variables ponderadas que predice un valor de salida objetivo dado un dato de entrada.

Ejemplo: predecir los precios de vivienda ( $X_1$ : *predial*,  $X_2$ : *antigüedad*).

$$\hat{Y} = 212000 + 109X_1 - 2000X_2$$

Una casa con características ( $x_1 = 10000$ ,  $x_2 = 20$ ), tendrá una predicción de:

$$\hat{Y} = 212000 + 109(10000) - 2000(20) = 1262000$$

# Regresión Lineal

**Vector de características (input):**  $x = (x_1, x_2, \dots, x_d)$

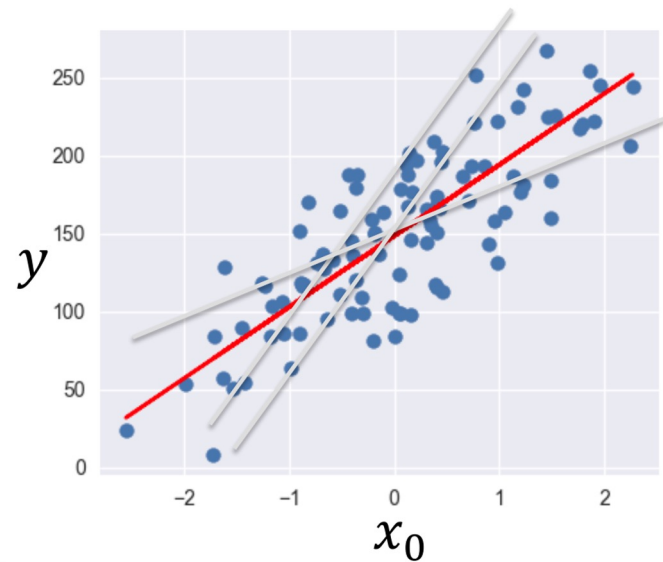
**Relación impuesta en el modelo:**  $y = w_0 + w_1x_1 + \dots + w_dx_d$

**Valor Predicho (output):**  $\hat{y} = \hat{w}_0 + \hat{w}_1x_1 + \hat{w}_2x_2 + \dots + \hat{w}_dx_d$

**Parámetros a estimar:**

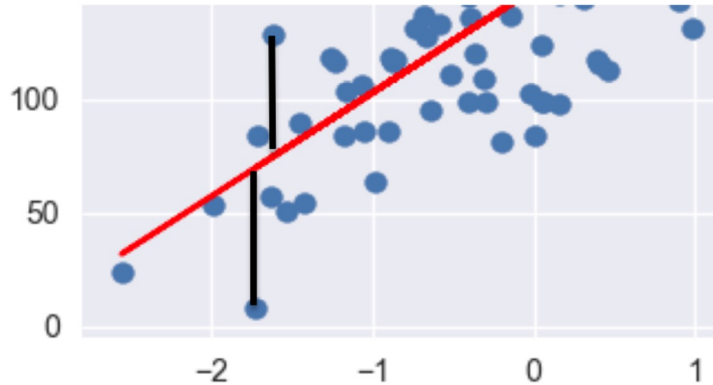
$w = (w_1, \dots, w_d)$ : pesos de las variables.

$w_0$ : intercepto



# Regresión Lineal

- Parámetros son estimados con los datos de entrenamiento.
- Hay muchas formas de estimar  $w$ .
- El algoritmo de aprendizaje encuentra los parámetros que optimizan una función objetivo, normalmente minimiza una una función de pérdida de los valores predichos vs los valores reales de la variable.



Encontrar  $W$  tal que minimice la suma de cuadrados del error (RSS), de donde se deriva el error cuadrático medio.

$$RSS(w) = \sum_{i=1}^n (y_i - (w_0 + w_1 x))^2$$



# Regresión Lineal

Datos de entrenamiento

$$y_1 \quad \mathbf{x}_1$$

$$y_2 \quad \mathbf{x}_2$$

$$\vdots$$

$$y_n \quad \mathbf{x}_i$$

$$\vdots$$

$$y_i \quad \mathbf{x}_n$$

Predicciones durante el entrenamiento

$$\hat{y}_1 = \hat{f}(\mathbf{x}_1) = \mathbf{x}_1^T \mathbf{w}$$

$$\hat{y}_2 = \hat{f}(\mathbf{x}_2) = \mathbf{x}_2^T \mathbf{w}$$

$$\vdots$$

$$\hat{y}_i = \hat{f}(\mathbf{x}_i) = \mathbf{x}_i^T \mathbf{w}$$

$$\vdots$$

$$\hat{y}_n = \hat{f}(\mathbf{x}_n) = \mathbf{x}_n^T \mathbf{w}$$

Se necesita encontrar  $\mathbf{w}$  que haga pequeña la diferencia entre

$$\mathbf{w} = \arg \min_{\mathbf{w}} \sum_{i=1}^n (y_i - \mathbf{x}_i^T \mathbf{w})^2 = \arg \min_{\mathbf{w}} (Y - \mathbf{X}\mathbf{w})^T (Y - \mathbf{X}\mathbf{w})$$

Solución analítica  $\mathbf{w} = (\mathbf{X}^T \mathbf{X})^{-1} (\mathbf{X}^T Y)$

# Clasificación

- Técnica que “aprende” automáticamente cómo clasificar objetos en dos o más clases determinadas.
- Este aprendizaje se basa en datos previamente etiquetados.

Ejemplos de Aplicación:

1. Detección de fraudes.
2. Detección de Spam.
3. Diagnóstico de enfermedades.
4. Retención de clientes.
5. Modelos de Fuga.
6. Categorizar Tweets.

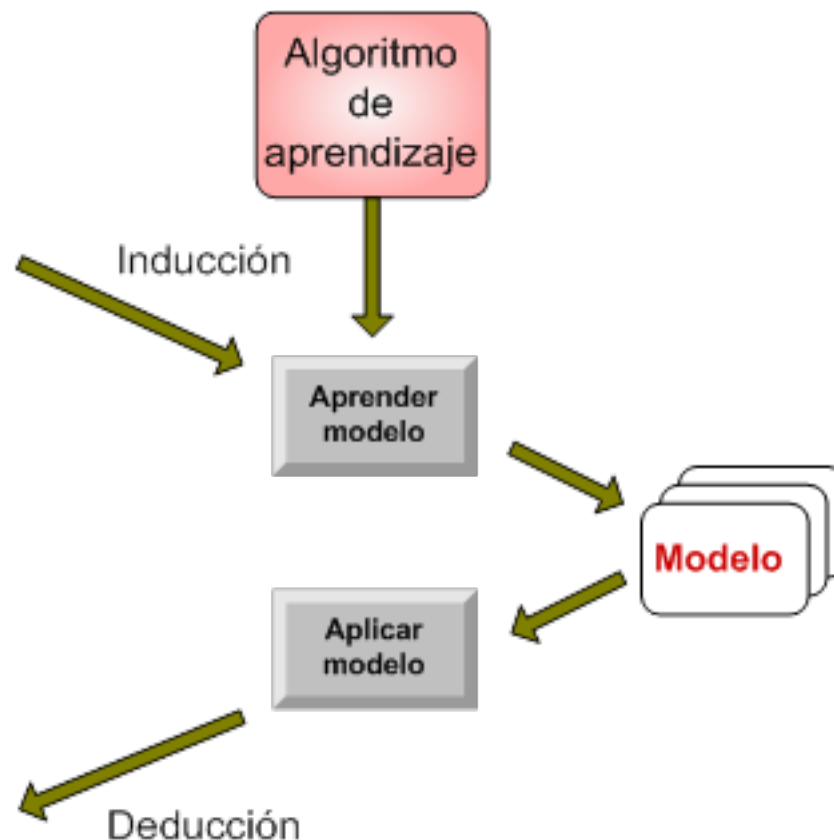
# Clasificación

<i>Id</i>	<i>Attrib1</i>	<i>Attrib2</i>	<i>Attrib3</i>	<i>Class</i>
1	Yes	Large	125 K	No
2	No	Medium	100 K	No
3	No	Small	70 K	No
4	Yes	Medium	120 K	No
5	No	Large	95 K	Yes
6	No	Medium	60 K	No
7	Yes	Large	220 K	No
8	No	Small	85 K	Yes
9	No	Medium	75 K	No
10	No	Small	90 K	Yes

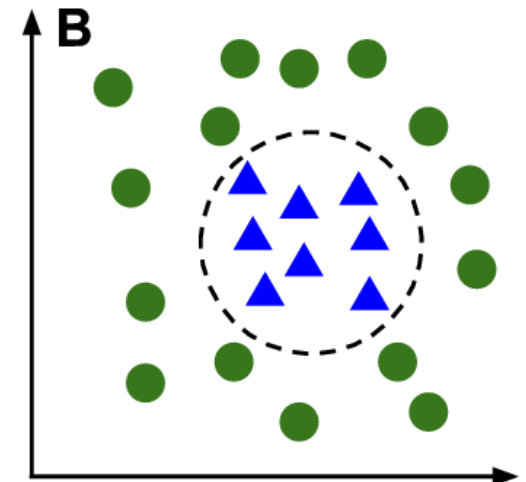
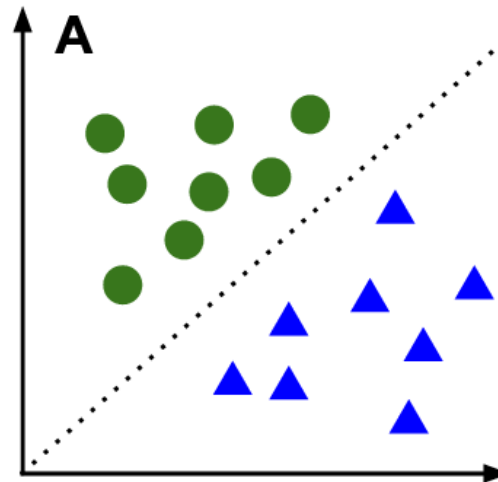
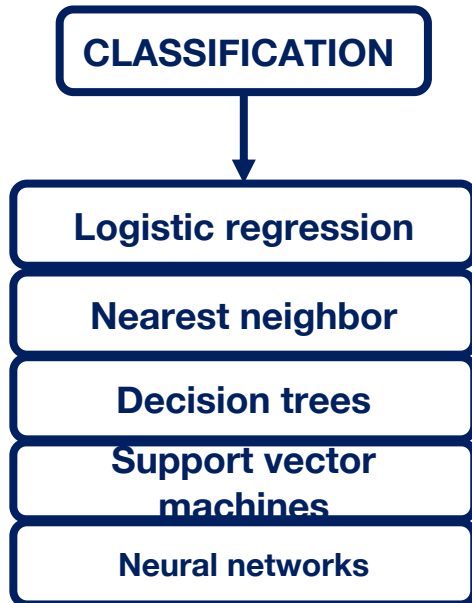
Conjunto de  
entrenamiento

<i>Id</i>	<i>Attrib1</i>	<i>Attrib2</i>	<i>Attrib3</i>	<i>Class</i>
11	No	Small	55 K	?
12	Yes	Medium	80 K	?
13	Yes	Large	110 K	?
14	No	Small	95 K	?
15	No	Large	67 K	?

Conjunto de prueba



# Técnicas de Clasificación



# Regresión Logística

$$y = \sigma(\mathbf{x}^T \mathbf{w})$$



La relación entre las variables de entrada y parámetros (pesos) del modelo es lineal

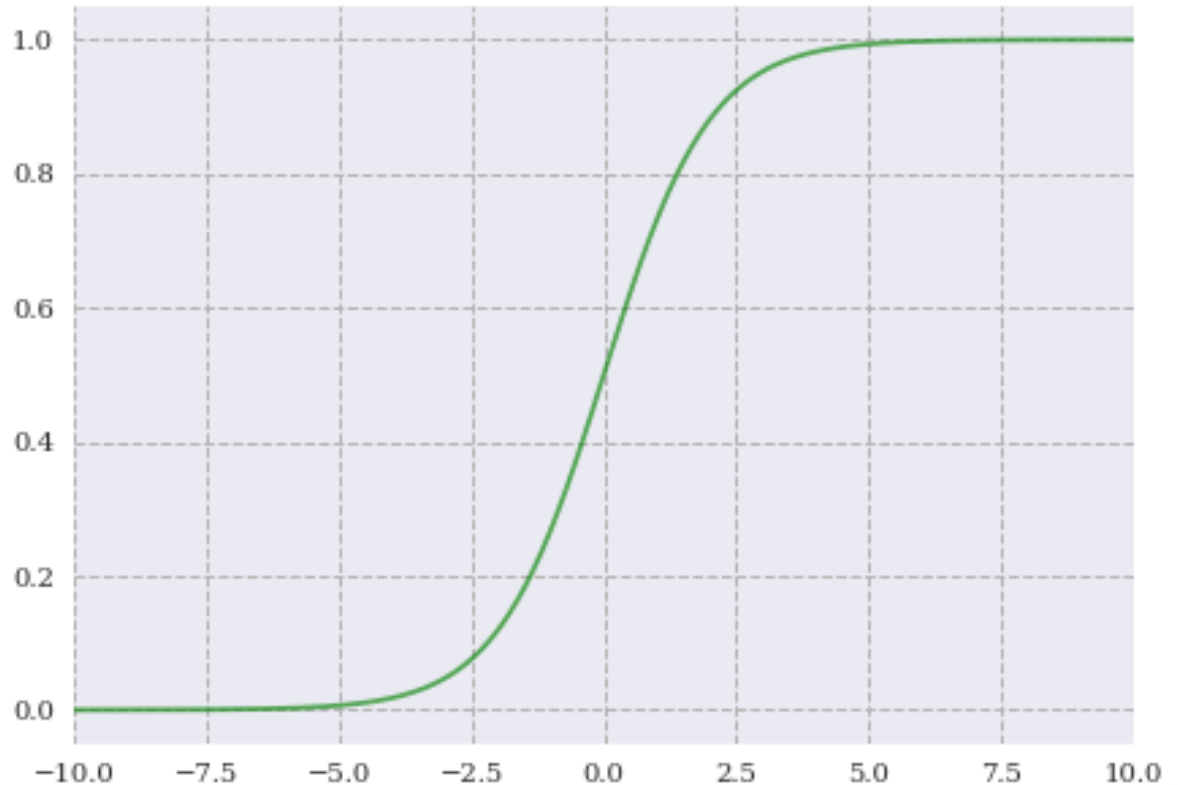
$$\sigma(x) = \frac{1}{1 + e^{-x}}$$



La función *sigmoid* permite *mappear* el resultado de la combinación lineal a un 1 o un 0

# Regresión Logística

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

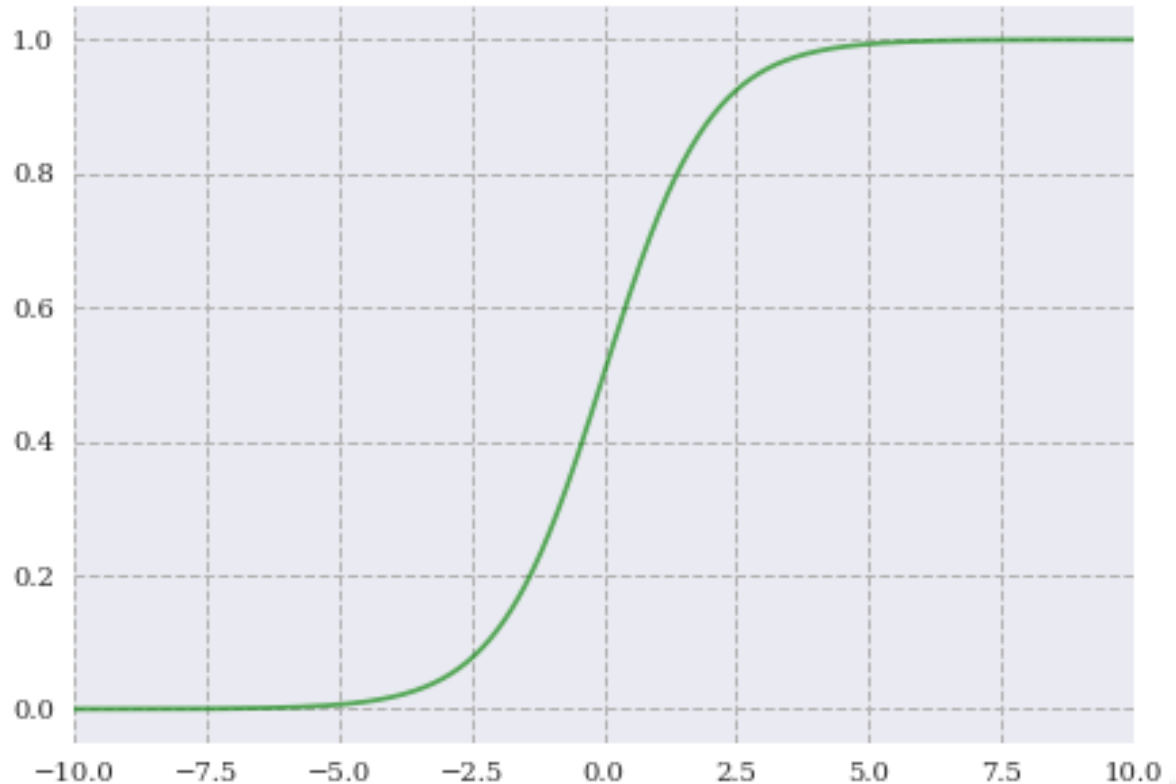


# Regresión Logística

La regresión logística es un modelo lineal que permite estimar la probabilidad de que el vector observado pertenezca a una clase.

$$\sigma(\mathbf{x}^T \mathbf{w}) \in (0, 1)$$

$$\hat{y} = \begin{cases} 1 & \text{si } \sigma(\mathbf{x}^T \mathbf{w}) \geq 0.5 \\ 0 & \text{si } \sigma(\mathbf{x}^T \mathbf{w}) < 0.5 \end{cases}$$

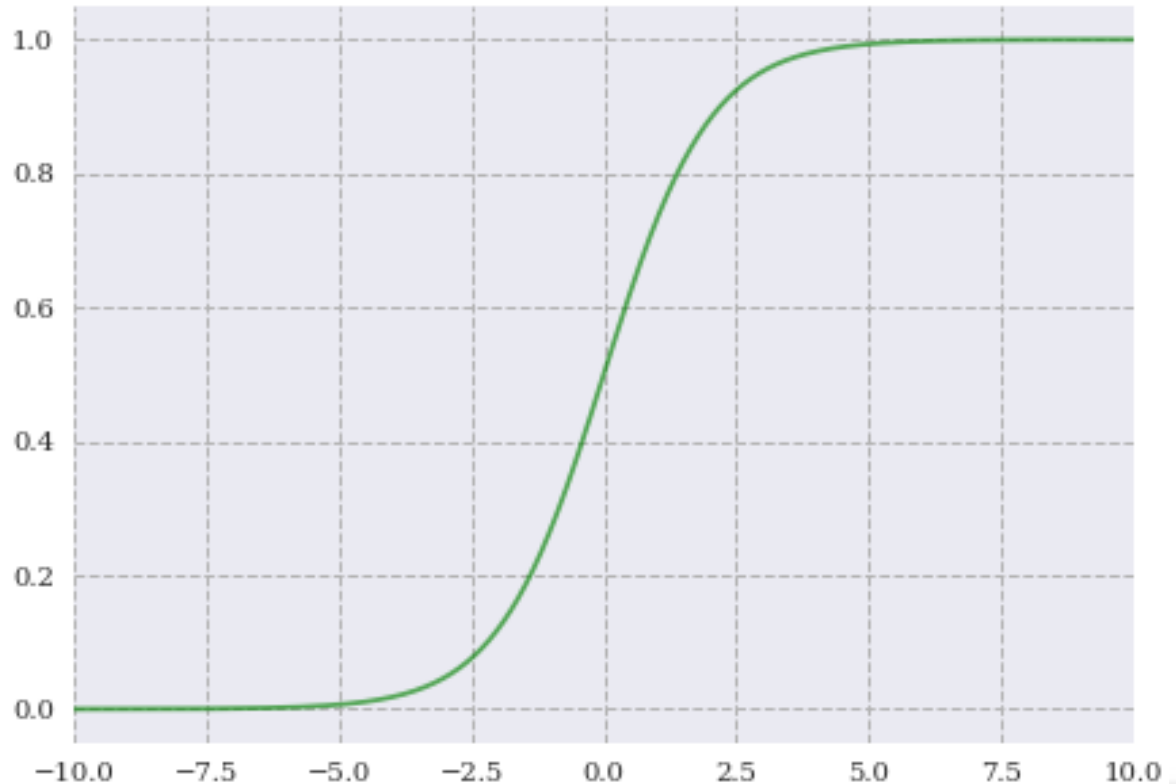


# Regresión Logística

La regresión logística es un modelo lineal que permite estimar la probabilidad de que el vector observado pertenezca a una clase.

$$\sigma(\mathbf{x}^T \mathbf{w}) \in (0, 1)$$

$$\hat{y} = \begin{cases} 1 & \text{si } \sigma(\mathbf{x}^T \mathbf{w}) \geq 0.5 \\ 0 & \text{si } \sigma(\mathbf{x}^T \mathbf{w}) < 0.5 \end{cases}$$





# Árbol de Decisión

Un árbol de decisión para clasificación es el resultado de aplicar una secuencia de preguntas; en cada paso de la secuencia la pregunta depende de las respuestas dadas a las preguntas previas.



# Árbol de Decisión

## Cualidades

No requiere escalar los datos

Puede usar variables categóricas y continuas sin mayor consideración

Son fáciles de interpretar, solo se debe navegar por el árbol generado

Se pueden usar para seleccionar variables relevantes.

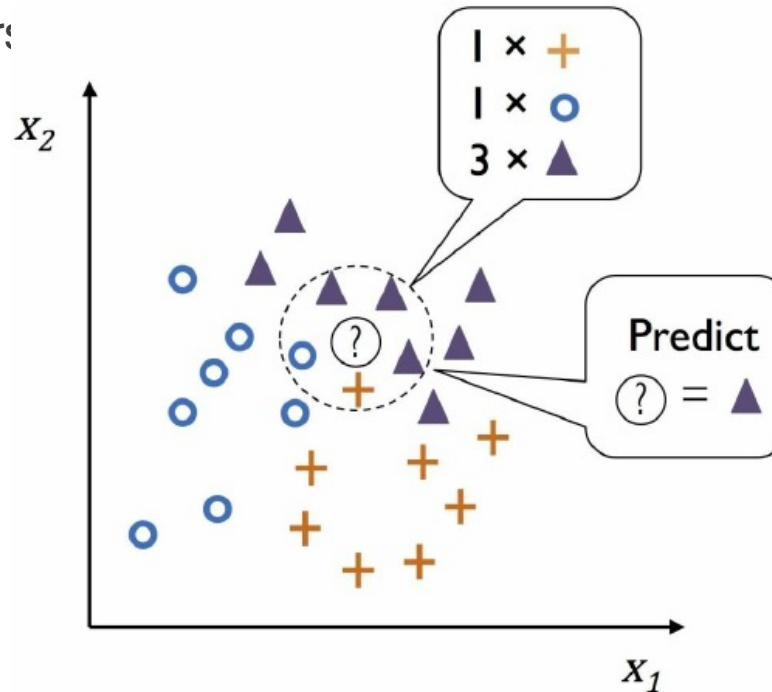
---

Si no se controla apropiadamente el crecimiento de un árbol, pueden sobreestimar

Es inestable: pequeñas variaciones en los datos pueden generar árboles muy diferentes.

# K – Vecinos más cercanos

KNN for K-nearest neighbors



Dado una observación, su grupo depende del grupo de las observaciones más cercanas.

Escoge un número de vecinos a tener en cuenta (K)

Escoge una métrica para calcular la distancia entre cualquier par de observaciones.

# Clasificación

Clasificación de múltiples clases.

**Algunos algoritmos tienen aplicabilidad directa para múltiples clases**

Árboles de decisión

K-NN

Redes neuronales

SVM

**Para todo lo demás existe...**

Uno vs uno (*one-vs-one*)

Uno vs el resto (*one-vs-rest*)

# Clasificación

Clasificación de múltiples clases: uno vs el resto



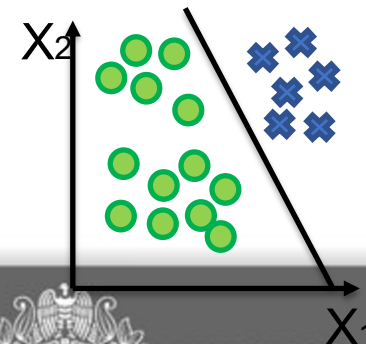
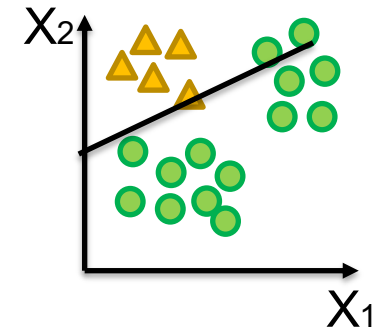
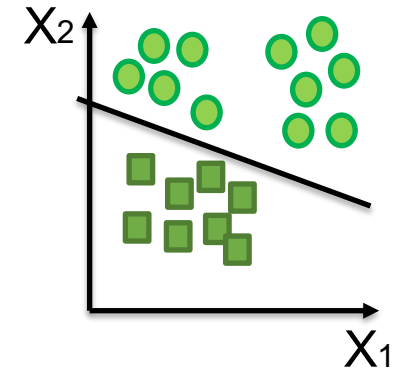
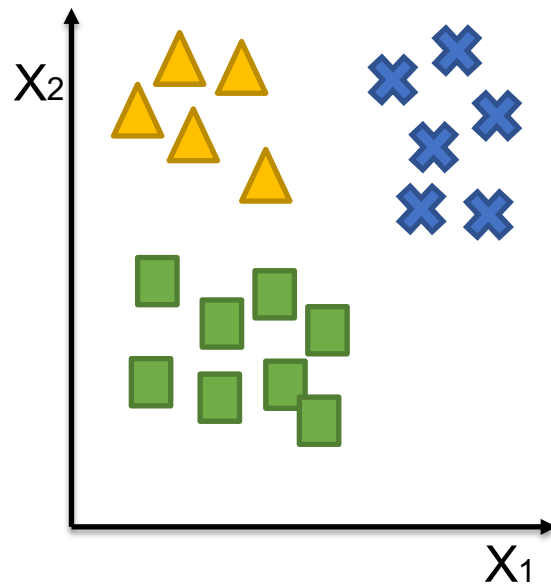
**Dividir el problema en  $k$  subproblemas de clasificación binaria:** es o no es de la clase  $k$

**Crear un clasificador  $\hat{f}_k$  par cada subproblema:** la clase  $k$  se define como la clase positiva y la unión de las otras clases se define como la clase negativa

**La predicción está basada en el modelo  $\hat{f}_k$  con mayor *certeza* para alguna observación  $x$ :** a la observación se le asigna la clase  $k$  que produce mayor *certeza*.

# Clasificación

Clasificación de múltiples clases: uno vs el resto



# Clasificación

Clasificación de múltiples clases: uno vs uno



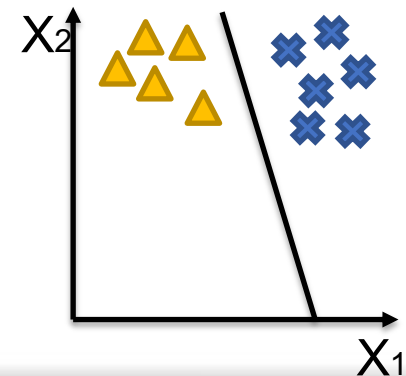
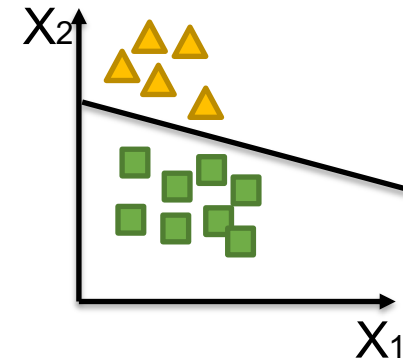
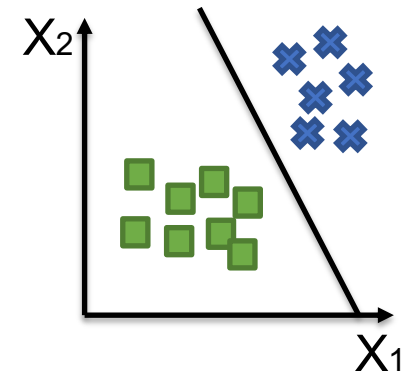
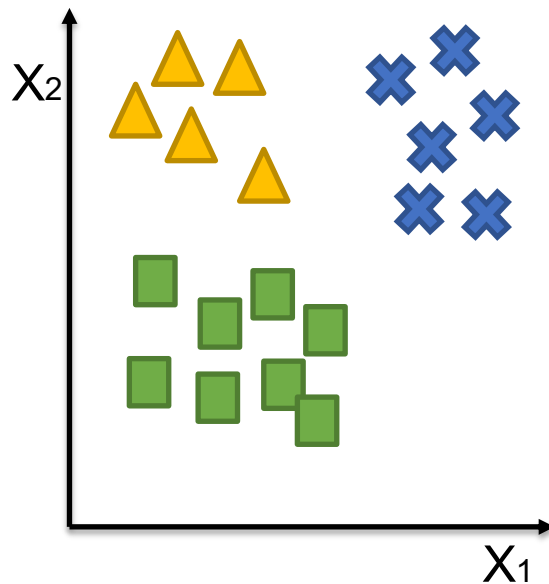
**Dividir el problema en  $\binom{k}{2}$  subproblemas de clasificación binaria:** es de la clase  $k_i$  o de la clase  $k_j \forall j \neq i$

**Crear un clasificador  $\widehat{f}_{ki}$  par cada subproblema:** la clase  $k_i$  se define como la clase positiva y la  $k_j$  se define como la clase negativa.

**La predicción está basada en cuántos modelos  $\widehat{f}_{ki}$  determinan que la observación  $x$ , pertenece a su clase:** cada clasificador vota si una observación pertenece o no a su clase y al final se cuentan todos los votos

# Clasificación

Clasificación de múltiples clases: uno vs uno





# Bibliografía

- **“Machine Learning and Pattern Recognition”** de Christopher Bishop.
- **“Elements of Statistical Learning”** Hastie, Tibshirani and Friedman
- **“Machine Learning: probabilistic Perspective”**, Kevin Murphy
- **“Machine Learning Yearnin”**, Andrew Ng

Ñapa:

- Kutner, M., C. Nachtsheim, J. Neter, and W. Li. 2005. *Applied Linear Statistical Models*. Fifth. McGraw Hill/Irwin.
- Montgomery, E. & Vining, D. & Peck. 2006. *Introducción Al Análisis de Regresión Lineal*. 3ed ed. México: Cecs.
- Link: [https://fhernanb.github.io/libro\\_regresion/rls.html](https://fhernanb.github.io/libro_regresion/rls.html)

# Gracias!!!

[ciencias.medellin.unal.edu.co](http://ciencias.medellin.unal.edu.co)

*Facultad de Ciencias  
Sede Medellín*



UNIVERSIDAD  
**NACIONAL**  
DE COLOMBIA