

# MÉTODOS SUPERVISADOS

Jorge Bedoya

# Métodos supervisados

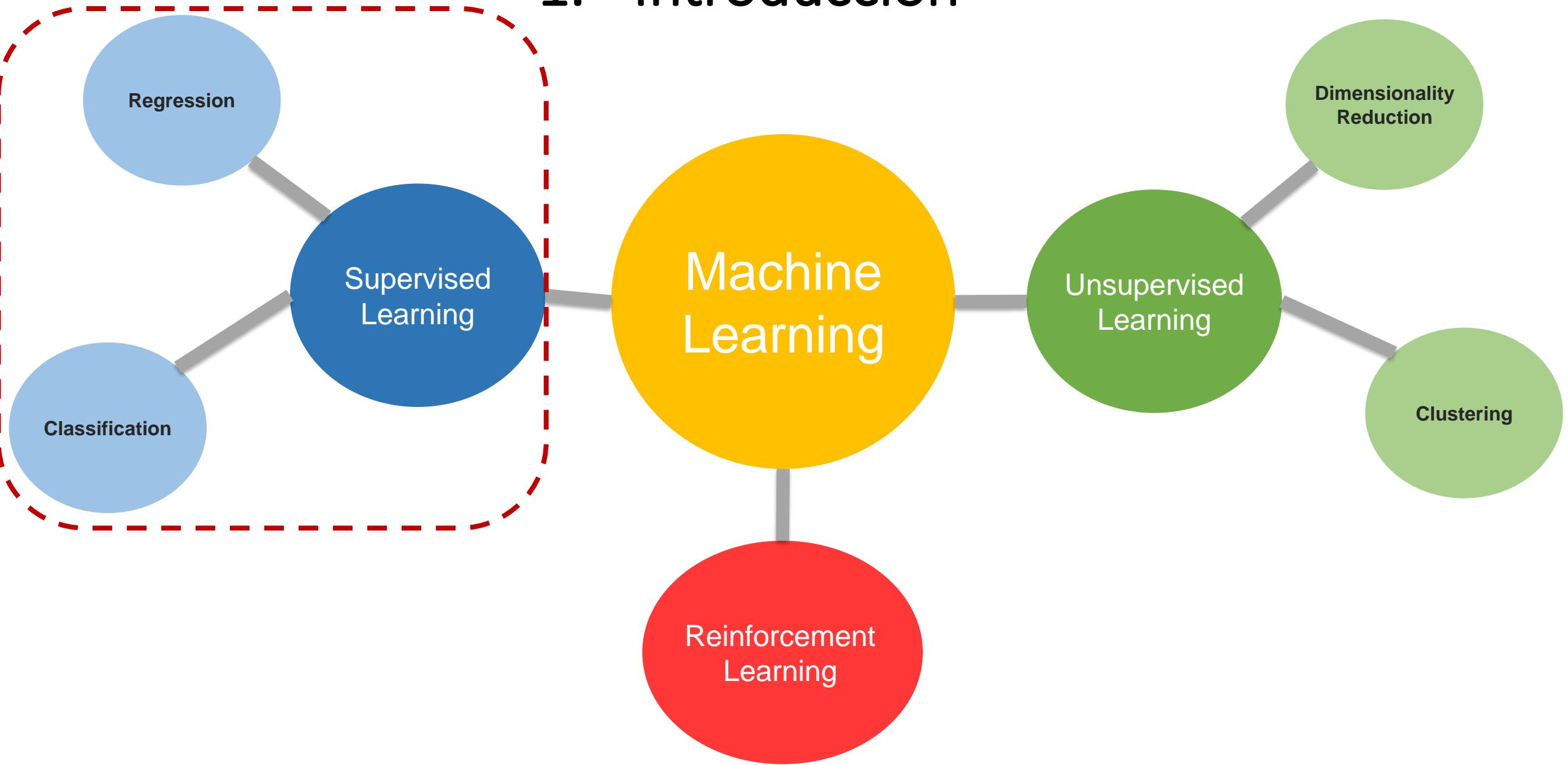
Temas a tratar:

1. Modelado supervisado
  2. Preparación de datos
  3. Sobreajuste y regularización
  4. Evaluación
  5. Técnicas
- Above the number 5, there is a blue curly brace that groups all the items from 1 to 7.
1. Regresión Lineal
  2. Regresión logística
  3. Clasificador Bayesiano
  4. K-NN
  5. Máquinas de Soporte Vectorial
  6. Árboles, Bagging + Random Forest.
  7. Boosting: Adaboost y Gradient boosting

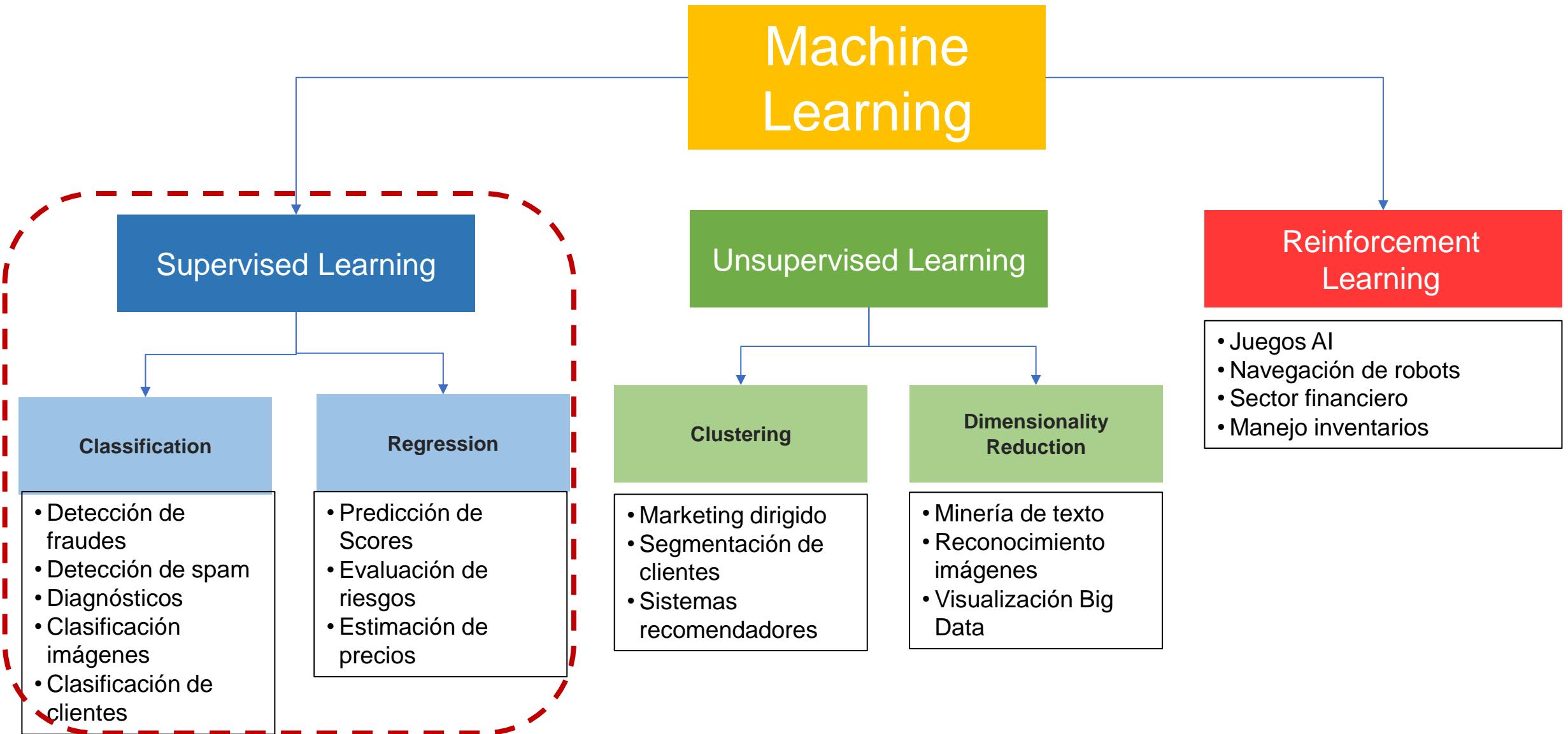


# 1. Modelado Supervisado

# 1. Introducción



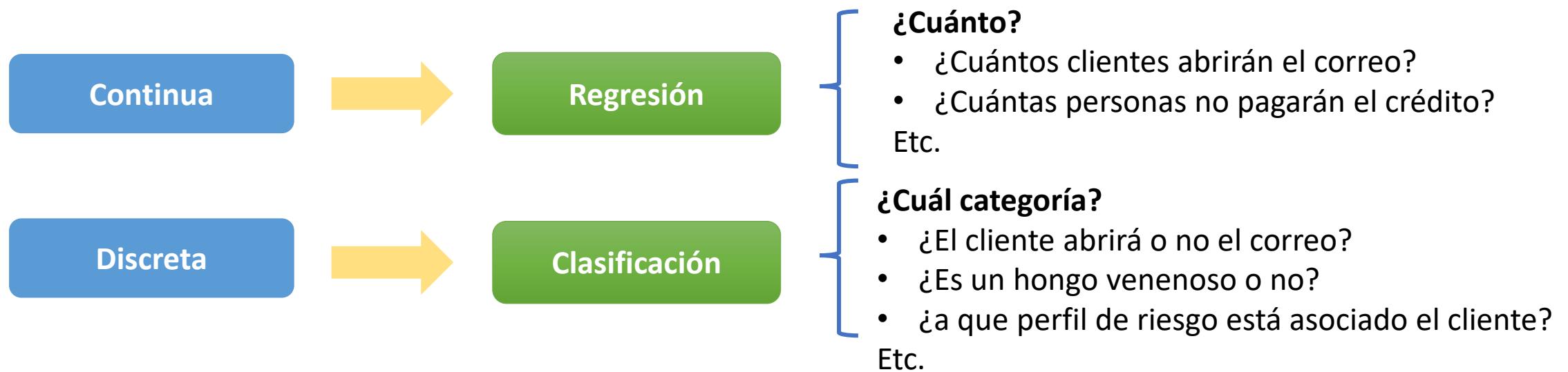
# 1. Introducción



# 1. Modelado supervisado

# 1. Modelado supervisado

- Existe una variable de salida o una variable a predecir (para uno o varios ejemplos)
- ¿Cómo es la variable de salida?



# 1. Modelamiento Predictivo: Proceso

➤ Intensión o interés de aprendizaje:

**Un banco quiere averiguar cuáles de sus clientes probablemente estarán en mora en el pago de sus deudas.**

➤ Variable objetivo: **CategoríaCliente?: Bueno/Malo** (Clasificación)

1. Con base en datos anteriores se compila un dataset de entrenamiento que contiene ejemplos etiquetados de clientes que han estado en mora con sus pagos.
2. Estos datos de entrenamiento alimentan un algoritmo de clasificación que es usado para entender quiénes son los clientes “bueno”(s) y “malo”(s).
3. Finalmente se ingresan datos nuevos sin etiquetar de los clientes con el fin de averiguar si un determinado cliente pertenece a la categoría de morosidad.

## 2. Preparación de datos

## 2. Preparación de datos

### ✓ Gestión de características categóricas

La mayor parte de los algoritmos exigen valores numéricos

¿cómo convertir estos valores en números?

#### Enfoques

- Label Encoding
- One Hot Encoding.

## 2. Preparación de datos

### Label Encoding

Identificar los distintos valores existentes y sustituir cada uno de ellos por un número

**Ejemplo:**

Nombre	Ciudad
Juan	M
Pedro	B
Maria	M
Isabel	C
Camilo	B



Nombre	Ciudad
Juan	0
Pedro	1
Maria	0
Isabel	3
Camilo	1

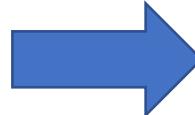
## 2. Preparación de datos

### One Hot Encoding

Crea una columna para cada valor distinto que existe en la característica que estamos codificando y, para cada registro, marcar con un 1 la columna a la que pertenezca dicho registro y dejar las demás con 0.

#### Ejemplo:

Nombre	Ciudad
Juan	M
Pedro	B
Maria	M
Isabel	C
Camilo	B



Nombre	Ciudad_M	Ciudad_B	Ciudad_C
Juan	1	0	0
Pedro	0	1	0
Maria	1	0	0
Isabel	0	0	1
Camilo	0	1	0

## 2. Preparación de datos

### ✓ Normalización y estandarización

Homogenizadores:

- MinMaxScaler
- Normalización (Normalizer)
- Estandarizador

<https://www.youtube.com/watch?v=-VuR14Qyl7E>

[https://www.cienciadedatos.net/documentos/py06\\_machine\\_learning\\_python\\_scikitlearn.html](https://www.cienciadedatos.net/documentos/py06_machine_learning_python_scikitlearn.html)

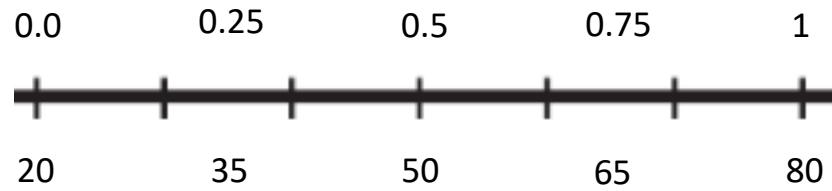
## 2. Preparación de datos

### ✓ MinMaxScaler

Toma los valores mínimos y máximos como referencia y luego asigna los valores en un rango de [0, 1] de acuerdo con su escalamiento

#### Ejemplo:

Nombre	Edad	Edad escalada
Juan	80	1
Pedro	50	0,5
Maria	35	0,25
Isabel	65	0,75
Camilo	20	0



## 2. Preparación de datos

### ✓ Normalización (Normalizer)

Utiliza la norma de un vector, de manera que cada valor de los datos se divide por la norma.

$$NormaL1(x_i) = \frac{x_i}{\sqrt{x_1^2 + x_2^2 + \dots + x_n^2}}$$

Ejemplo:

Nombre	Edad	Normalizada
Juan	80	0.66
Pedro	50	0.41
Maria	35	0.29
Isabel	65	0.54
Camilo	20	0.16

Notas:

- En sklearn debe transponerse (datos.T)
- Comúnmente genera valores muy pequeños

## 2. Preparación de datos



### Estandarizador

Consiste en cambiar la distribución de los datos para que tengan una media = 0 y una desviación = 1

$$\text{Estandarizado } (x_i) = \frac{x_i - \text{media}_i}{\text{std}}$$

$$\text{EstRobusto } (x_i) = \frac{x_i - \text{RangoIntencuartílico}_i}{\text{std}}$$

#### Ejemplo:

Nombre	Edad	Estandarizada	EstRobusto
Juan	80	1.26	2.11
Pedro	50	0.00	0.84
Maria	35	-0.63	0.21
Isabel	65	0.63	1.48
Camilo	20	-1.26	-0.42

EstRobusto facilita la exclusión de datos atípicos

$$\text{Media} = 50$$

$$\text{Std} = 23.72$$

$$\text{IQR} = Q_3 - Q_1 = 30$$

## 2. Preparación de datos

Reducción de la dimensionalidad:

✓ **Selección de características**

- Selección hacia adelante
- Eliminación hacia atrás
- PCA
- RFE
- Inducción del árbol de decisión

✓ **Extracción de características**

Discretización de los datos:

✓ **Binning**

✓ **Agrupamiento**



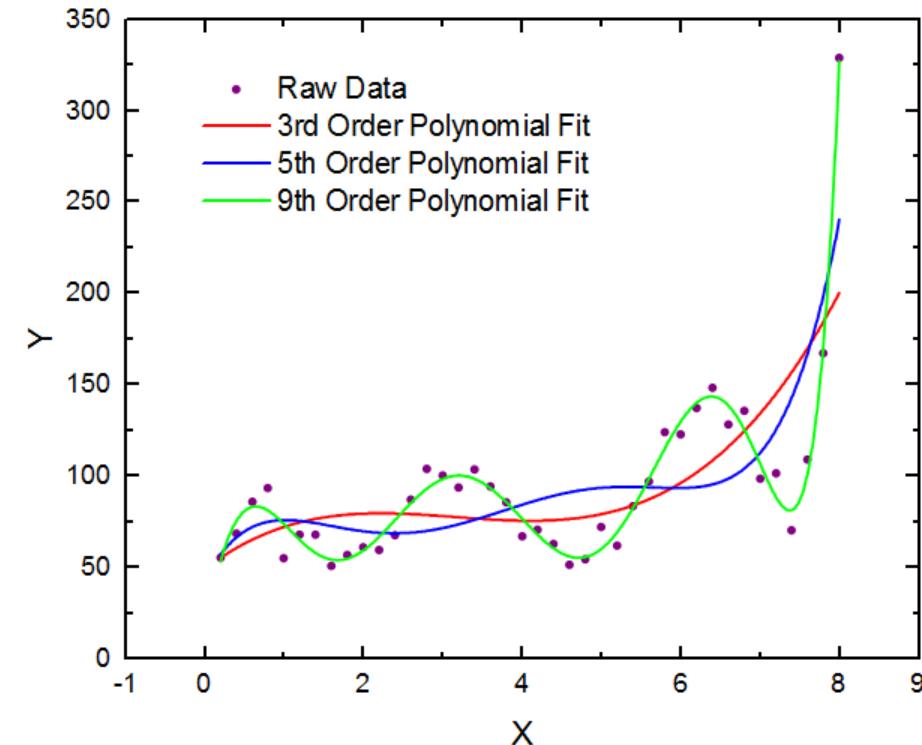
# 3. Sobreajuste y regularización

## ✓ Sobreajuste (*Overfitting*)

$$h(X) = \theta_0 + \theta_1 X^1 + \theta_2 X^2 + \theta_3 X^3 \dots \theta_n X^n$$

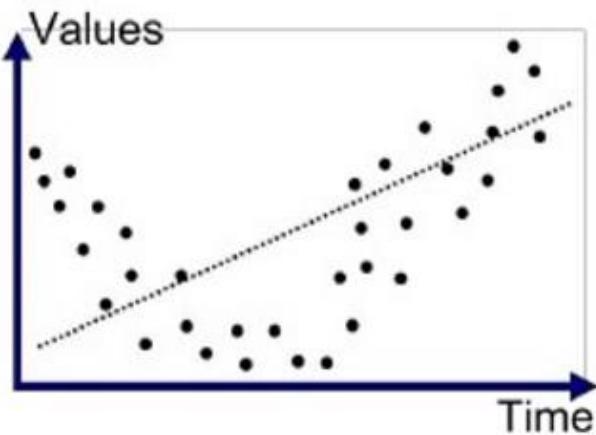
Where 'n' is the degree of the equation  
"θ" is the slope

Si tenemos muchas tetas significa que hay muchas pendientes y, por lo tanto, nuestra línea puede cambiar de dirección de muchas maneras diferentes.

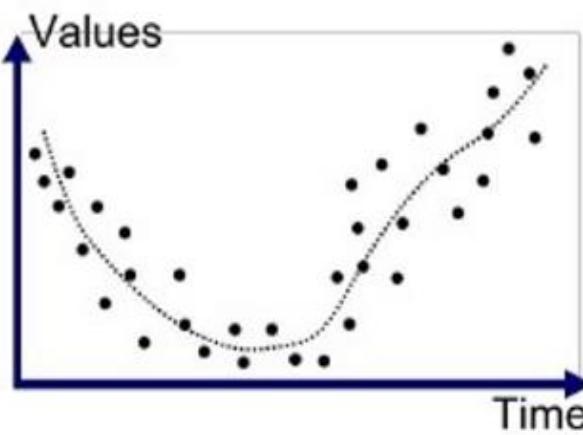


### 3. Sobreajuste y regularización

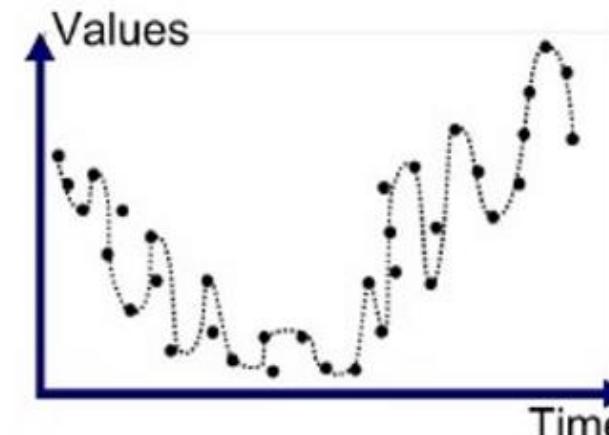
#### ✓ Sobreajuste (*Overfitting*)



Underfitted



Good Fit/Robust



Overfitted

# 3. Sobreajuste y regularización



## Regularización

La regularización consiste en añadir una penalización a la función de coste. Esta penalización produce modelos más simples que generalizan mejor.

- Lasso =  $L1$ ,
- Ridge =  $L2$
- *ElasticNet* = combina Lasso y Ridge.

# 3. Sobreajuste y regularización

## ✓ Regularización Lasso (L1)

La complejidad  $C$  se mide como la media del valor absoluto de los coeficientes del modelo.

$$C = \frac{1}{N} \sum_{j=1}^N |w_i|$$

### ¿Cuándo usar L1?

- Se sospecha que varios de los atributos de entrada son irrelevantes.
- Lasso funciona mejor cuando los atributos no están muy correlacionados entre ellos.



# 3. Sobreajuste y regularización

## ✓ Regularización Ridge (L2)

La complejidad  $C$  se mide como la media del cuadrado de los coeficientes del modelo.

$$C = \frac{1}{2N} \sum_{j=1}^N w_i^2$$

### ¿Cuándo usar L2?

- Se sospecha que varios de los atributos de entrada estén correlacionados entre ellos.
- Ridge funciona mejor cuando la mayoría de los atributos son relevantes.



# 3. Sobreajuste y regularización

## ✓ Regularización ElasticNet (L1 y L2)

ElasticNet combina las regularizaciones L1 y L2. Con el parámetro  $r$  podemos indicar que importancia relativa tienen Lasso y Ridge respectivamente.

$$C = r \cdot \text{Lasso} + (1 - r) \cdot \text{Ridge}$$

\* con  $r$  se indica la importancia relativa que tiene Lasso y Ridge respectivamente

## ¿Cuándo usar ElasticNet?

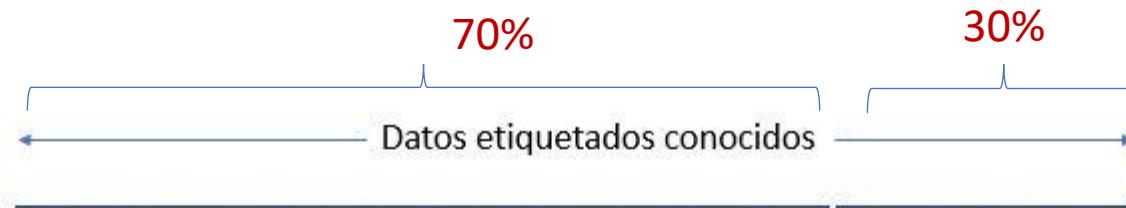
- Cuando se tiene un gran número de atributos: Algunos de ellos serán irrelevantes y otros estarán correlacionados entre ellos..



## 4. Evaluación

# 4. Modelamiento Predictivo: Método de retención (*holdout method*)

- Es un método que consiste separar los datos en datos de entrenamiento (*training*) y prueba (*test*)



Datos de entrenamiento

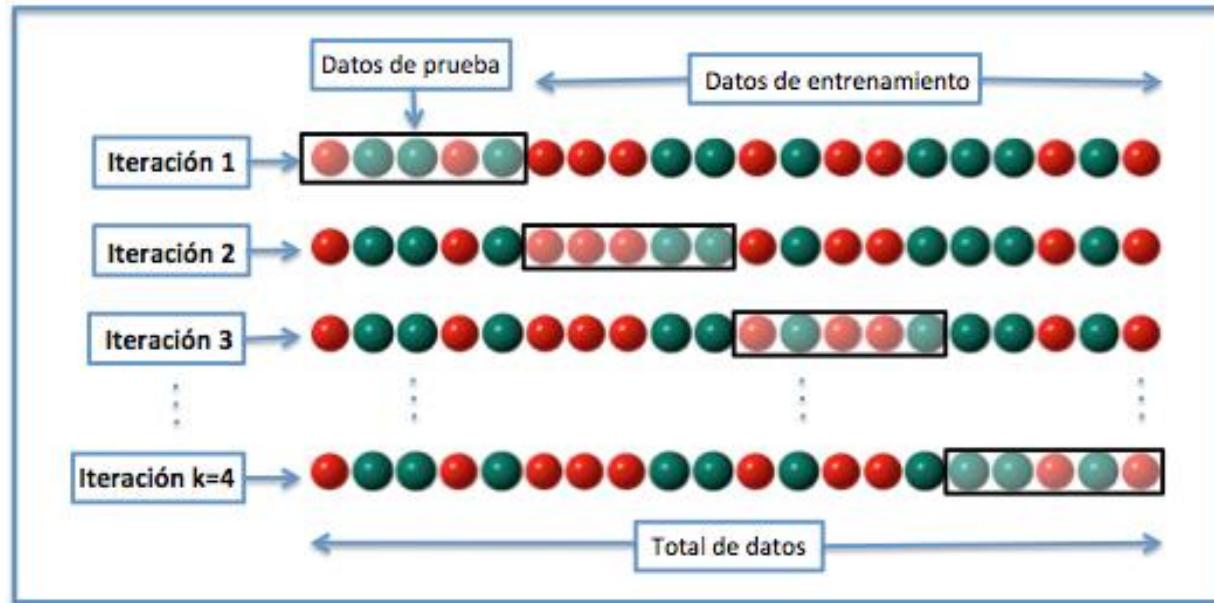
Datos de validación



Datos de entrenamiento

Datos de validación Datos de prueba

# 4. Modelamiento Predictivo: Validación cruzada(*cross validation*)



- Los datos de muestra se dividen en K subconjuntos (k-folds).
- Uno de los subconjuntos se utiliza como datos de prueba y el resto ( $K-1$ ) como datos de entrenamiento.
- El proceso de validación cruzada es repetido durante k iteraciones con cada uno de los posibles subconjuntos de datos de prueba.
- Finalmente se realiza la media aritmética de los resultados de cada iteración para obtener un único resultado.

# 4. Evaluación: Matriz de confusión

Matriz de confusión: Clasificación

		PREDICCIÓN	
		Positivo	Negativo
REAL	Positivo	VP	FN
	Negativo	FP	VN

- **VP** es la cantidad de *positivos* que fueron *clasificados correctamente* como positivos por el modelo.
- **VN** es la cantidad de *negativos* que fueron *clasificados correctamente* como negativos por el modelo.
- **FN** es la cantidad de *positivos* que fueron *clasificados incorrectamente* como negativos.
- **FP** es la cantidad de *negativos* que fueron *clasificados incorrectamente* como positivos.

# 4. Evaluación: Matriz de confusión

Matriz de confusión: Clasificación

		PREDICCIÓN	
		Positivo	Negativo
REAL	Positivo	VP	FN
	Negativo	FP	VN

Exactitud (*Accuracy*): proporción de instancias identificadas correctamente entre todas las instancias

# 4. Evaluación: Matriz de confusión

Matriz de confusión: Clasificación

		PREDICCIÓN	
		Positivo	Negativo
REAL	Positivo	VP	FN
	Negativo	FP	VN

Tasa de errores: proporción de instancias identificadas incorrectamente entre todas las instancias

# 4. Evaluación: Matriz de confusión

Matriz de confusión: Clasificación

		PREDICCIÓN	
		Positivo	Negativo
REAL	Positivo	VP	FN
	Negativo	FP	VN

The diagram illustrates a 2x2 confusion matrix with four cells: True Positive (VP) in the top-left (green oval), False Positive (FP) in the bottom-left (blue oval), False Negative (FN) in the top-right (green oval), and True Negative (VN) in the bottom-right (blue oval).

TPR  
FPR

$$\text{Sensibilidad} = \frac{VP}{\text{Total Positivos}} \quad \text{Recall}$$

$$\text{Especificidad} = \frac{VN}{\text{Total Negativos}}$$

# 4. Evaluación: Matriz de confusión

Matriz de confusión: Clasificación

		PREDICCIÓN	
		Positivo	Negativo
REAL	Positivo	VP	FN
	Negativo	FP	VN

$$\text{Precisión} = \frac{VP}{\text{Total clasificados positivos}}$$

$$\text{VPN}^* = \frac{VN}{\text{Total clasificados negativos}}$$

\*Valor de precisión negativo

# 4. Evaluación: Matriz de confusión

Matriz de confusión: Clasificación

## F1 Score

Combina la **precisión** y el **recall** para proporcionar una medida única de su rendimiento.

Es especialmente útil cuando tienes un conjunto de datos desbalanceado

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{\text{TP}}{\text{TP} + \frac{1}{2}(\text{FP} + \text{FN})}$$

TP = number of true positives

FP = number of false positives

FN = number of false negatives

# 4. Evaluación: Curva ROC (Receiver Operating Characteristic)

ROC: representación gráfica de la **sensibilidad** frente a la **especificidad** para un sistema claseificador binario

		PREDICCIÓN	
		Positivo	Negativo
REAL	Positivo	VP 	FN 
	Negativo	FP 	VN 

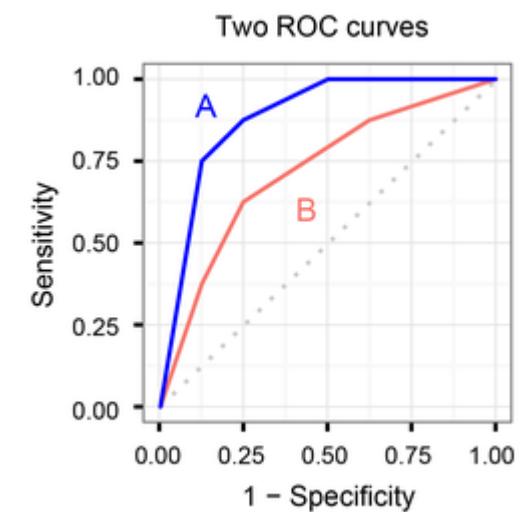
TPR  
**vs.**  
FPR



$$\text{Sensibilidad} = \frac{VP}{\text{Total Positivos}}$$

**vs.**

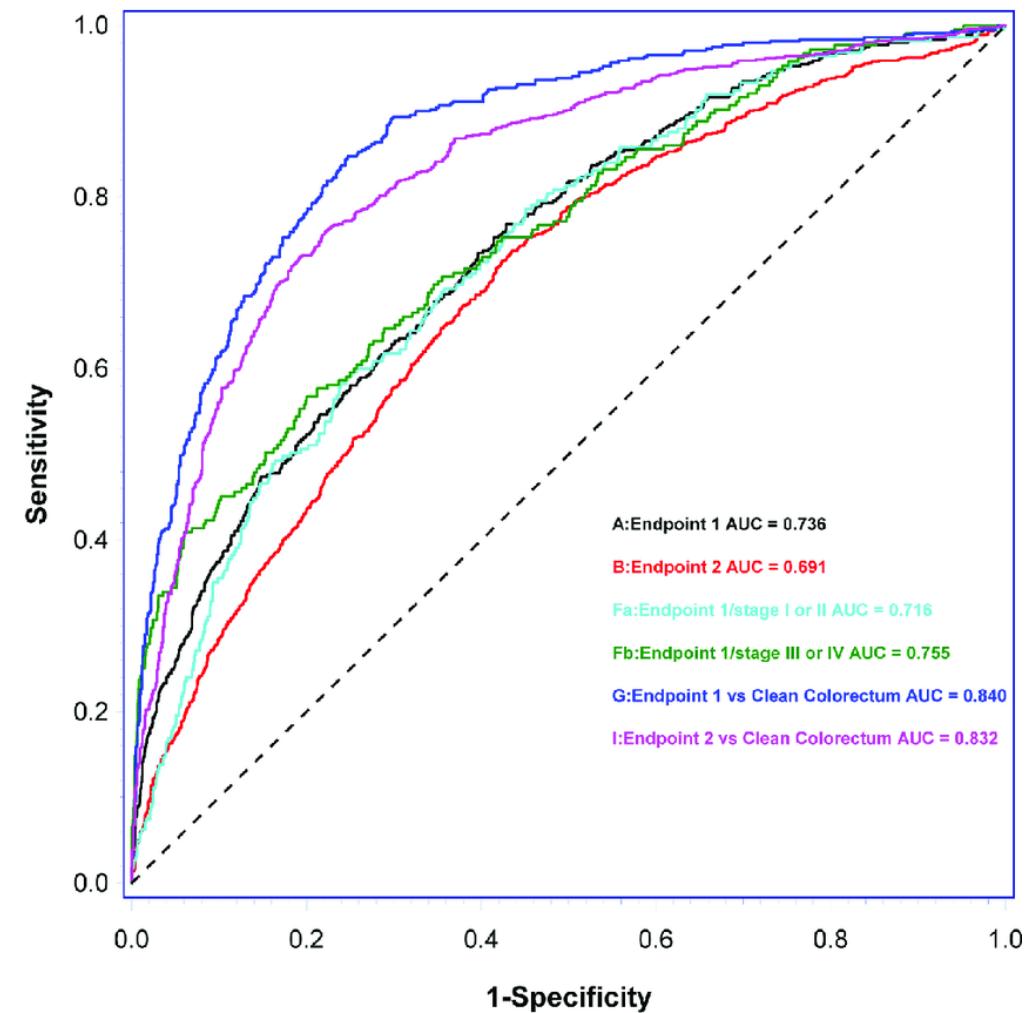
$$\text{Especificidad} = \frac{VN}{\text{Total Negativos}}$$



# 4. Evaluación: Curva ROC (Receiver Operating Characteristic)

AUC: Área bajo la curva

- [0.5] → Test Aleatorio
- [0.5 0.6) → Test malo
- [0.6 0.75) → Test regular
- [0.75 0.9) → Test bueno
- [0.9 0.97) → Test muy bueno
- [0.97 1) → Test excelente

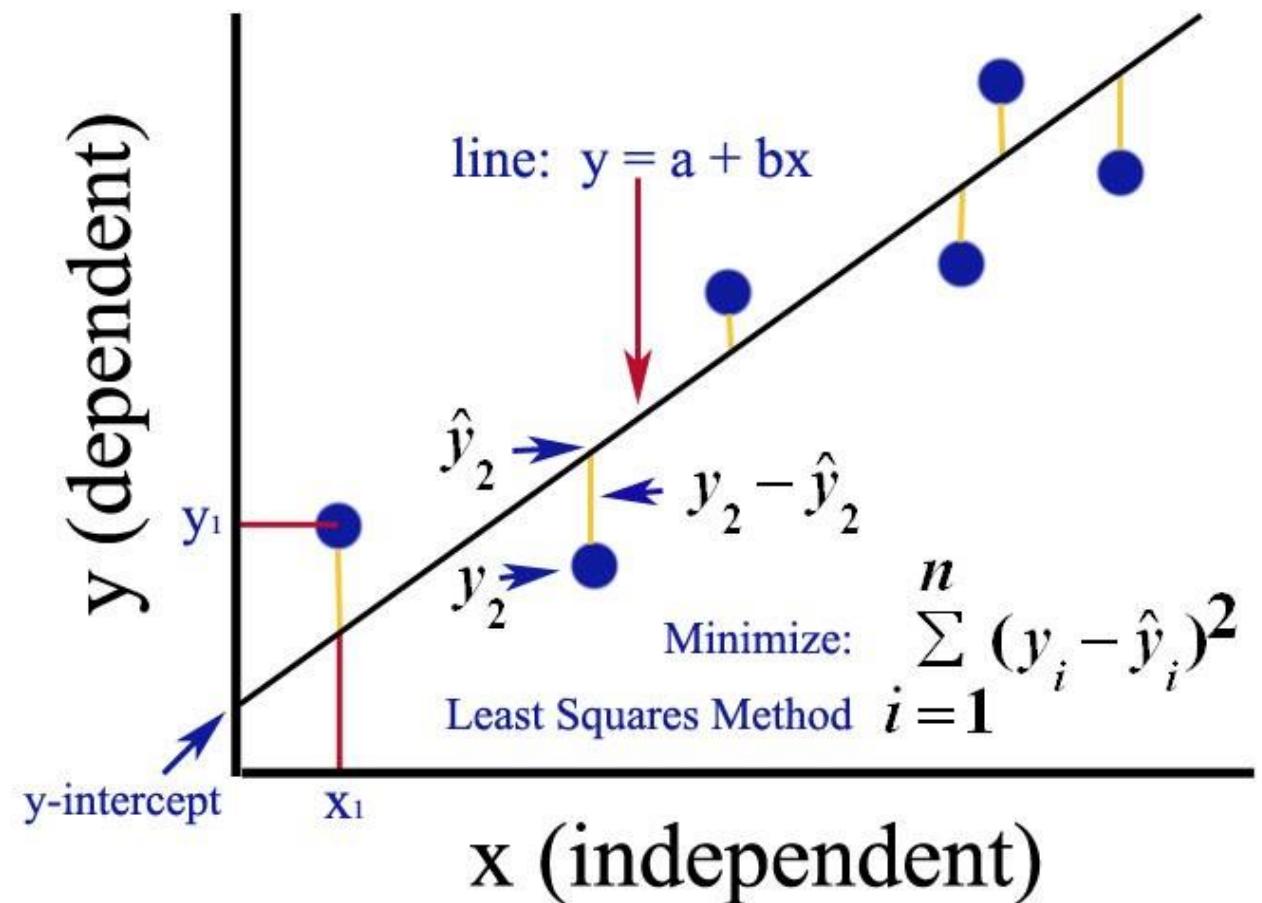


# 4. Evaluación: Funciones de valor residual

Funciones de valor residual (Regresión):  
Diferencia entre el valor predicho  
(o *score*) y el valor real.

Error medio cuadrado (Mean squared error) o MSE

$$MSE = \frac{1}{N} \sum_{i=1}^N (f(x_i) - y_i)^2$$



# 4. Evaluación: Funciones de valor residual

## Funciones de valor residual (Regresión)

Error medio cuadrado (Mean squared error) o MSE

$$MSE = \frac{1}{N} \sum_{i=1}^N (f(x_i) - y_i)^2$$

Raíz del error cuadrático medio (Root mean squared error) o RMSE,

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (f(x_i) - y_i)^2}$$

Error absoluto promedio (Mean absolute error) o MAE,

$$MAE = \frac{1}{N} \sum_{i=1}^N |f(x_i) - y_i|$$

Error absoluto medio (Median absolute error)

$$\text{Median Absolute Error} = \text{Median}(|f(x_i) - y_i|)$$

# 4. Evaluación: Funciones de valor residual

Funciones de valor residual (Regresión): Ejemplo

$$MSE = \frac{1}{N} \sum_{i=1}^N (f(x_i) - y_i)^2$$

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (f(x_i) - y_i)^2}$$

$$MAE = \frac{1}{N} \sum_{i=1}^N |f(x_i) - y_i|$$

Mean Square Error = 0.9553

Root Mean Square Error = 0.9774

Mean Absolute Error = 0.7747

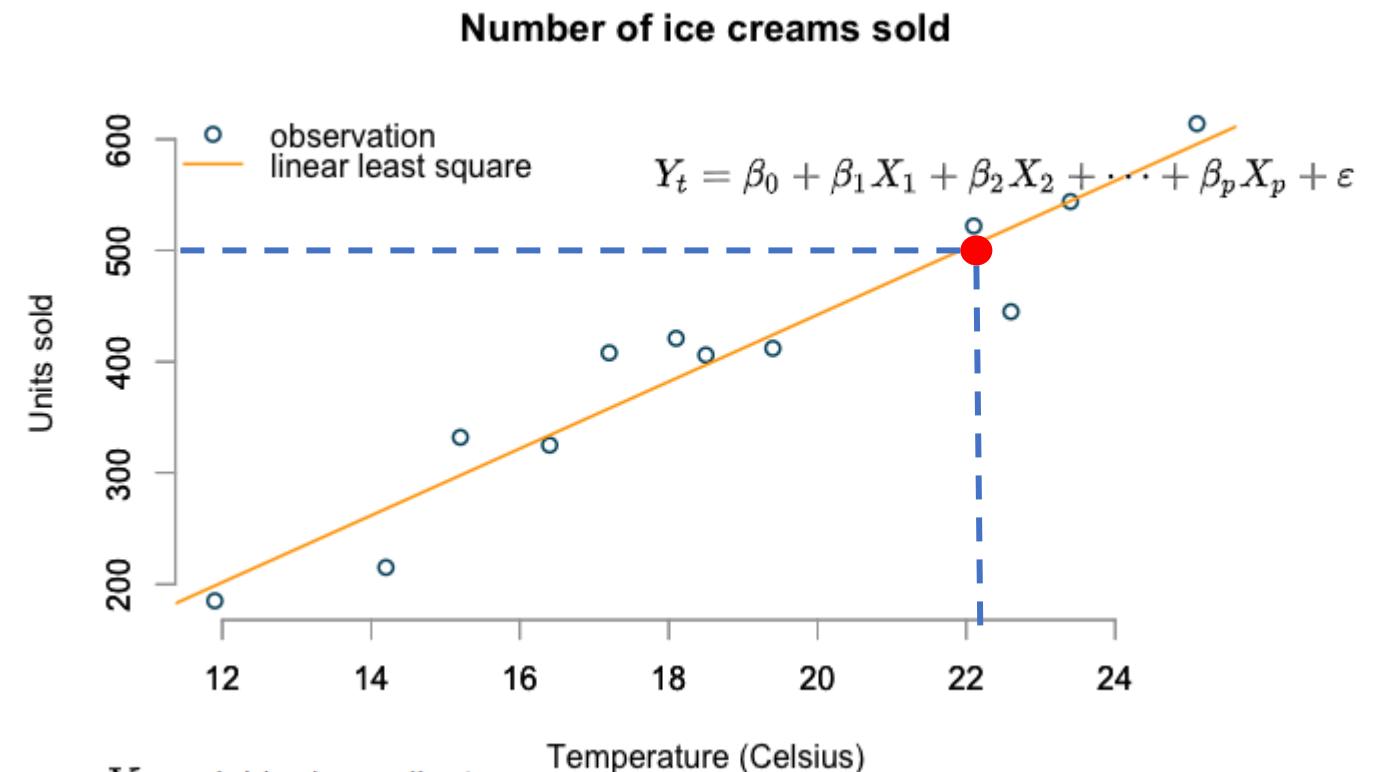
Median Absolute Error = 0.6252

$$\text{Median Absolute Error} = \text{Median}(|f(x_i) - y_i|)$$

# 5. Técnicas

# Técnicas: 1. Regresión lineal

1. Expresar su relación lineal.
2. Hacer una estimación de los parámetros del modelo por medio del ajuste del modelo.



Y<sub>t</sub>: variable dependiente,

X<sub>1</sub>, X<sub>2</sub>, …, X<sub>p</sub>: variables explicativas.

$\beta_0, \beta_1, \beta_2, \dots, \beta_p$ : parámetros, miden la influencia que las variables explicativas tienen sobre el regrediendo.

$\varepsilon$  es la perturbación aleatoria

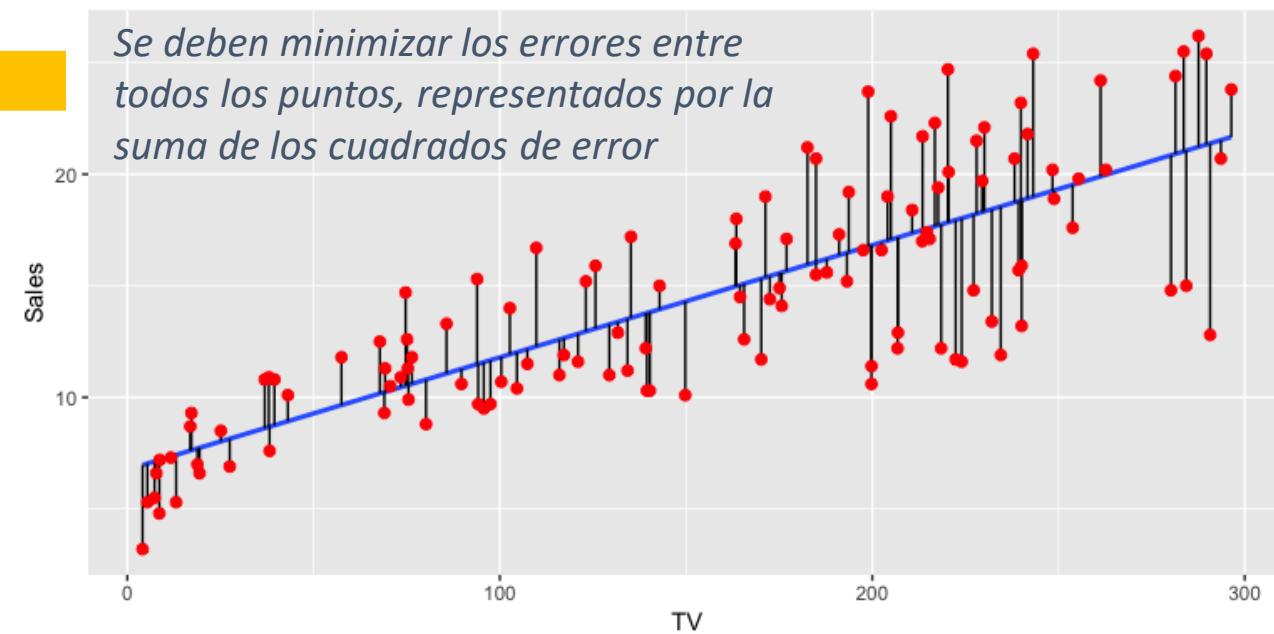
# Técnicas: 1. Regresión lineal simple

- Técnica estadística para predecir los valores de una variable **continua dependiente** con base en los valores de una variable **independiente**.

$$MSE = \frac{1}{N} \sum_{i=1}^N (f(x_i) - y_i)^2$$



- Paramétrico
- Asume que existe una correlación lineal entre la variable de respuesta y la variable explicativa.
- Rápido y requiere pocos recursos

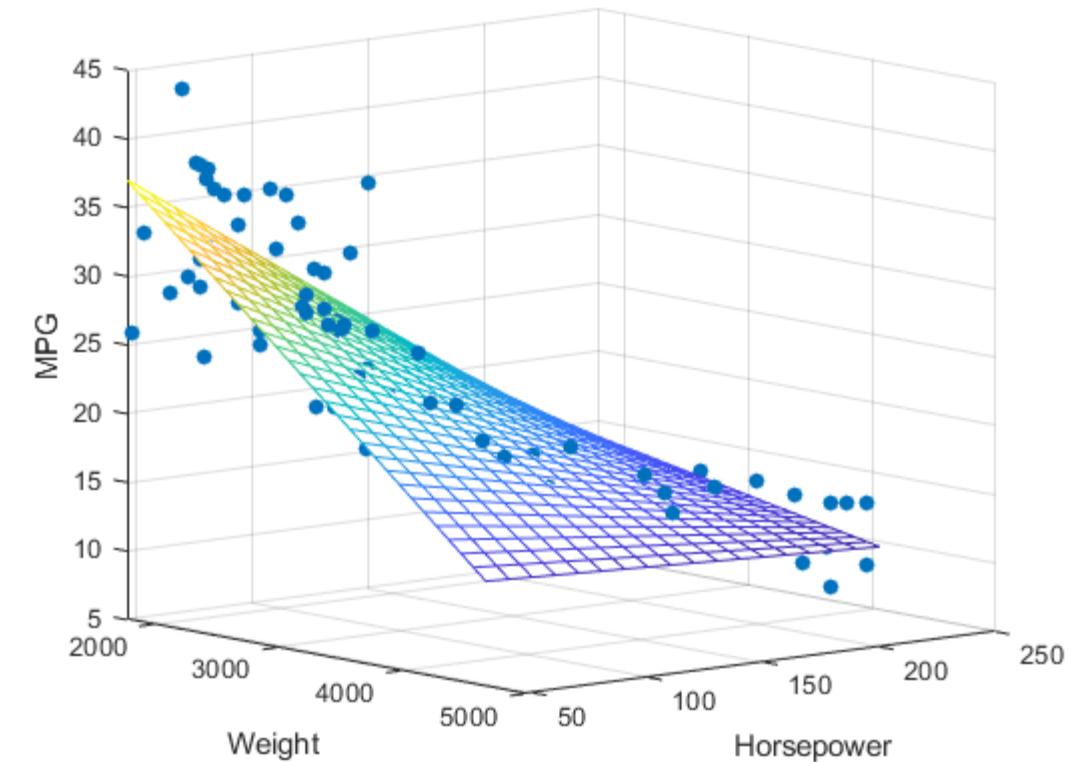


# Técnicas: 1. Regresión lineal múltiple

- Se refiere a utilizar más de dos variables explicativas a la vez para predecir la variable de respuesta

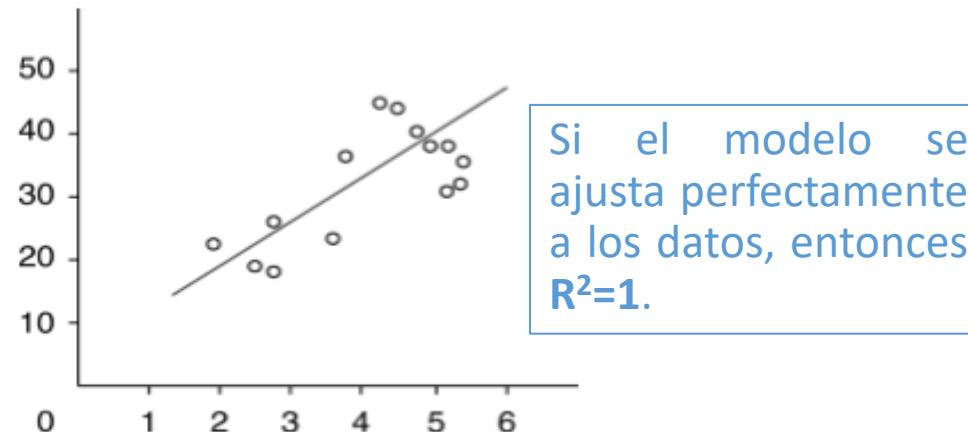
$$MSE = \frac{1}{N} \sum_{i=1}^N (f(x_i) - y_i)^2$$

- Se recomienda hallar la correspondiente relevancia de cada variable explicativa
- Comprobar el *Coeficiente de determinación R<sup>2</sup>*



# Técnicas: 1. Regresión lineal múltiple

- El **coeficiente de determinación  $R^2$**  corresponde al porcentaje de variación en la variable de respuesta determinado por la variable explicativa, comprendiendo valores que varían entre 0 y 1



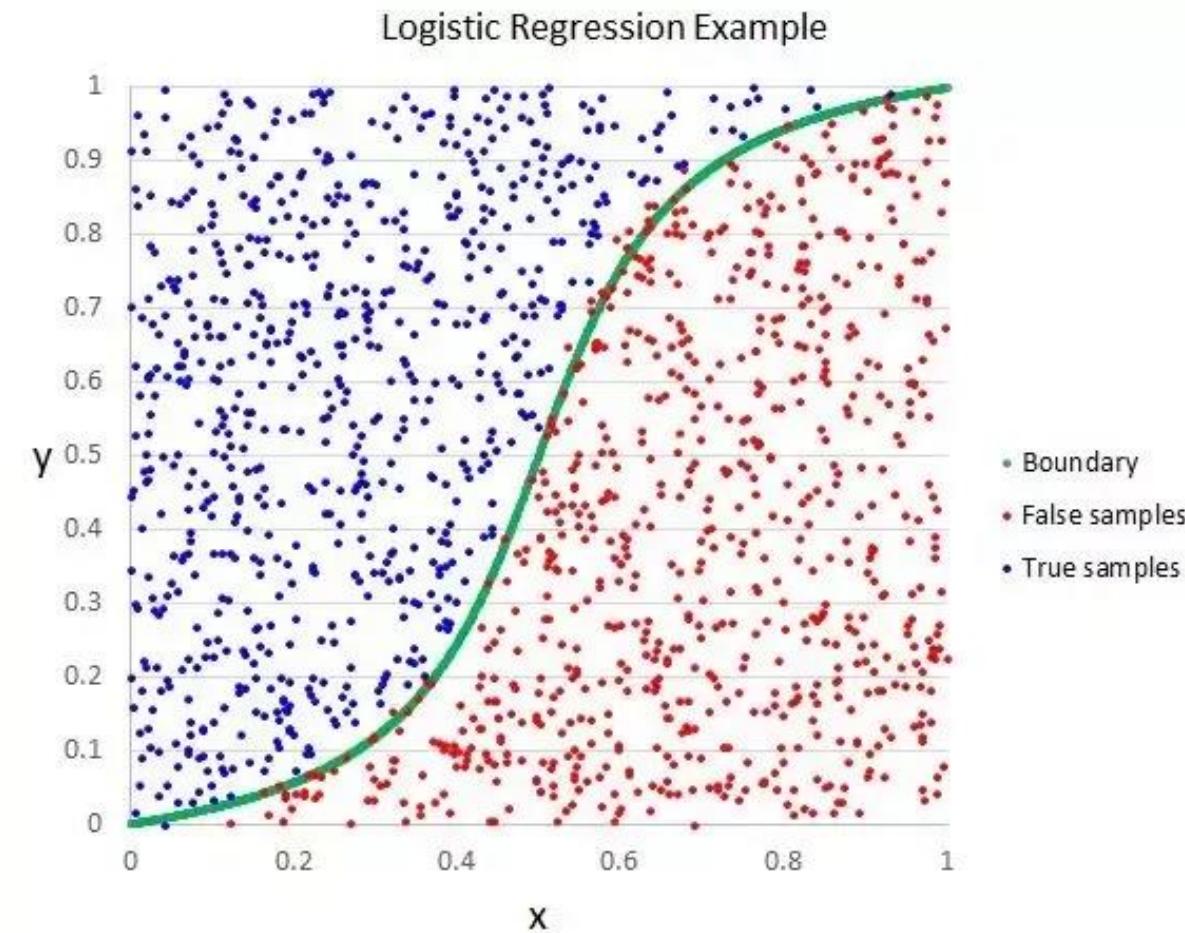
# Técnicas: 2. Regresión logística

- Variación que se usa para **clasifcación binaria** (entre 0 y 1) usando la siguiente función:

$$Y_t = \ln \frac{p}{1 - p}$$

- No es práctico usar una línea recta para estimar probabilidades (que varían entre 0 y 1)
- Establece la probabilidad de que la instancia pertenezca a la clase objetivo\*

\*NO indica que este sea el valor de la variable de respuesta, tal y como es el caso de la regresión lineal



# Técnicas: 2. Regresión logística

- Cuál es la probabilidad de éxito cuando el valor de la variable predictora es  $x$ .

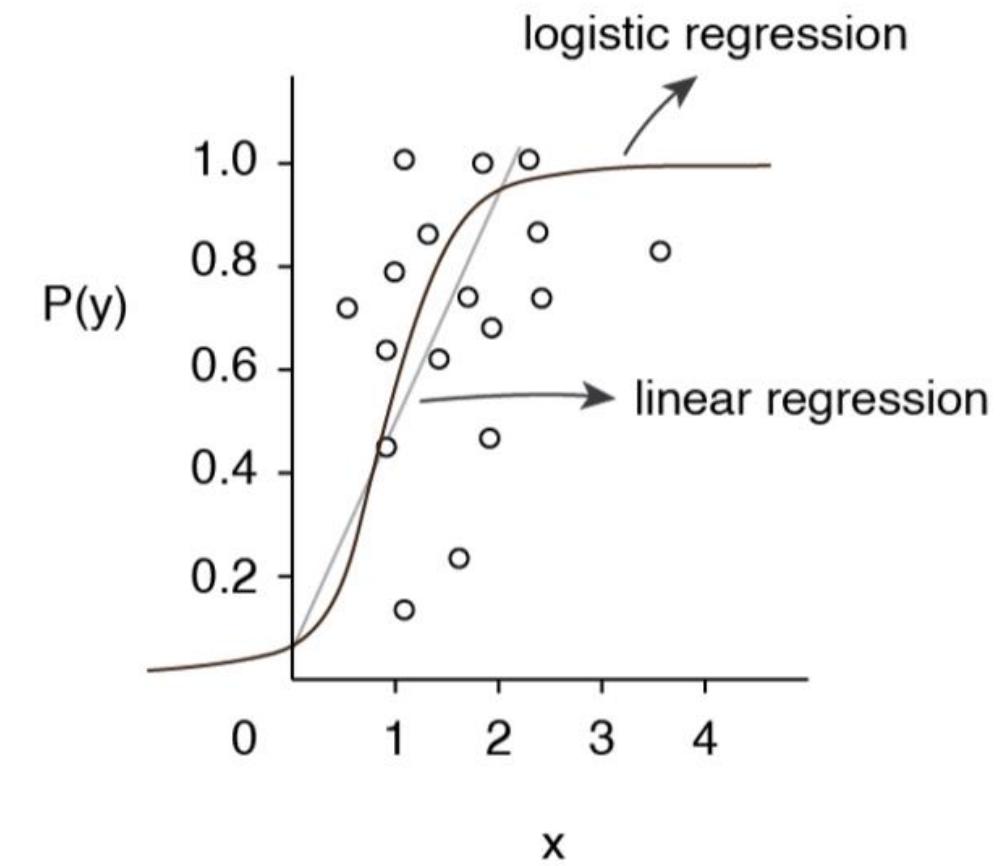
$$Y_t = \beta_0 + \beta_1 X$$

$$Y_t = \ln \frac{p}{1-p}$$

$$\ln \frac{p}{1-p} = \beta_0 + \beta_1 X$$

$$p = \frac{e^{\beta_0 + \beta_1 X}}{1 + e^{\beta_0 + \beta_1 X}}$$

Probability	Log-Odds
0.5	0.0
> 0.5	> 0.0
< 0.5	< 0.0



# Técnicas: 3. Clasificador Bayesiano

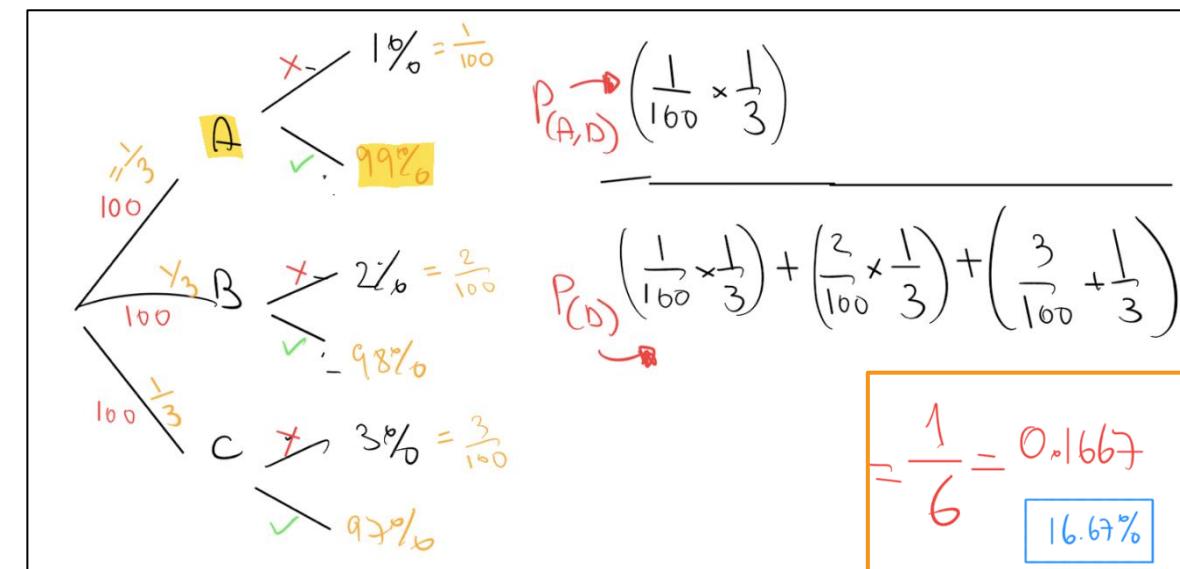
- Permite seleccionar una clase a partir de probabilidades condicionadas

## Ejemplo Bayes:

Las máquinas A, B y C fabrican piezas con un porcentaje de defectos de: 1% 2% y 3%, respectivamente.

Se mezclan 300 piezas, 100 de cada máquina, y se elige una pieza al azar, que resulta ser defectuosa.

¿Cuál es la probabilidad de que haya sido fabricada en la máquina A?



$$P(A_i|B) = \frac{P(B|A_i)P(A_i)}{P(B)}$$

# Técnicas: 3. Clasificador Bayesiano

- Vincula la probabilidad del suceso A dado el B con la probabilidad del suceso B dado A

$$P(A_i|B) = \frac{P(B|A_i)P(A_i)}{P(B)}$$

- Se basa en probabilidades
- Es capaz de tener en cuenta las características que parecen insignificantes (características independientes)
- Permite seleccionar las mejores instancias

- Poca información de falsos positivos y negativos
- Modelos eficientes y rápidos

Caso Monty Hall:

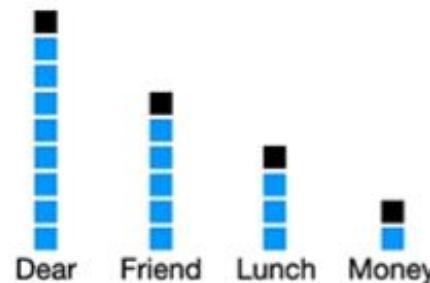
<https://estadisticaparatodos.es/taller/montyhall/montyhall.html>

# Técnicas: 3. Clasificador Bayesiano

- **Multinomial:** Características describen frecuencias discretas. Por ejemplo contar palabras
- **Gaussian:** predicciones sobre características normalmente distribuidas
- **Bernulli:** predicciones para características binarias Es adecuado para variables categóricas binarias o variables que se pueden convertir en binarias a partir de un umbral

# Técnicas: 3. Clasificador Bayesiano

## ➤ Multinomial

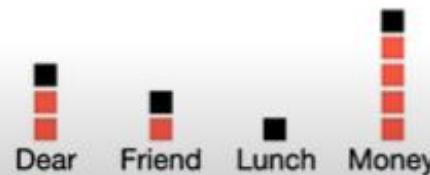


**Lunch Money Money Money Money**

$$\begin{aligned} p(\text{Dear} | N) &= 0.43 \\ p(\text{Friend} | N) &= 0.29 \\ p(\text{Lunch} | N) &= 0.19 \\ p(\text{Money} | N) &= 0.10 \end{aligned}$$

$$p(N) \times p(\text{Lunch} | N) \times p(\text{Money} | N)^4 = 0.00001$$

...we classify the message as **spam**.



$$p(S) \times p(\text{Lunch} | S) \times p(\text{Money} | S)^4 = 0.00122$$

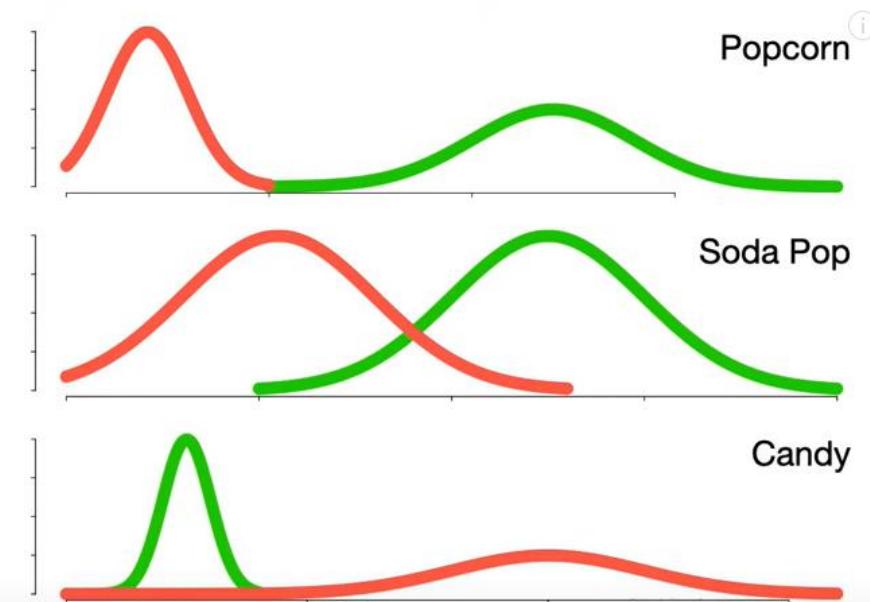
# Técnicas: 3. Clasificador Bayesiano

## ➤ Gaussian



Popcorn (grams)	Soda Pop (ml)	Candy (grams)
24.3	750.7	0.2
28.2	533.2	50.5
etc.	etc.	etc.

Popcorn (grams)	Soda Pop (ml)	Candy (grams)
2.1	120.5	90.7
4.8	110.9	102.3
etc.	etc.	etc.



**Gaussian Naive Bayes** is named after the **Gaussian** distributions that represent the data in the **Training Dataset**.



# Técnicas: 4. k-NN (*Nearest Neighbour*)

- Se basa en la intuición de que datos similares tendrán clases similares.

**¿Cómo se mide la similitud?**

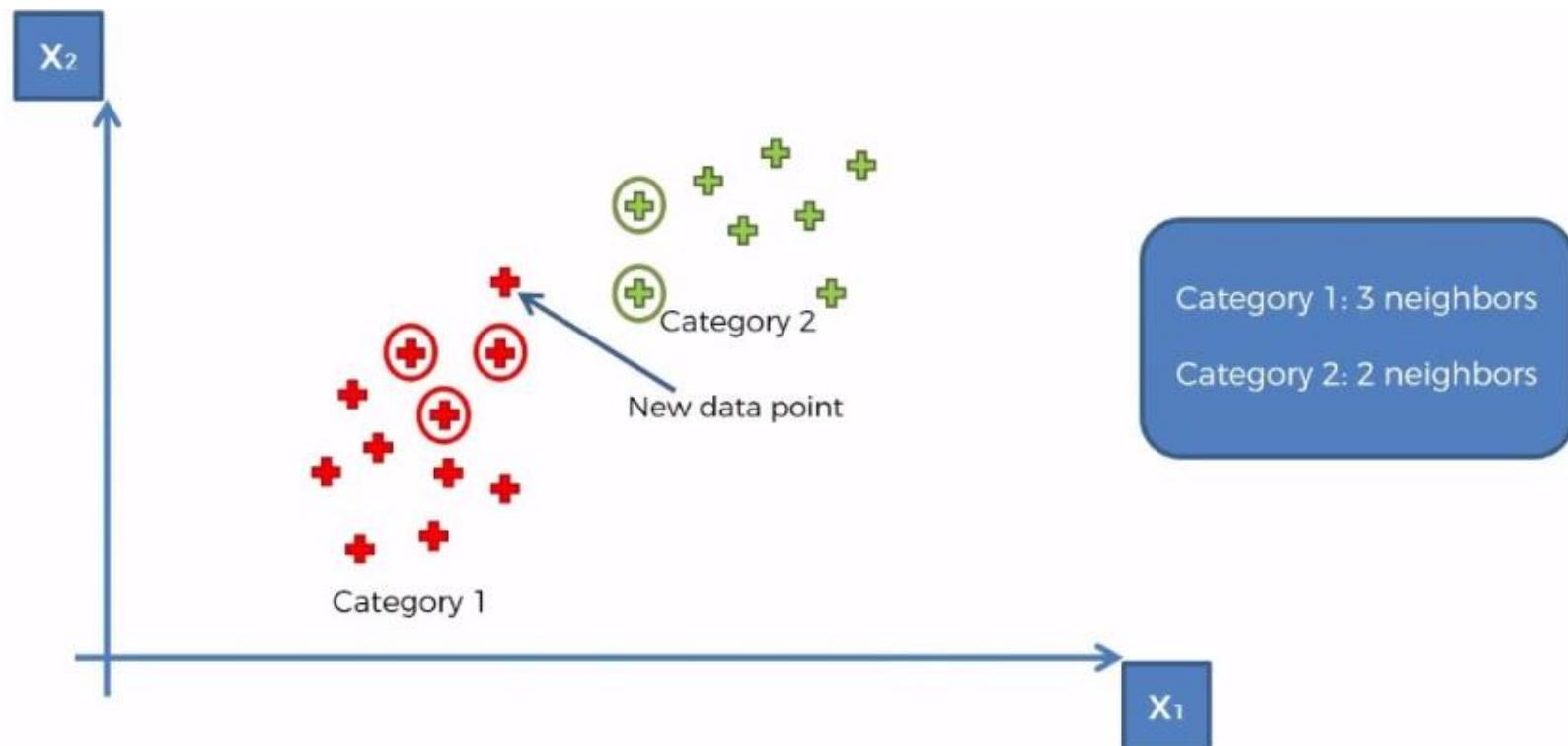


- No-paramétrico
- Lazy learning
- Buen desempeño en instancias difíciles de explicar
- Requiere gran cantidad de memoria
- Se ve afectado por datos atípicos

<b>Distancia Euclídea:</b> $\sqrt{\sum_{i=1}^n (x_i - y_i)^2}$	Valores Continuos (conveniente normalizar entre 0-1 antes)
<b>Distancia de Manhattan:</b> $\sum_{i=1}^n  x_i - y_i $	
<b>Distancia de Chebychev:</b> $\max_{i=1..n}  x_i - y_i $	
<b>Distancia del coseno:</b> <i>cada ejemplo es un vector y la distancia es el coseno del ángulo que forman</i>	Valores Continuos. No es necesario normalizar
<b>Distancias por Diferencia:</b> <i>ejemplo: if <math>x=y</math> then <math>D=0</math> else <math>D=1</math></i>	Valores Discretos
<b>Distancia de Edición:</b>	
<b>Distancias Específicas:</b> para los ejemplos complejos de CBR.	

# Técnicas: 4. k-NN (*Nearest Neighbour*)

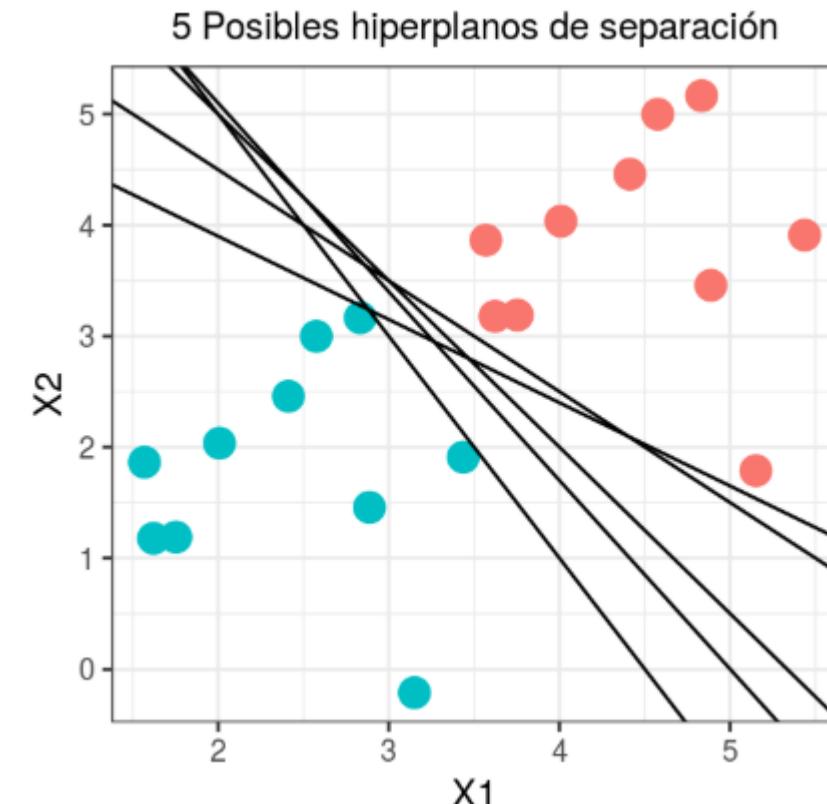
1. Se miran los  $k$ -casos más cercanos.
2. Se calcula la distancia media por clase o se asigna a la clase con más elementos.



El valor de  $k$  se suele determinar heurísticamente  $k = \sqrt{n}$  donde  $n$  es el número de ejemplos. (Es una opción con base teórica)

# Técnicas: 5. Máquinas de soporte vectorial (Support Vector Machines, SVMs)

- Buscar un hiperplano que separe lo mejor posible las clases.



<https://www.codificandobits.com/blog/maquinas-de-soporte-vectorial/>

[https://rpubs.com/Joaquin\\_AR/267926](https://rpubs.com/Joaquin_AR/267926)

# Técnicas: 5. Máquinas de soporte vectorial (Support Vector Machines, SVMs)

- Hiperplano óptimo de separación, que se corresponde con el hiperplano que se encuentra más alejado de todas las observaciones de entrenamiento.
- Para obtenerlo, se tiene que calcular la distancia perpendicular de cada observación a un determinado hiperplano. La menor de estas distancias (conocida como margen) determina qué tan alejado está el hiperplano de las observaciones de entrenamiento.

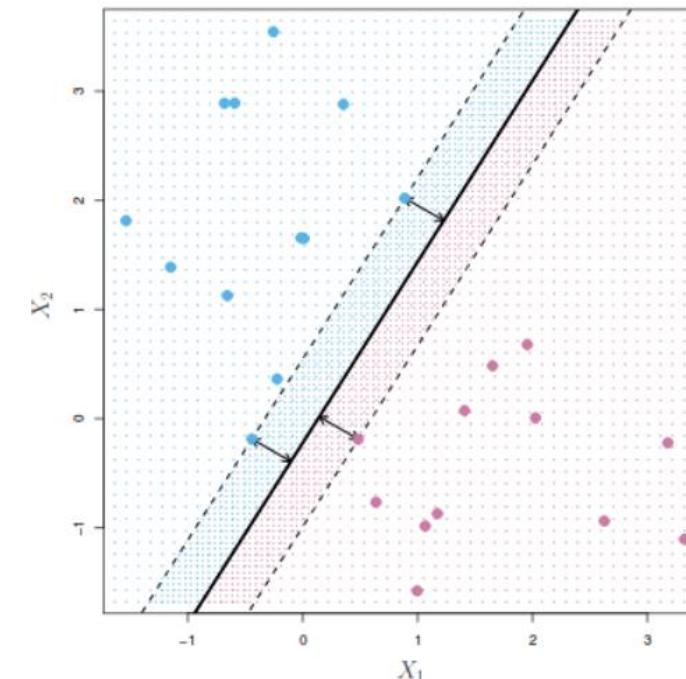
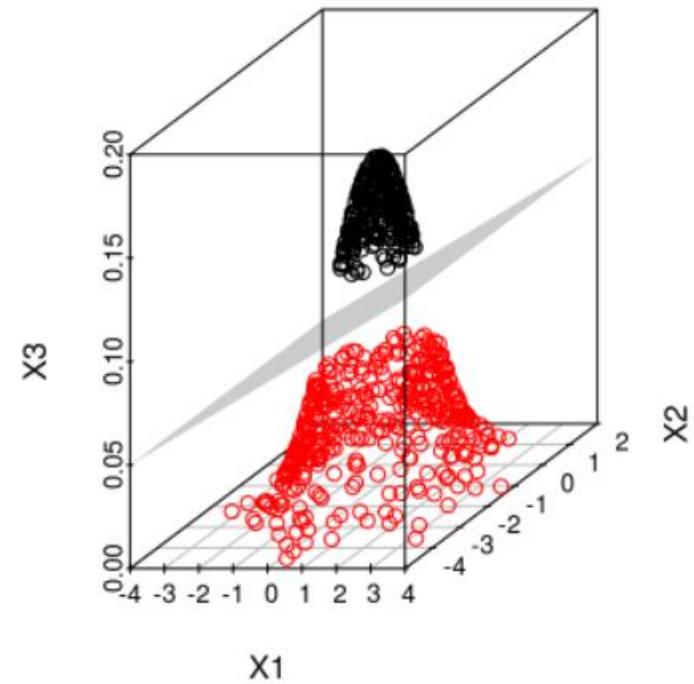
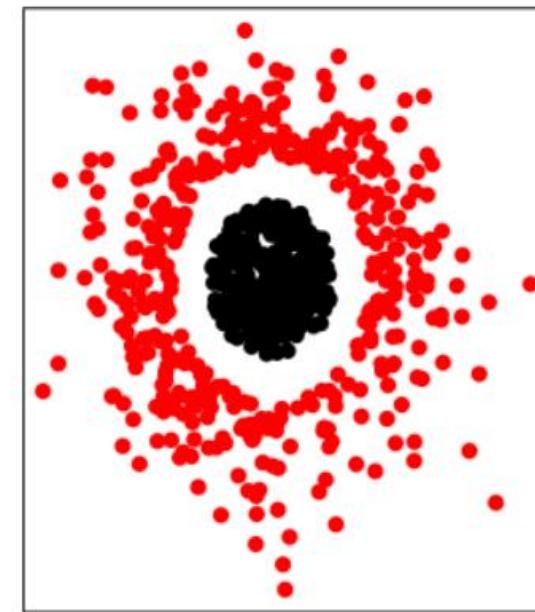


Imagen maximal margin hyperplane obtenida del libro ISLR

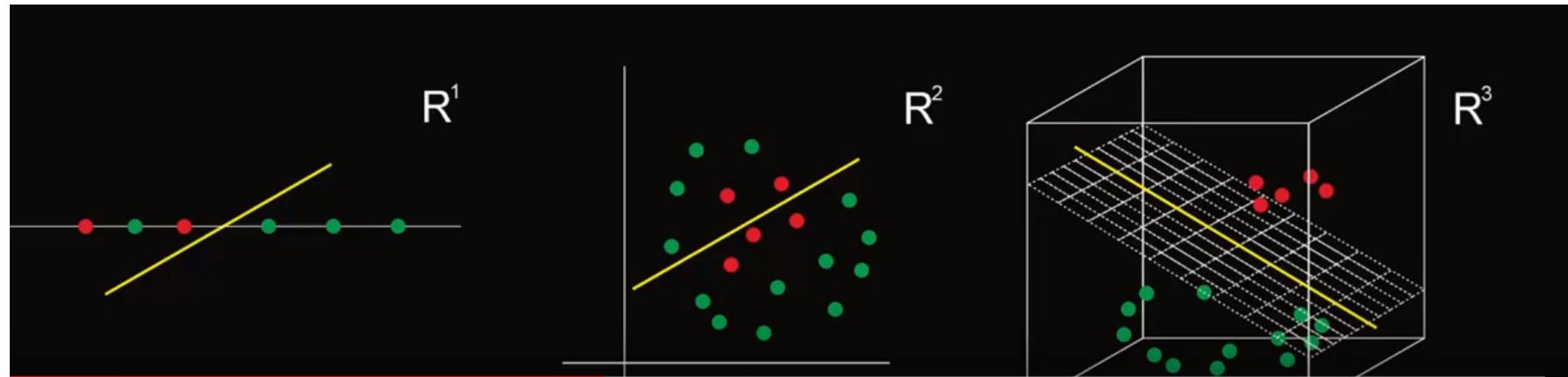
# Técnicas: 5. Máquinas de soporte vectorial (Support Vector Machines, SVMs)

- Pueden usar muchos tipos de **funciones del núcleo**.
- Las funciones del núcleo más comunes son **lineal** y la **función de base radial o RBF** (conjunto de núcleos locales en forma de Gauss para encontrar una separación no lineal de las clases).



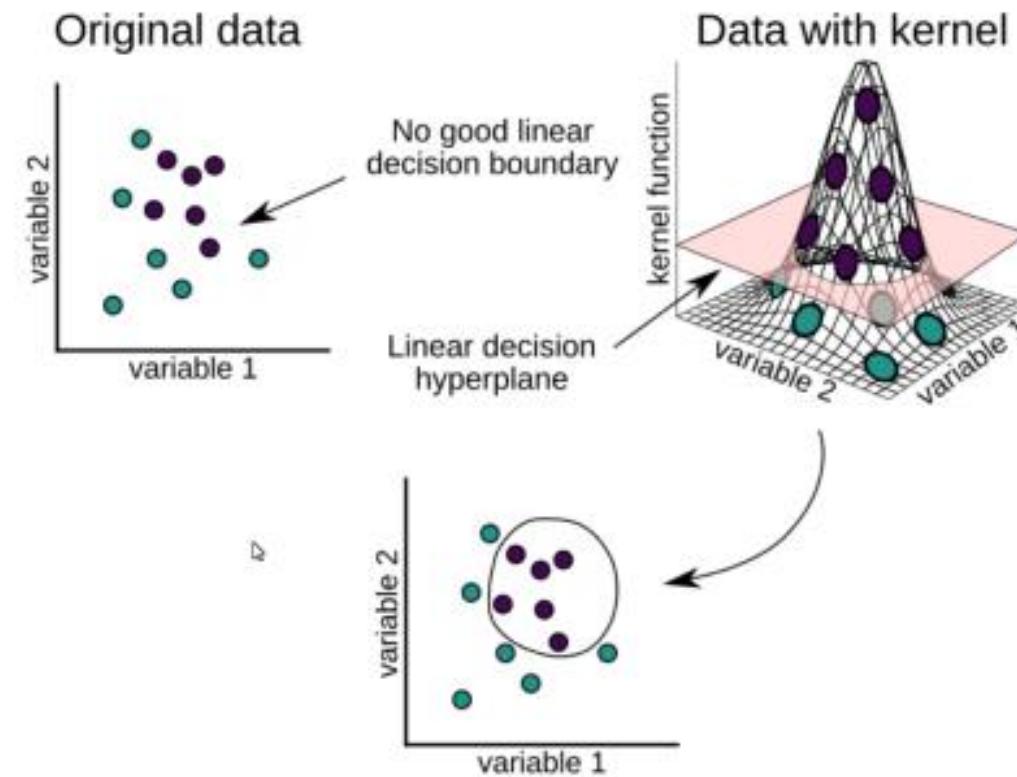
# Técnicas: 5. Máquinas de soporte vectorial (Support Vector Machines, SVMs)

- Encontrar el **vector de soporte** con la tasa de error más baja o la separación máxima

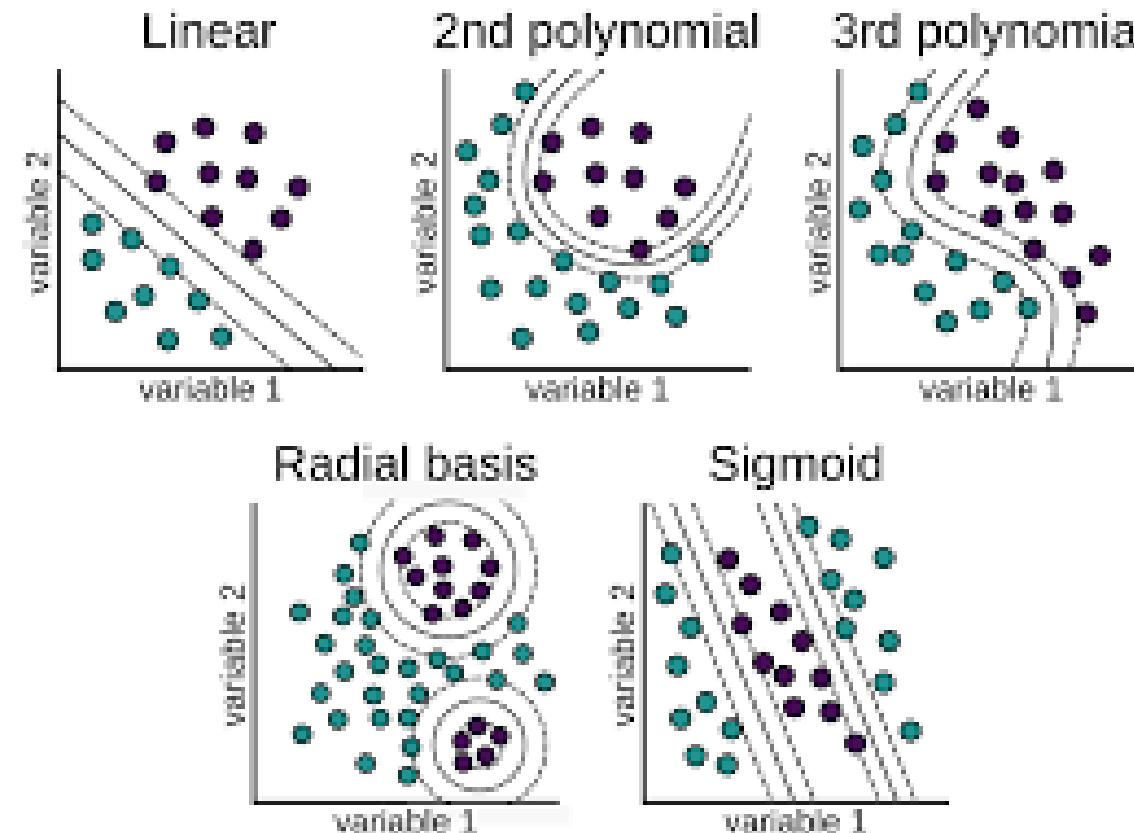


# Técnicas: 5. Máquinas de soporte vectorial (Support Vector Machines, SVMs)

➤ Ejemplo:

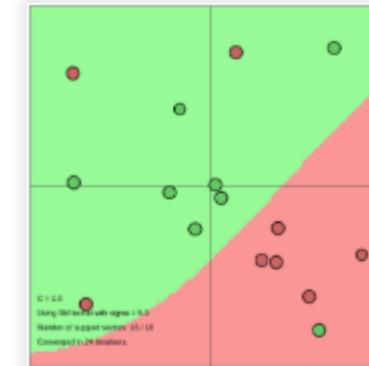
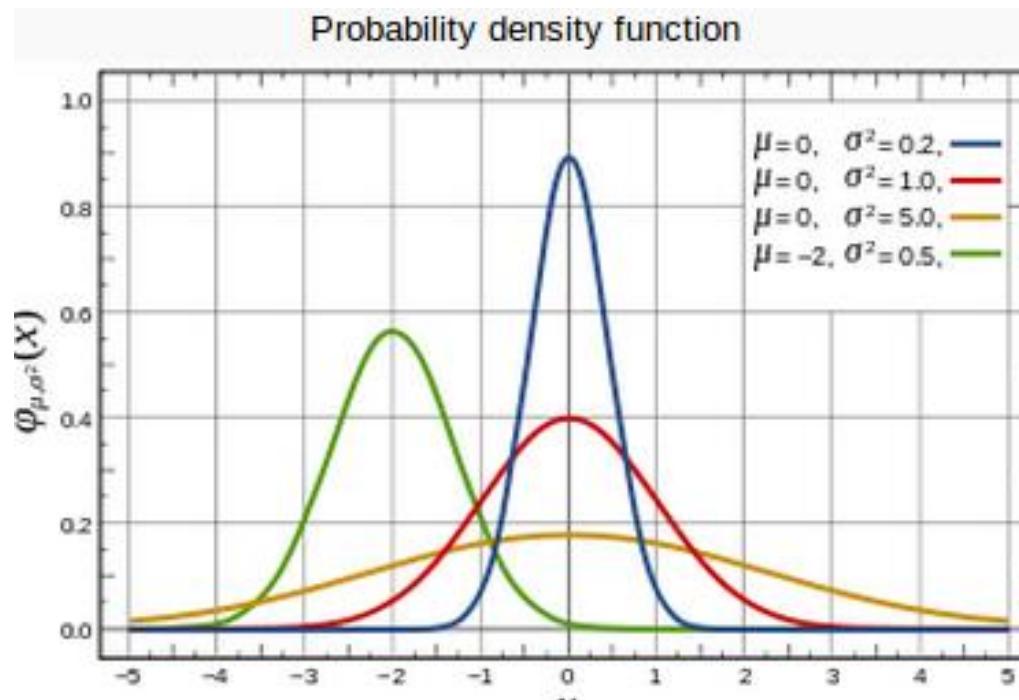


# Técnicas: 5. Máquinas de soporte vectorial (Support Vector Machines, SVMs)

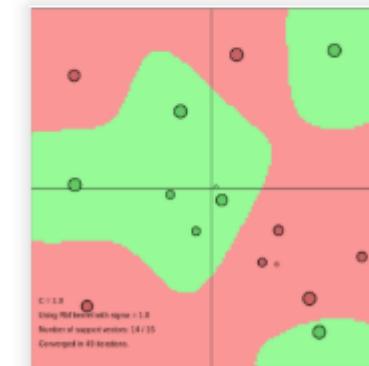


<https://www.youtube.com/watch?v=3lwicUTEgHs>

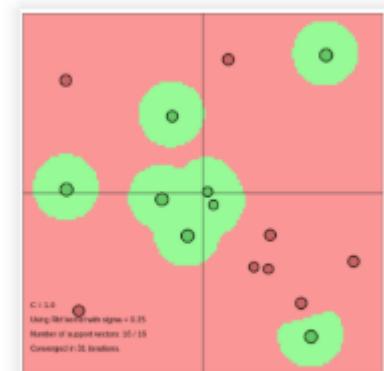
# Técnicas: 5. Máquinas de soporte vectorial (Support Vector Machines, SVMs)



Sigma = 5



Sigma = 1



Sigma = 0.25

# Técnicas: 5. Máquinas de soporte vectorial (Support Vector Machines, SVMs)

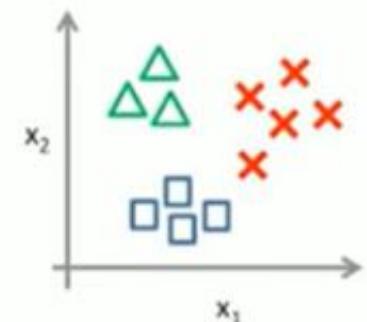
Main Dataset

Features			Classes
x1	x2	x3	G
x4	x5	x6	B
x7	x8	x9	R
x10	x11	x12	G
x13	x14	x15	B
x16	x17	x18	R

Class 1 :- Green   Class 2 :- Blue   Class 3 :- Red

## One vs. All (One-vs-Rest)

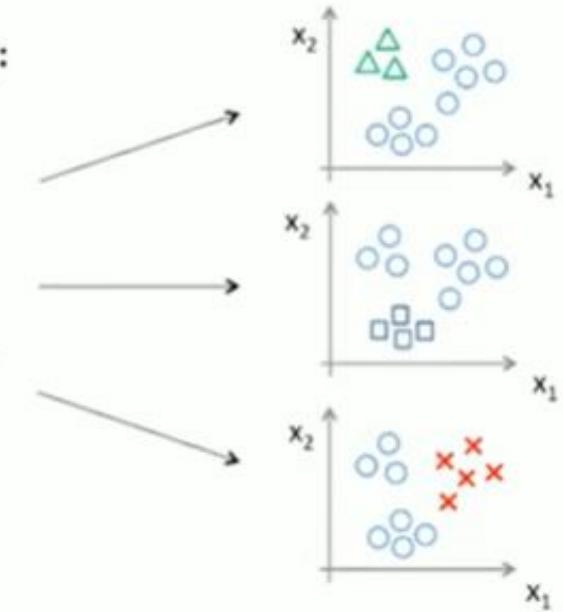
One-vs-all (one-vs-rest):



Class 1: Green

Class 2: Blue

Class 3: Red



# Técnicas: 5. Máquinas de soporte vectorial (Support Vector Machines, SVMs)

## One vs. All (One-vs-Rest)

Training Dataset 1  
Class :- Green

Features			Green
x1	x2	x3	+1
x4	x5	x6	-1
x7	x8	x9	-1
x10	x11	x12	+1
x13	x14	x15	-1
x16	x17	x18	-1

Training Dataset 2  
Class :- Blue

Features			Blue
x1	x2	x3	-1
x4	x5	x6	+1
x7	x8	x9	-1
x10	x11	x12	-1
x13	x14	x15	+1
x16	x17	x18	-1

Training Dataset 3  
Class :- Red

Features			Red
x1	x2	x3	-1
x4	x5	x6	-1
x7	x8	x9	+1
x10	x11	x12	-1
x13	x14	x15	-1
x16	x17	x18	+1

**Green** class classifier -> **Positive** with a probability score of **(0.9)**

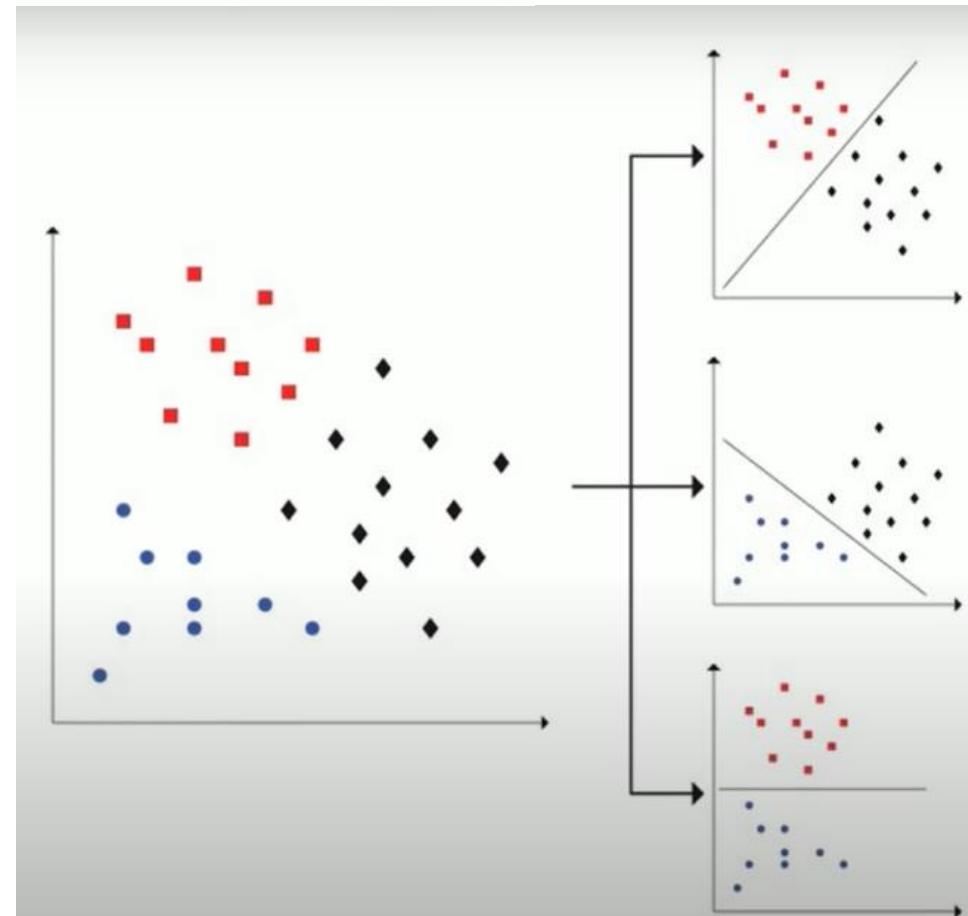
**Blue** class classifier -> **Positive** with a probability score of **(0.4)**

**Red** class classifier -> **Negative** with a probability score of **(0.5)**

Hence, based on the positive responses and decisive probability score, we can say that our test input belongs to the **Green** class.

# Técnicas: 5. Máquinas de soporte vectorial (Support Vector Machines, SVMs)

## One vs. One (OvO)



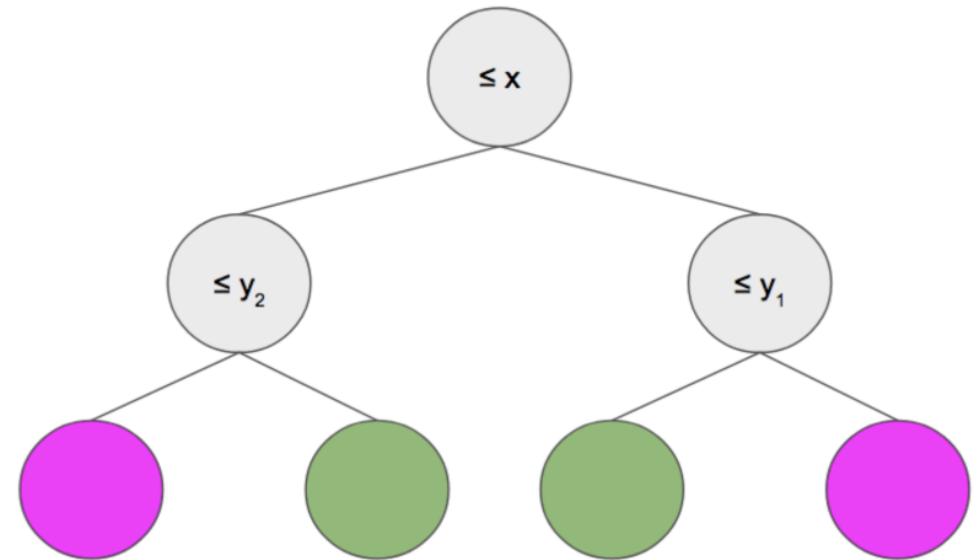
We divide this problem into  $N * (N-1)/2 = 3$  binary classifier problems:

- Classifier 1: Green vs. Blue
- Classifier 2: Green vs. Red
- Classifier 3: Blue vs. Red

When we input the test data to the classifier, then the model with the majority counts is concluded as a result.

# Técnicas: 6. Arboles de decisión

1. Se selecciona la característica que mejor logra dividir los ejemplos de acuerdo con el valor objetivo.
2. Cada prueba divide los datos de ejemplo en dos ramas.
3. Se repite el proceso de forma recurrente hasta cumplir los criterios de parada.

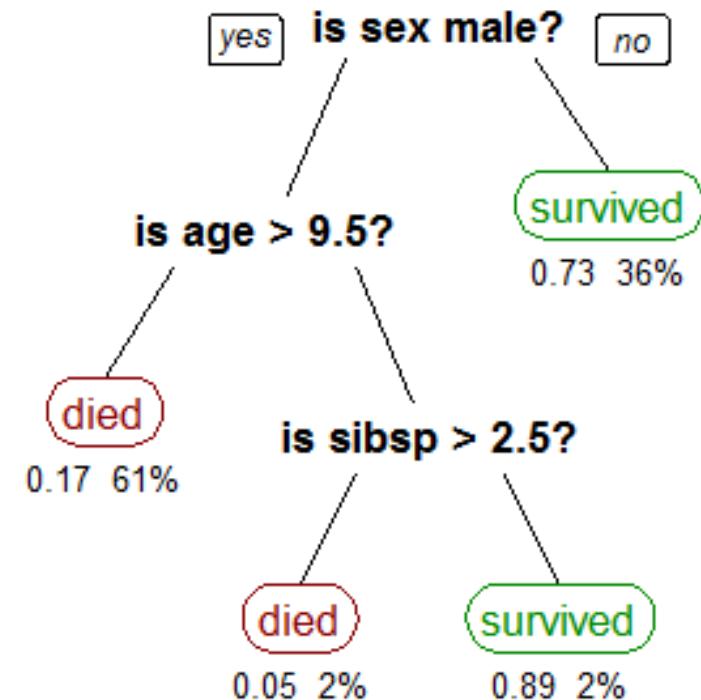


Un árbol de decisión puede emplear valores de categoría o numéricos para los nodos de decisión.

# Técnicas: 6. Arboles de decisión

## Criterios de parada

- Un grupo contiene la mayoría de las instancias que cuentan con el mismo valor objetivo.
- Ya han sido buscadas todas las características de manera exhaustiva.
- El árbol ha alcanzado una profundidad específica en términos de cantidad de ramas.



# Técnicas: 6. Arboles de decisión: Random Forest

## Métodos ensemble

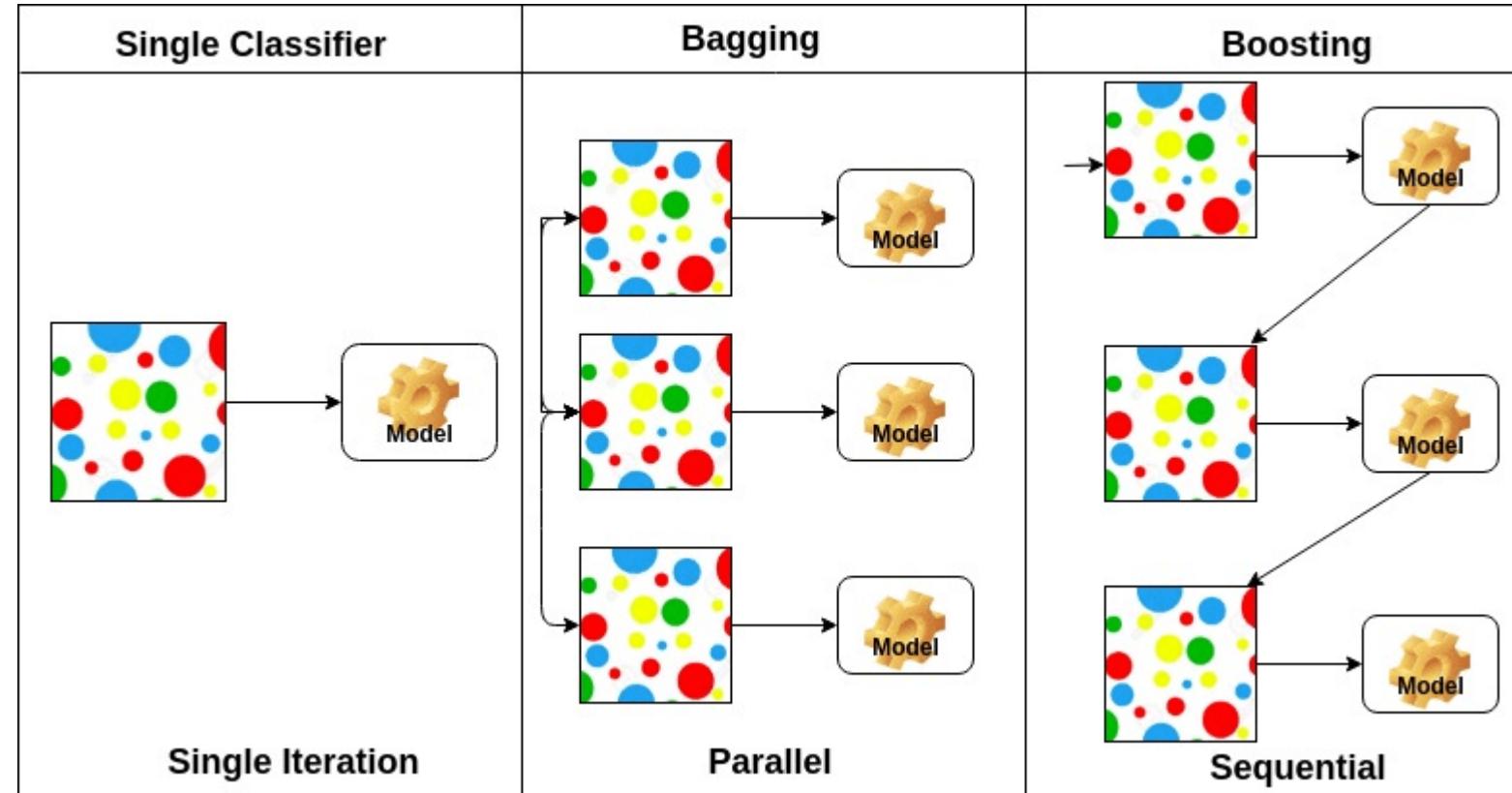
Combinan **múltiples modelos** en uno nuevo con el objetivo de lograr un equilibrio entre el Sesgo y varianza, **consiguiendo así mejores predicciones** que cualquiera de los modelos individuales originales.



1. Bagging

2. Boosting

# Técnicas: 6. Arboles de decisión: Random Forest



# Técnicas: 6. Arboles de decisión: Random Forest

## 1. Bagging

- Los modelos **Random Forest** están dentro de esta categoría.
- Se ajustan múltiples modelos, cada uno con un subconjunto distinto de los datos de entrenamiento.
- Para predecir, todos los modelos que forman el agregado participan aportando su predicción.
- Como valor final, se toma la media de todas las predicciones (variables continuas) o la clase más frecuente (variables categóricas).

# Técnicas: 6. Arboles de decisión: Random Forest

## 2. Boosting

- Los métodos de boosting más empleados son: **AdaBoost**, **Gradient Boosting** y **Stochastic Gradient Boosting**.
- Se ajustan secuencialmente múltiples modelos sencillos, llamados *weak learners*, de forma que cada modelo aprende de los errores del anterior.
- Como valor final, al igual que en *bagging*, se toma la media de todas las predicciones (variables continuas) o la clase más frecuente (variables cualitativas).

# Técnicas: 6. Arboles de decisión: Random Forest

## 1. Aplicar bootstrap

- Bootstrap es un método de remuestreo
- La idea es obtener muestras aleatorias con reemplazamiento (es posible incluir el mismo dato varias veces), siendo esas muestras de igual tamaño N que el conjunto de datos original.

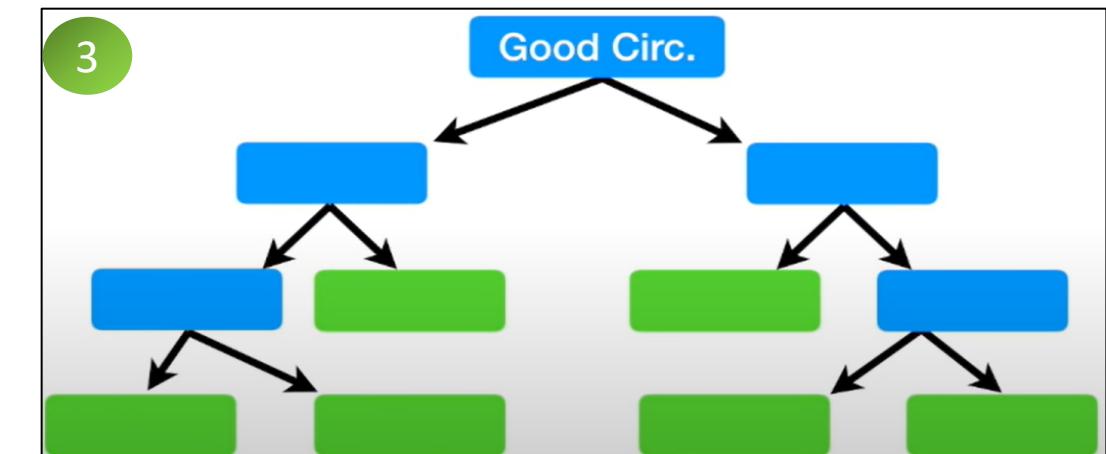
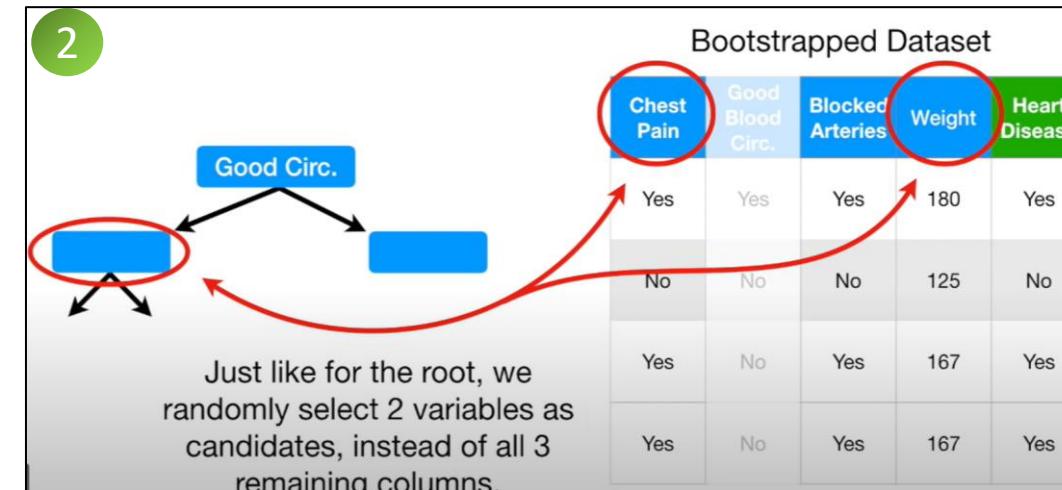
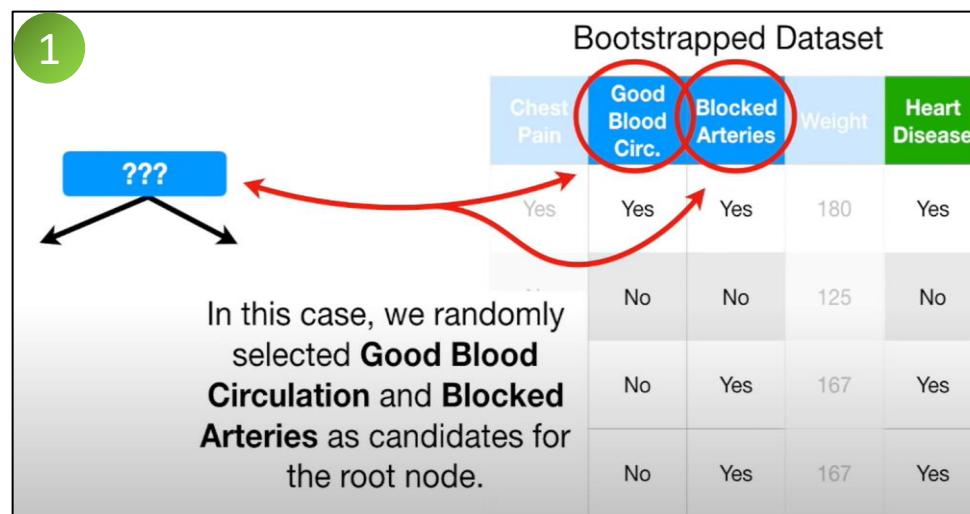
The diagram illustrates the bootstrap process. On the left, the "Original Dataset" is shown as a table with five columns: Chest Pain, Good Blood Circ., Blocked Arteries, Weight, and Heart Disease. It contains four rows of data. On the right, the "Bootstrapped Dataset" is shown as a similar table. Four arrows point from specific rows in the original dataset to specific rows in the bootstrapped dataset, demonstrating how the same data can be selected multiple times with replacement.

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease	
No	No	No	125	No	
Yes	Yes	Yes	180	Yes	
Yes	Yes	No	210	No	
Yes	No	Yes	167	Yes	

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

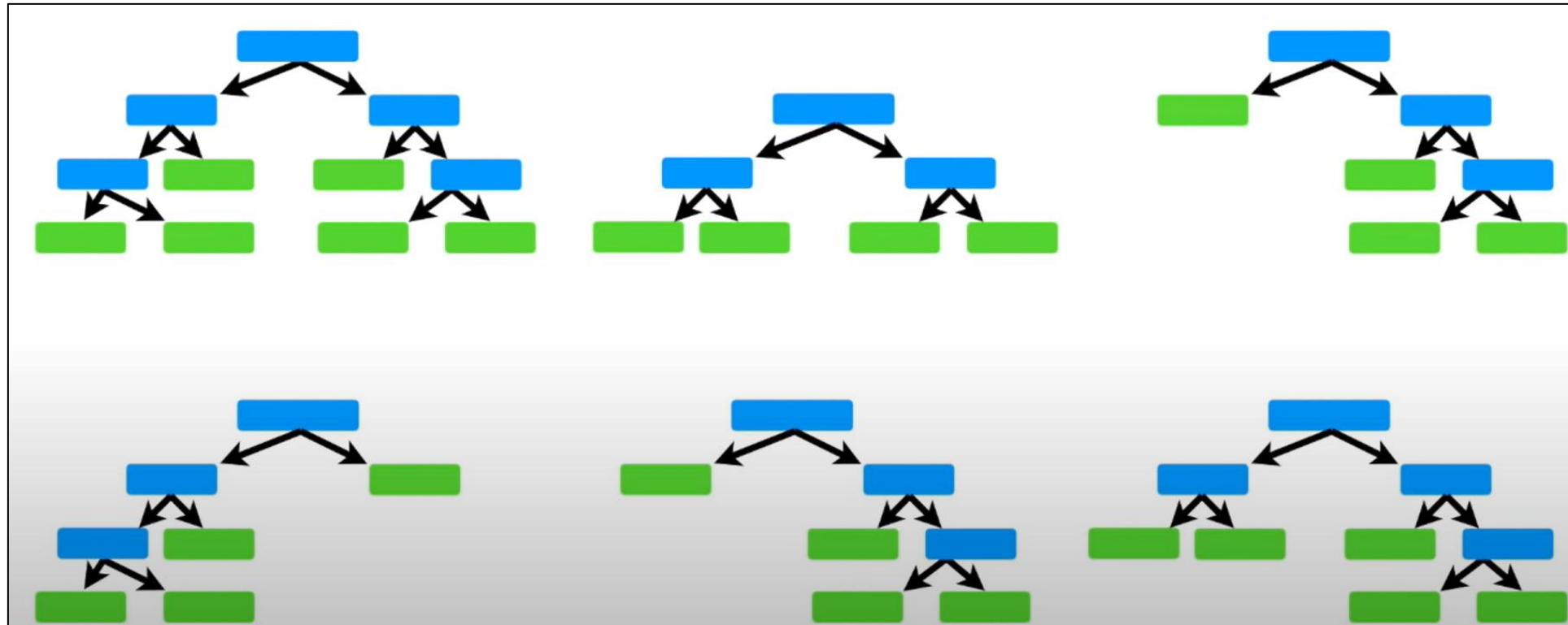
# Técnicas: 6. Arboles de decisión: Random Forest

## 2. Seleccionar aleatoriamente las variables en cada paso de construcción del árbol



# Técnicas: 6. Árboles de decisión: Random Forest

3. Se construyen tantos árboles como se consideren

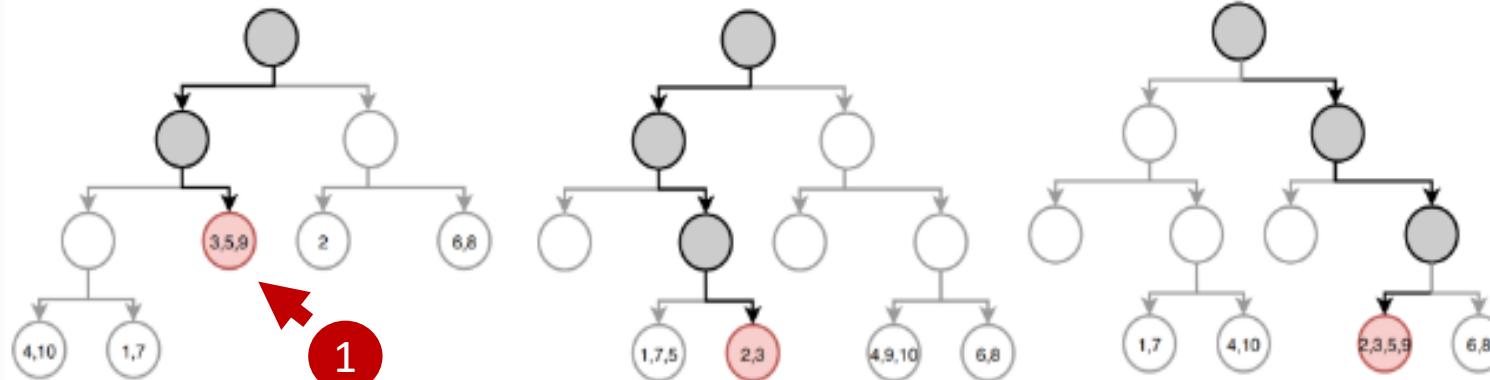


# Técnicas: 6. Arboles de decisión: Random Forest

## Ejemplo para cálculo del estimador:

Supóngase que se dispone de 10 observaciones, cada una con un valor de variable respuesta Y y unos predictores X.

id	1	2	3	4	5	6	7	8	9	10
Y	10	18	24	8	2	9	16	10	20	14
X	...	...	...	...	...	...	...	...	...	...



$$\hat{\mu} = \frac{15.33333 + 21 + 16}{3} = 17.4$$

$$\hat{y}_{arbol_1} = \frac{24 + 2 + 20}{3} = 15.33333$$

$$\hat{y}_{arbol_2} = \frac{18 + 24}{2} = 21$$

$$\hat{y}_{arbol_3} = \frac{18 + 24 + 2 + 20}{4} = 16$$

# Técnicas: 6. Arboles de decisión: Random Forest

**Out of bag:** Estos ejemplos que quedaron por fuera del dataset inicial. Pueden ser utilizados para evaluar el modelo

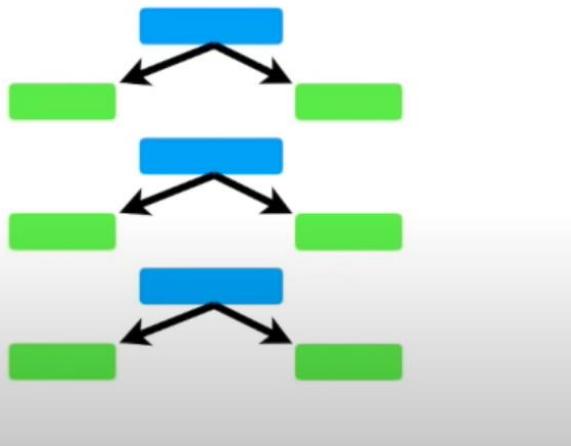
Original Dataset				
Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
No	No	No	125	No
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes

This is called the  
**“Out-Of-Bag Dataset”**

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	No	210	No

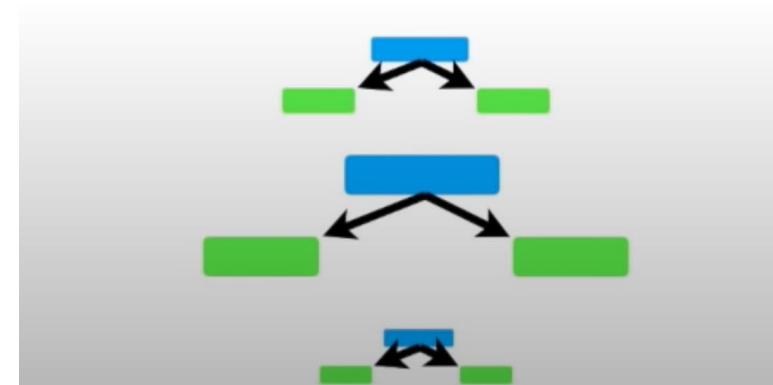
# Técnicas: 7. AdaBoost (Adaptive Boosting)

1) AdaBoost combines a lot of “weak learners” to make classifications. The weak learners are almost always stumps.



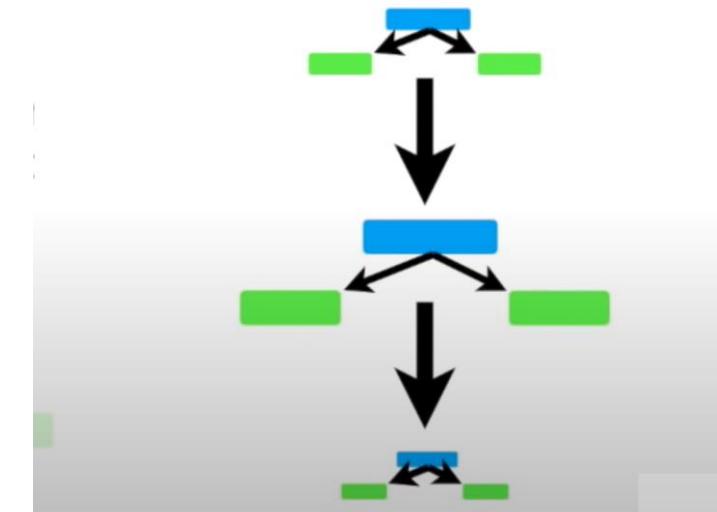
Clasificadores “débiles”

2) Some stumps get more say in the classification than others.



Ponderación

3) Each stump is made by taking the previous stump's mistakes into account.

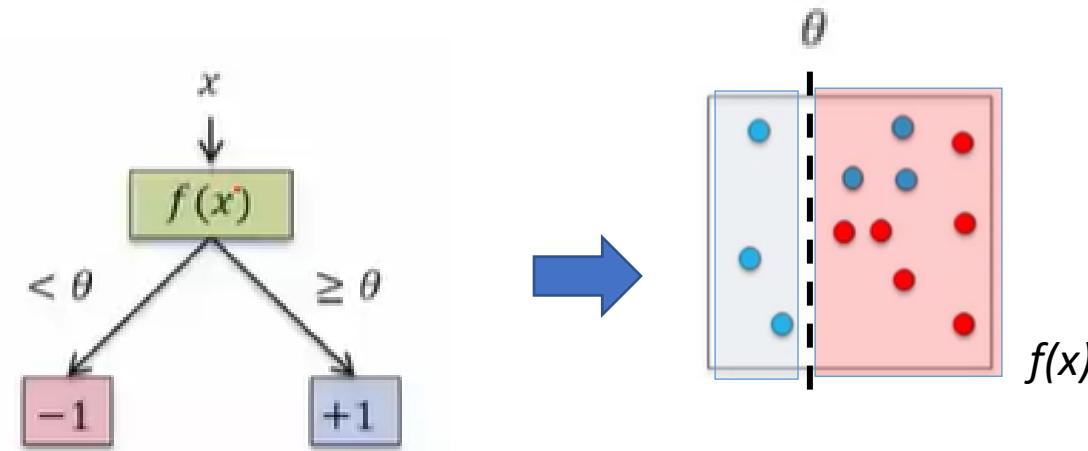


Aprende de los errores del anterior

# Técnicas: 7. AdaBoost (Adaptive Boosting)

## Clasificador débil (weak classifier)

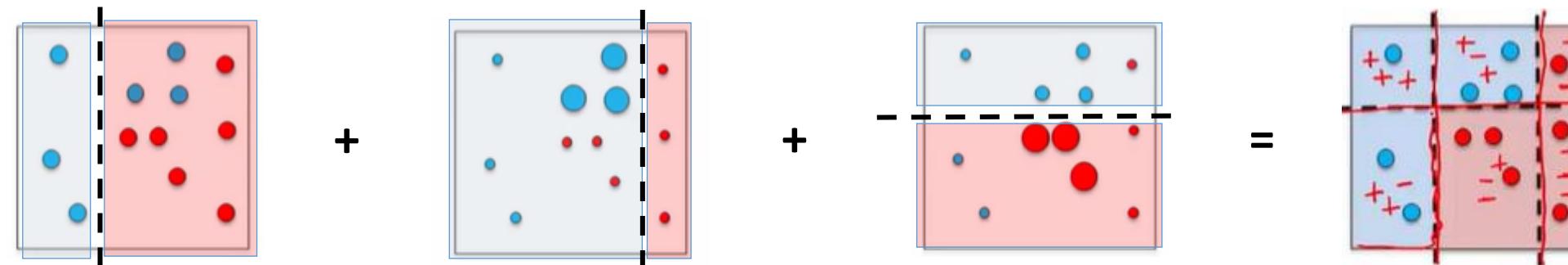
1. Decision Stump: árbol de decisión binario de profundidad 1



# Técnicas: 7. AdaBoost (Adaptive Boosting)

## Definición

1. Combinación de clasificadores “débiles” en un clasificador global “Fuerte”
2. Cada clasificador simple aprende dando un peso mayor a los mal clasificados previamente y un peso menor a los bien clasificados previamente

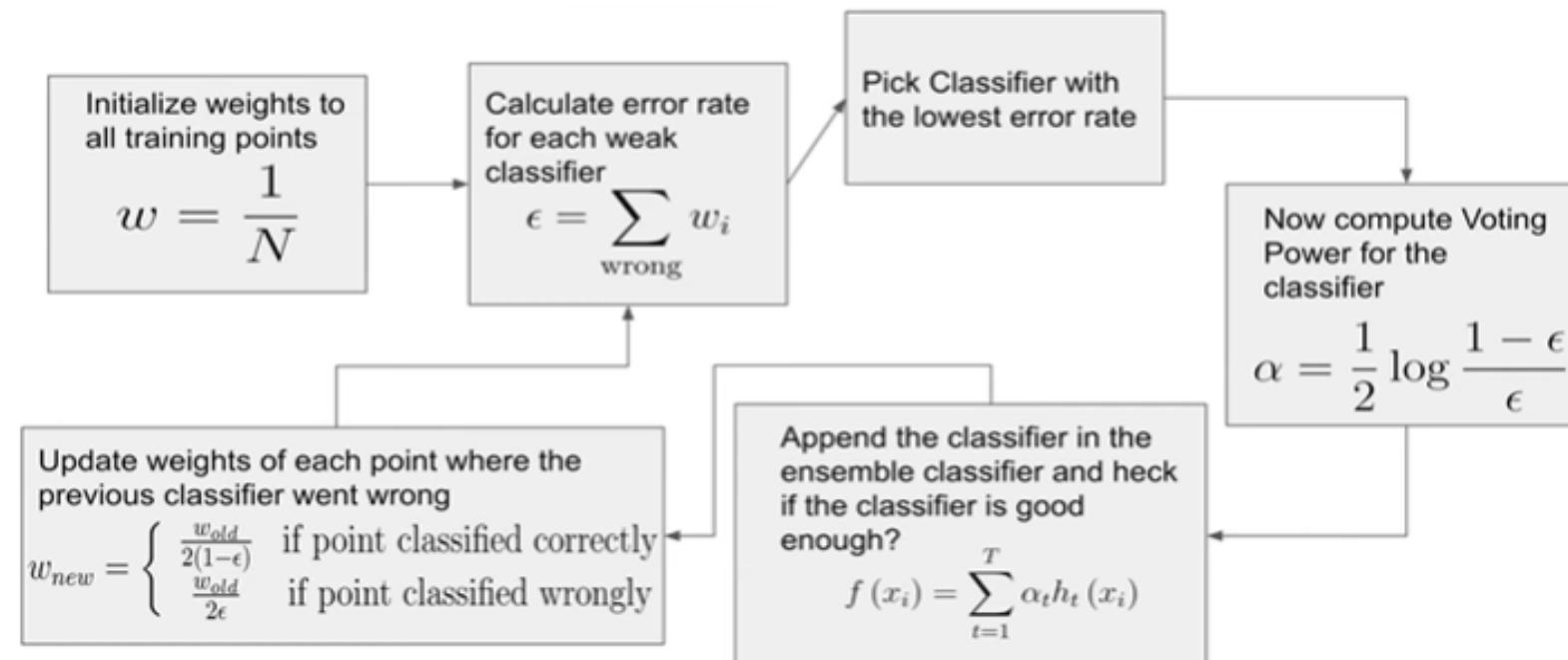


# Técnicas: 7. AdaBoost (Adaptive Boosting)

## Algoritmo

1. Entrenar un clasificador.
2. Use el clasificador.
3. Identifique los casos que fueron mal clasificados.
4. Construya un nuevo clasificador que clasifique mejor los casos mal clasificados del punto anterior.
5. Repita los pasos 2 a 4 varias veces.
6. Asígnele un peso a cada clasificador y júntelos para obtener un clasificador con mejor desempeño.

# Técnicas: 7. AdaBoost (Adaptive Boosting)



# Técnicas: 7. AdaBoost (Adaptive Boosting)

$$w = \frac{1}{N}$$

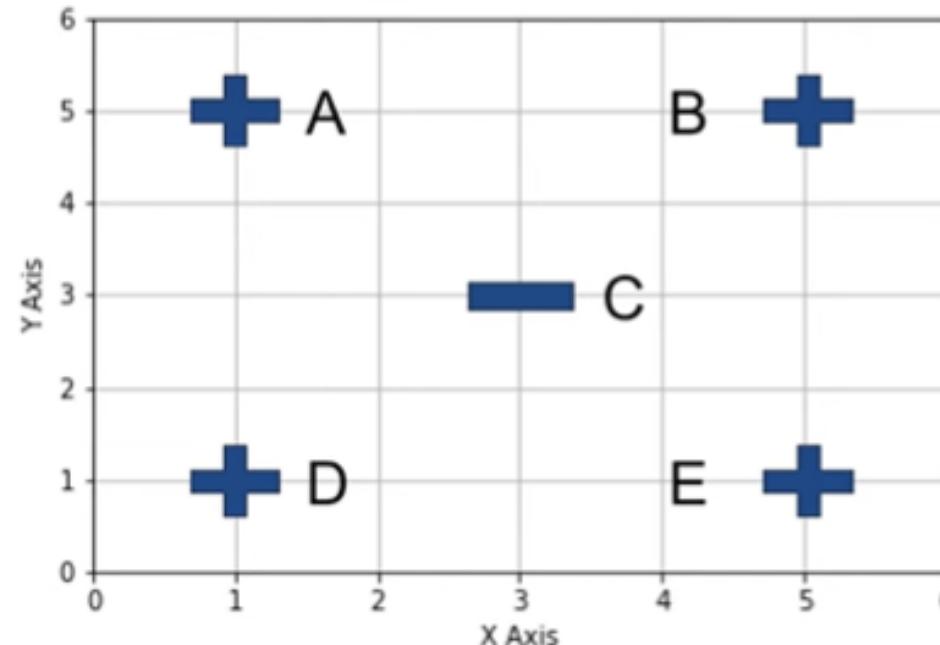
$$\epsilon = \sum_{\text{wrong}} w_i$$

Pick Lowest  
Error Rate  
Classifier

$$\alpha = \frac{1}{2} \log \frac{1 - \epsilon}{\epsilon}$$

$$f(x_i) = \sum_{t=1}^T \alpha_t h_t(x_i)$$

$$w_{\text{new}} = \begin{cases} \frac{w_{\text{old}}}{2(1-\epsilon)} \\ \frac{w_{\text{old}}}{2\epsilon} \end{cases}$$



- Let's start with 5 points.
- 4 points belong to class 1 & are denoted by plus sign.
- 1 point belongs to class 0 and is denoted by minus sign.

# Técnicas: 7. AdaBoost (Adaptive Boosting)

$$w = \frac{1}{N}$$

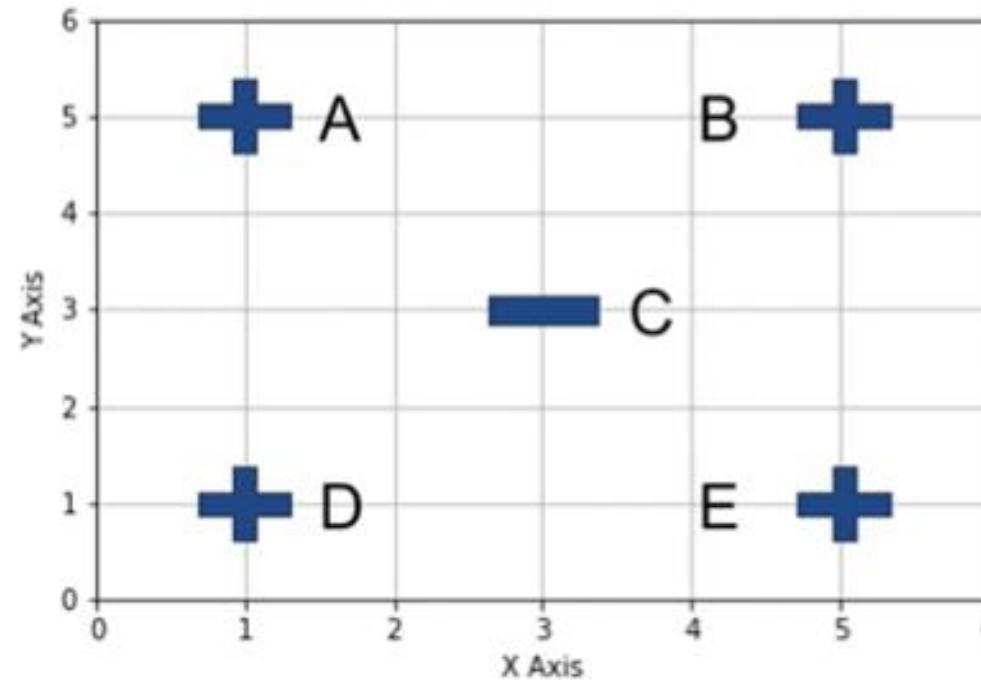
$$\epsilon = \sum_{\text{wrong}} w_i$$

Pick Lowest  
Error Rate  
Classifier

$$\alpha = \frac{1}{2} \log \frac{1 - \epsilon}{\epsilon}$$

$$f(x_i) = \sum_{t=1}^T \alpha_t h_t(x_i)$$

$$w_{\text{new}} = \begin{cases} \frac{w_{\text{old}}}{2(1-\epsilon)} \\ \frac{w_{\text{old}}}{2\epsilon} \end{cases}$$



Points	Weight
Wa	1/5
Wb	1/5
Wc	1/5
Wd	1/5
We	1/5

# Técnicas: 7. AdaBoost (Adaptive Boosting)

$$w = \frac{1}{N}$$

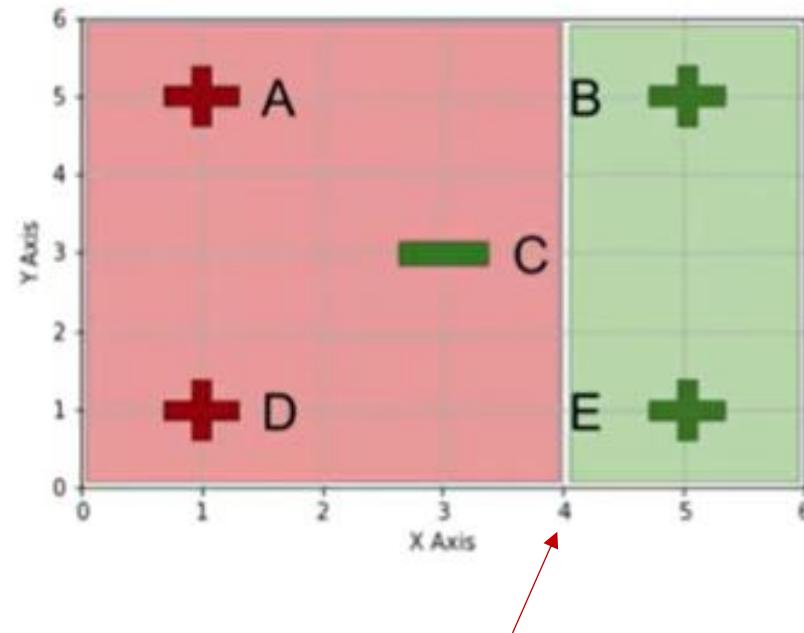
$$\epsilon = \sum_{\text{wrong}} w_i$$

Pick Lowest Error Rate Classifier

$$\alpha = \frac{1}{2} \log \frac{1 - \epsilon}{\epsilon}$$

$$f(x_i) = \sum_{t=1}^T \alpha_t h_t(x_i)$$

$$w_{\text{new}} = \begin{cases} \frac{w_{\text{old}}}{2(1-\epsilon)} \\ \frac{w_{\text{old}}}{2\epsilon} \end{cases}$$



C	Wrong	Error
X < 2	B & E	2/5
X < 4	B, C & E	3/5
X < 6	C	1/5
X > 2	A, D & C	3/5
X > 4	A, D	2/5
X > 6	A, B, D, E	4/5

Points	Weight
Wa	1/5
Wb	1/5
Wc	1/5
Wd	1/5
We	1/5

# Técnicas: 7. AdaBoost (Adaptive Boosting)

$$w = \frac{1}{N}$$

$$\epsilon = \sum_{\text{wrong}} w_i$$

Pick Lowest Error Rate Classifier

$$\alpha = \frac{1}{2} \log \frac{1-\epsilon}{\epsilon}$$

$$f(x_i) = \sum_{t=1}^T \alpha_t h_t(x_i)$$

$$w_{\text{new}} = \begin{cases} \frac{w_{\text{old}}}{2(1-\epsilon)} \\ \frac{w_{\text{old}}}{2\epsilon} \end{cases}$$

C	Wrong	Error
X < 2	B & E	2/5
X < 4	B, C & E	3/5
X < 6	C	1/5
X > 2	A, D & C	3/5
X > 4	A, D	2/5
X > 6	A, B, D, E	4/5

- Going through the error rates of each classifier, we find that X < 6 is the best performing classifier.
- Now, that we have decided on the classifier, let's now find the voting power of the classifier.

Points	Weight
Wa	1/5
Wb	1/5
Wc	1/5
Wd	1/5
We	1/5

# Técnicas: 7. AdaBoost (Adaptive Boosting)

$$w = \frac{1}{N}$$

$$\epsilon = \sum_{\text{wrong}} w_i$$

Pick Lowest  
Error Rate  
Classifier

$$\alpha = \frac{1}{2} \log \frac{1 - \epsilon}{\epsilon}$$

$$f(x_i) = \sum_{t=1}^T \alpha_t h_t(x_i)$$

$$w_{new} = \begin{cases} \frac{w_{old}}{2(1-\epsilon)} \\ \frac{w_{old}}{2\epsilon} \end{cases}$$

$$\epsilon = 1/5$$

$$\alpha = \frac{1}{2} \log \frac{1 - \frac{1}{5}}{\frac{1}{5}}$$

$$h(x) = \frac{1}{2} \log 4 * F(x < 6)$$

$$\alpha = \frac{1}{2} \log 4$$

# Técnicas: 7. AdaBoost (Adaptive Boosting)

$$w = \frac{1}{N}$$

$$\epsilon = \sum_{\text{wrong}} w_i$$

Pick Lowest  
Error Rate  
Classifier

$$\alpha = \frac{1}{2} \log \frac{1-\epsilon}{\epsilon}$$

$$f(x_i) = \sum_{t=1}^T \alpha_t h_t(x_i)$$

$$w_{new} = \begin{cases} \frac{w_{old}}{2(1-\epsilon)} & \text{if point classified correctly} \\ \frac{w_{old}}{2\epsilon} & \text{if point classified wrongly} \end{cases}$$

$$w_{new} = \begin{cases} \frac{w_{old}}{2(1-\epsilon)} & \text{if point classified correctly} \\ \frac{w_{old}}{2\epsilon} & \text{if point classified wrongly} \end{cases}$$

- Points C is incorrectly classifier.
- Plugging in the value of error =  $\frac{1}{5}$  and initial weight of  $\frac{1}{5}$  in the 1st equation.
- We get new weight =  $\frac{1}{2}$

Points	Weight
Wa	1/8
Wb	1/8
Wc	1/2
Wd	1/8
We	1/8

# Técnicas: 7. AdaBoost (Adaptive Boosting)

$$w = \frac{1}{N}$$

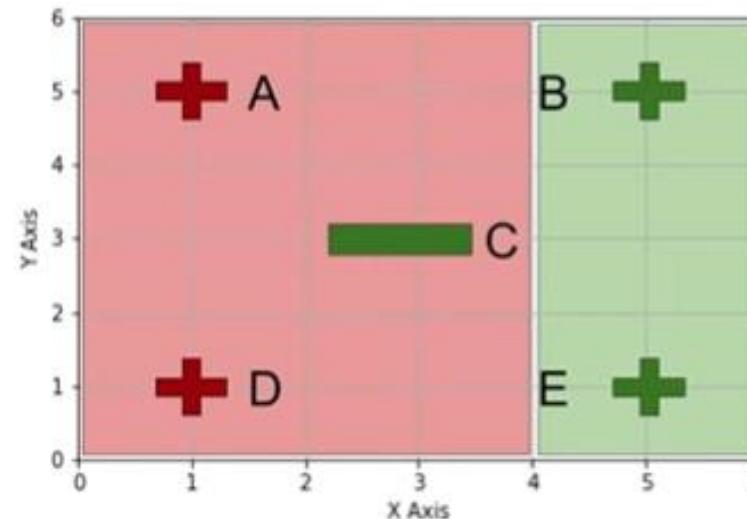
$$\epsilon = \sum_{\text{wrong}} w_i$$

Pick Lowest Error Rate Classifier

$$\alpha = \frac{1}{2} \log \frac{1 - \epsilon}{\epsilon}$$

$$f(x_i) = \sum_{t=1}^T \alpha_t h_t(x_i)$$

$$w_{\text{new}} = \begin{cases} \frac{w_{\text{old}}}{2(1-\epsilon)} & \text{if } \epsilon < \frac{1}{2} \\ \frac{w_{\text{old}}}{2\epsilon} & \text{otherwise} \end{cases}$$



C	Wrong	Error
X < 2	B & E	2/8
X < 4	B, C & E	6/8
X < 6	C	4/8
X > 2	A, C & D	6/8
X > 4	A, D	2/8
X > 6	A, B, D, E	4/8

Points	Weight
Wa	1/8
Wb	1/8
Wc	1/2
Wd	1/8
We	1/8

# Técnicas: 7. AdaBoost (Adaptive Boosting)

$$w = \frac{1}{N}$$

$$\epsilon = \sum_{\text{wrong}} w_i$$

Pick Lowest  
Error Rate  
Classifier

$$\alpha = \frac{1}{2} \log \frac{1-\epsilon}{\epsilon}$$

$$f(x_i) = \sum_{t=1}^T \alpha_t h_t(x_i)$$

$$w_{\text{new}} = \begin{cases} \frac{w_{\text{old}}}{2(1-\epsilon)} \\ \frac{w_{\text{old}}}{2\epsilon} \end{cases}$$

C	Wrong	Error
X < 2	B & E	2/8
X < 4	B, C & E	6/8
X < 6	C	4/8
X > 2	A, D & C	6/8
X > 4	A, D	2/8
X > 6	A, B, D, E	4/8

- Going through the error rates of each classifier, we find that there are 2 candidates X < 2 and X > 4 with minimum error rate.
- In this round, we will choose the 1st one i.e. X < 2 classifier.

# Técnicas: 7. AdaBoost (Adaptive Boosting)

$$w = \frac{1}{N}$$

$$\epsilon = \sum_{\text{wrong}} w_i$$

Pick Lowest  
Error Rate  
Classifier

$$\alpha = \frac{1}{2} \log \frac{1-\epsilon}{\epsilon}$$

$$f(x_i) = \sum_{t=1}^T \alpha_t h_t(x_i)$$

$$w_{new} = \begin{cases} \frac{w_{old}}{2(1-\epsilon)} \\ \frac{w_{old}}{2\epsilon} \end{cases}$$

$$\epsilon = 2/8$$

$$\alpha = \frac{1}{2} \log \frac{1-\frac{1}{4}}{\frac{1}{4}}$$

$$\alpha = \frac{1}{2} \log 3$$

$$h(x) = \frac{1}{2} \log 4 * F(x < 6) + \frac{1}{2} \log 3 * F(x < 2)$$

Learning rate

# Técnicas: 7. AdaBoost (Adaptive Boosting)

$$w = \frac{1}{N}$$

$$\epsilon = \sum_{\text{wrong}} w_i$$

Pick Lowest  
Error Rate  
Classifier

$$\alpha = \frac{1}{2} \log \frac{1 - \epsilon}{\epsilon}$$

$$f(x_i) = \sum_{t=1}^T \alpha_t h_t(x_i)$$

$$w_{\text{new}} = \begin{cases} \frac{w_{\text{old}}}{2(1-\epsilon)} & \text{if point classified correctly} \\ \frac{w_{\text{old}}}{2\epsilon} & \text{if point classified wrongly} \end{cases}$$

$$w_{\text{new}} = \begin{cases} \frac{w_{\text{old}}}{2(1-\epsilon)} & \text{if point classified correctly} \\ \frac{w_{\text{old}}}{2\epsilon} & \text{if point classified wrongly} \end{cases}$$

- Points B and E are incorrectly classified.
- Plugging in the value of error = 2/8 and initial weights from previous round we get the following points weight table

Points	Weight
Wa	1/12
Wb	3/12
Wc	4/12
Wd	1/12
We	3/12

# Técnicas: 7. AdaBoost (Adaptive Boosting)

$$w = \frac{1}{N}$$

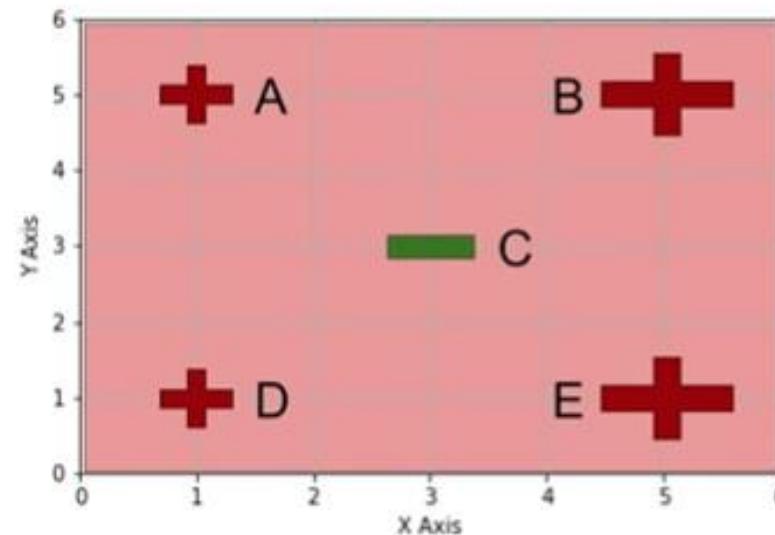
$$\epsilon = \sum_{\text{wrong}} w_i$$

Pick Lowest  
Error Rate  
Classifier

$$\alpha = \frac{1}{2} \log \frac{1 - \epsilon}{\epsilon}$$

$$f(x_i) = \sum_{t=1}^T \alpha_t h_t(x_i)$$

$$w_{\text{new}} = \begin{cases} \frac{w_{\text{old}}}{2(1-\epsilon)} & \text{if } \epsilon < \frac{1}{2} \\ \frac{w_{\text{old}}}{2\epsilon} & \text{otherwise} \end{cases}$$



C	Wrong	Error
$X < 2$	B & E	1/2
$X < 4$	B, C & E	10/12
$X < 6$	C	4/12
$X > 2$	A, C & D	1/2
$X > 4$	A, D	2/12
$X > 6$	A, B, D, E	8/12

Points	Weight
Wa	1/12
Wb	3/12
Wc	4/12
Wd	1/12
We	3/12

# Técnicas: 7. AdaBoost (Adaptive Boosting)

$$w = \frac{1}{N}$$

$$\epsilon = \sum_{\text{wrong}} w_i$$

Pick Lowest  
Error Rate  
Classifier

$$\alpha = \frac{1}{2} \log \frac{1-\epsilon}{\epsilon}$$

$$f(x_i) = \sum_{t=1}^T \alpha_t h_t(x_i)$$

$$w_{new} = \begin{cases} \frac{w_{old}}{2(1-\epsilon)} \\ \frac{w_{old}}{2\epsilon} \end{cases}$$

C	Wrong	Error
X < 2	B & E	1/2
X < 4	B, C & E	10/12
X < 6	C	4/12
X > 2	A, D & C	1/2
X > 4	A, D	2/12
X > 6	A, B, D, E	8/12

- Going through the error rates of each classifier, we find that X> 4 is the classifier with minimum error rate.
- In this round, we will choose X> 4 classifier.

# Técnicas: 7. AdaBoost (Adaptive Boosting)

$$w = \frac{1}{N}$$

$$\epsilon = \sum_{\text{wrong}} w_i$$

Pick Lowest  
Error Rate  
Classifier

$$\alpha = \frac{1}{2} \log \frac{1-\epsilon}{\epsilon}$$

$$f(x_i) = \sum_{t=1}^T \alpha_t h_t(x_i)$$

$$w_{new} = \begin{cases} \frac{w_{old}}{2(1-\epsilon)} \\ \frac{w_{old}}{2\epsilon} \end{cases}$$

$$\epsilon = 1/6$$

$$\alpha = \frac{1}{2} \log \frac{1-\frac{1}{6}}{\frac{1}{6}}$$

$$\alpha = \frac{1}{2} \log 5$$

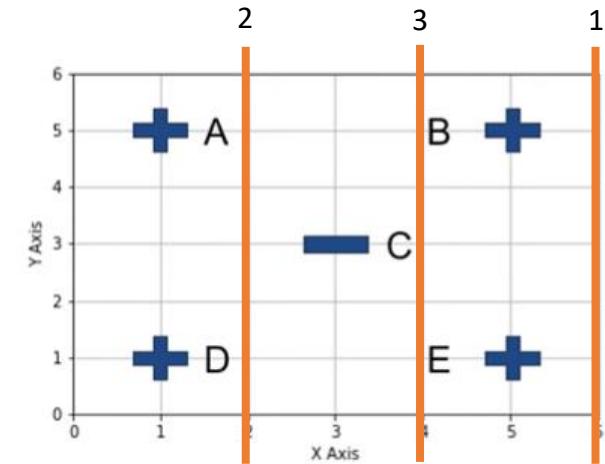
$$h(x) = \frac{1}{2} \log 4 * F(x < 6) + \frac{1}{2} \log 3 * F(x < 2) + \frac{1}{2} \log 5 * F(x > 4)$$

# Técnicas: 7. AdaBoost (Adaptive Boosting)

## Como resultado tenemos

Combinación de 3 clasificadores “débiles” en un clasificador global “Fuerte”

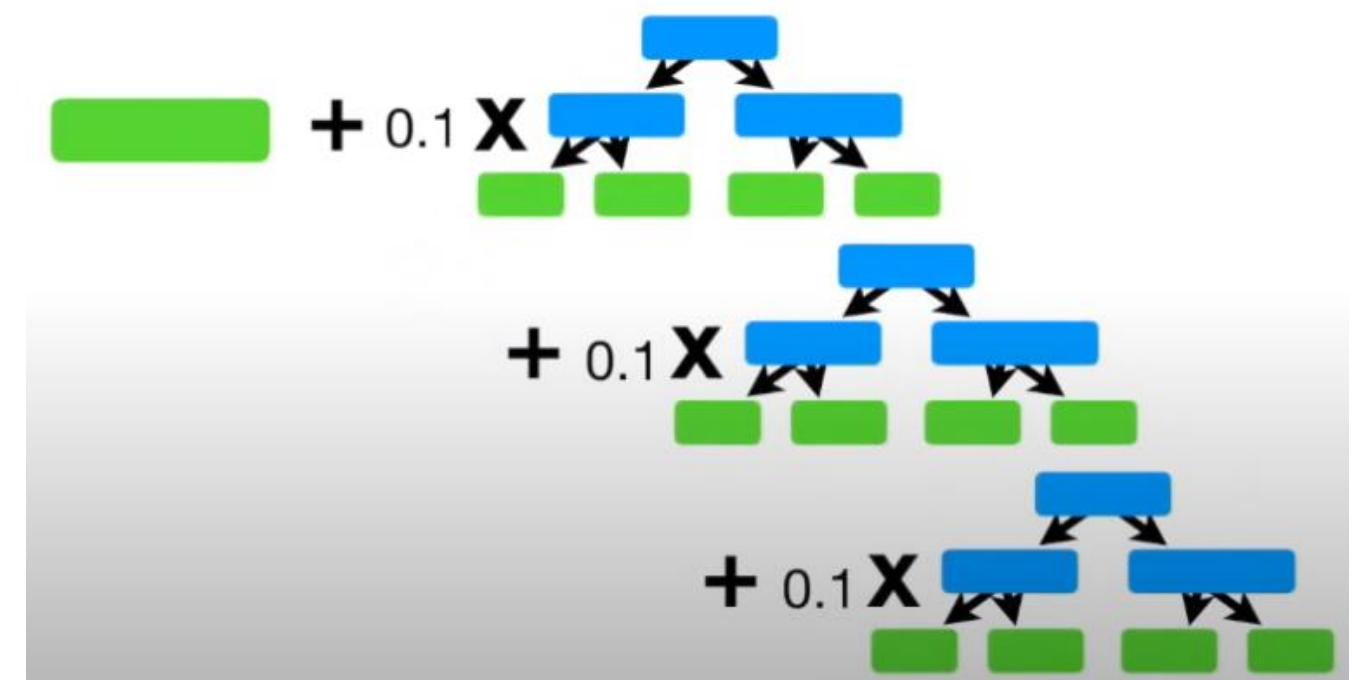
1. El primer clasificador: clasifica mal el punto C
2. El segundo clasificador: clasifica mal B y E
3. El tercer clasificador: clasifica mal A y B



$$h(x) = \frac{1}{2} \log 4 * F(x < 6) + \frac{1}{2} \log 3 * F(x < 2) + \frac{1}{2} \log 5 * F(x > 4)$$

# Técnicas: 8. Gradient Boosting

- *Gradient Boosting* es una generalización del algoritmo *AdaBoost*
- Su objetivo es crear modelos de forma secuencial, donde cada modelo ajusta los residuos (errores) de los modelos anteriores



# Técnicas: 8. Gradient Boosting



<https://www.youtube.com/watch?v=3CC4N4z3GJc>

<https://www.youtube.com/watch?v=jxuNLH5dXCs>

# Técnicas: 8. Gradient Boosting

Height (m)	Favorite Color	Gender	Weight (kg)	Residual
1.6	Blue	Male	88	16.8
1.6	Green	Female	76	4.8
1.5	Blue	Female	56	-15.2
1.8	Red	Male	73	1.8
1.5	Green	Male	77	5.8
1.4	Blue	Female	57	-14.2

Average Weight

71.2

$$(88 - 71.2) = 16.8$$

(Observed Weight - Predicted Weight)

# Técnicas: 8. Gradient Boosting

Height (m)	Favorite Color	Gender	Weight (kg)	Residual
1.6	Blue	Male	88	16.8
1.6	Green	Female	76	4.8
1.5	Blue	Female	56	-15.2
1.8	Red	Male	73	1.8
1.5	Green	Male	77	5.8
1.4	Blue	Female	57	-14.2

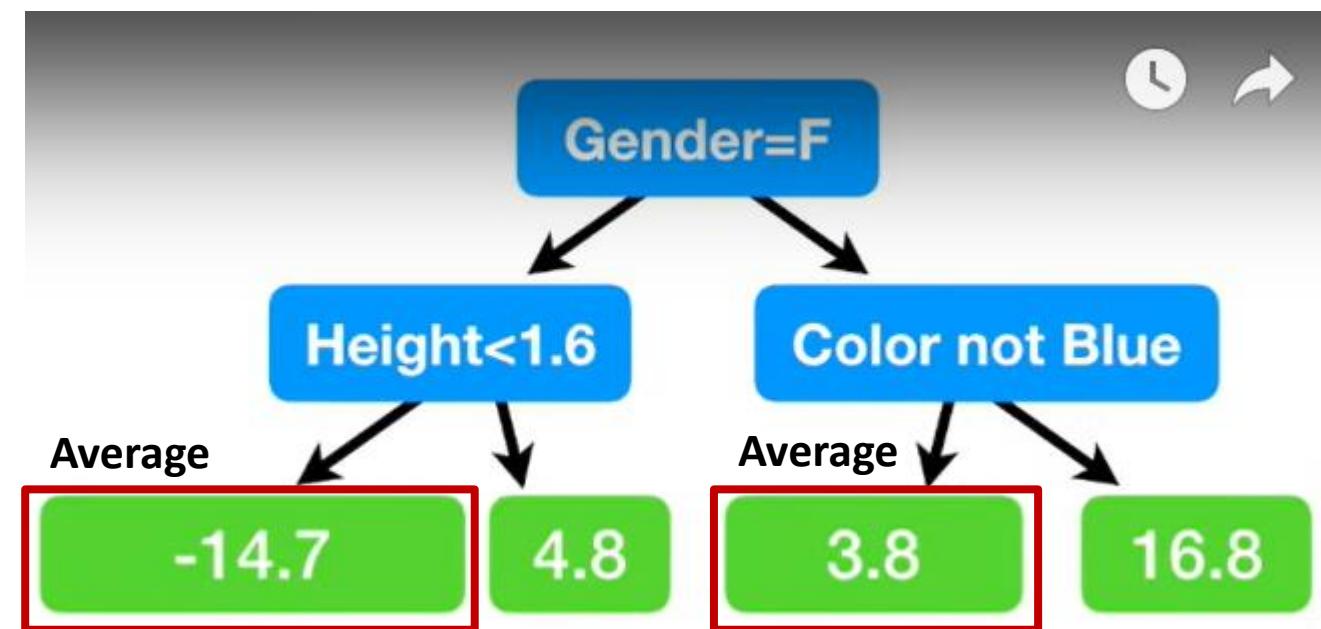
Now we will build a **Tree**, using **Height, Favorite Color and Gender...**



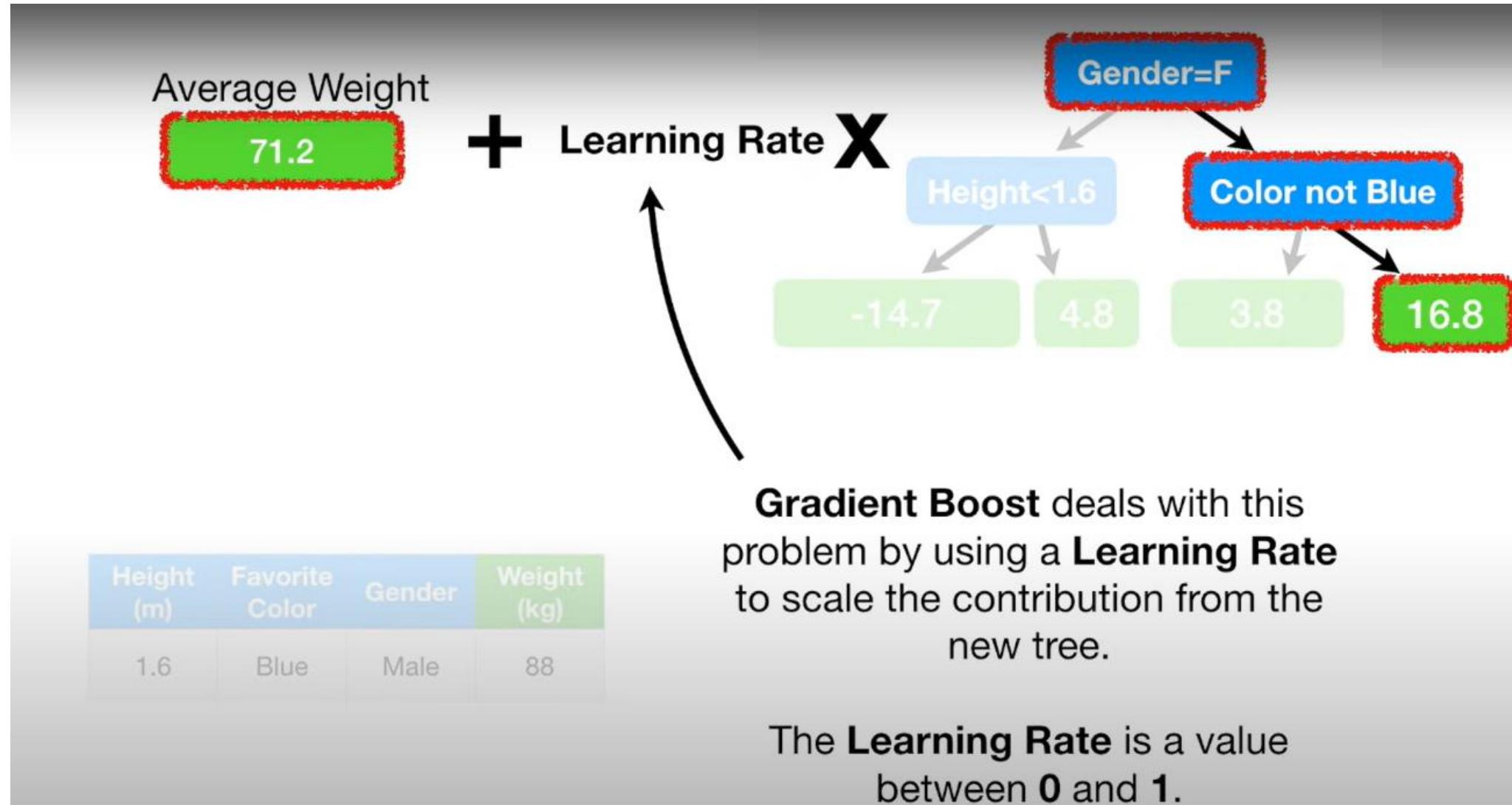
to Predict the **Residuals**.

# Técnicas: 8. Gradient Boosting

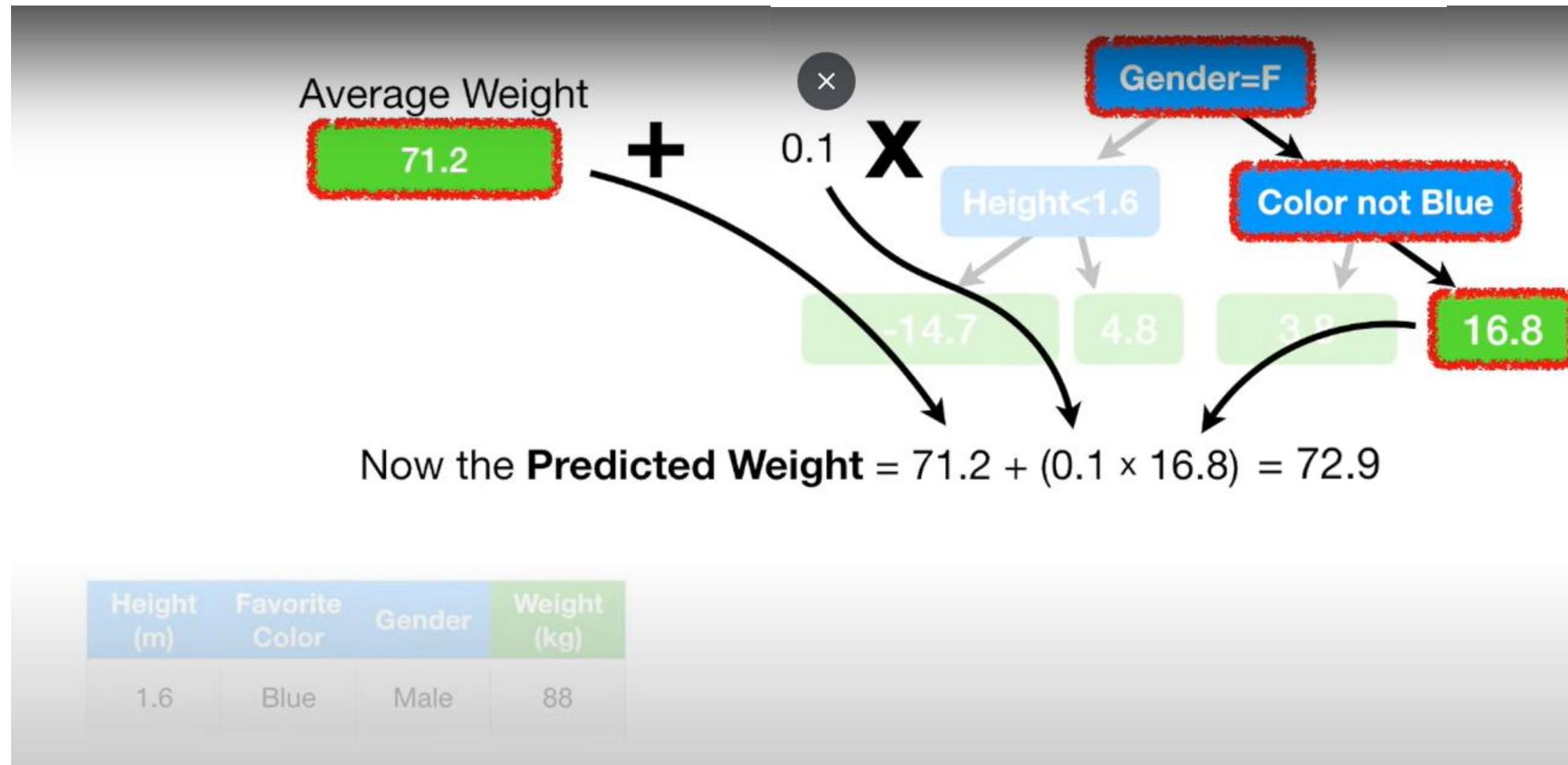
Height (m)	Favorite Color	Gender	Weight (kg)	Residual
1.6	Blue	Male	88	16.8
1.6	Green	Female	76	4.8
1.5	Blue	Female	56	-15.2
1.8	Red	Male	73	1.8
1.5	Green	Male	77	5.8
1.4	Blue	Female	57	-14.2



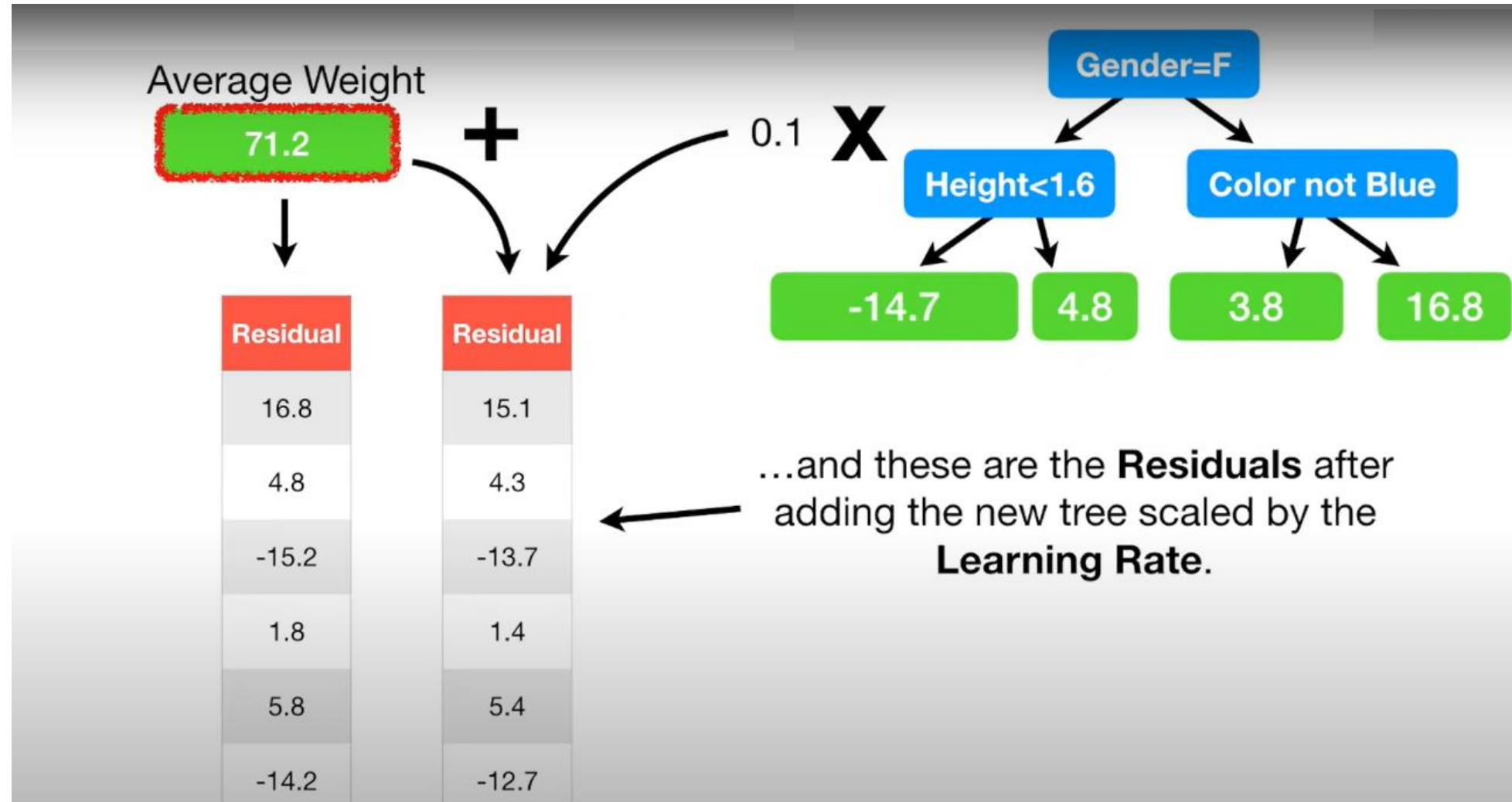
# Técnicas: 8. Gradient Boosting



# Técnicas: 8. Gradient Boosting



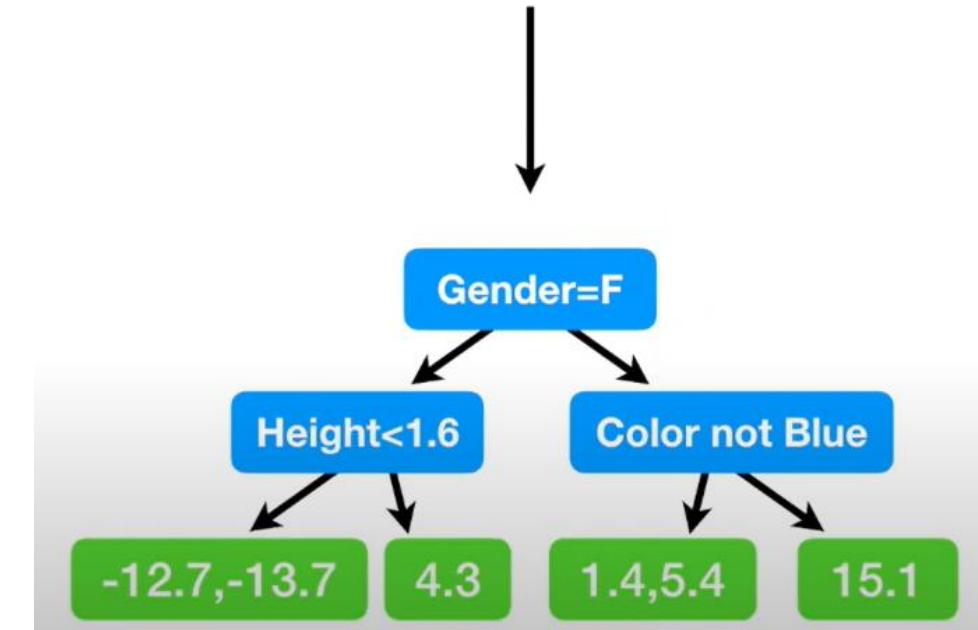
# Técnicas: 8. Gradient Boosting



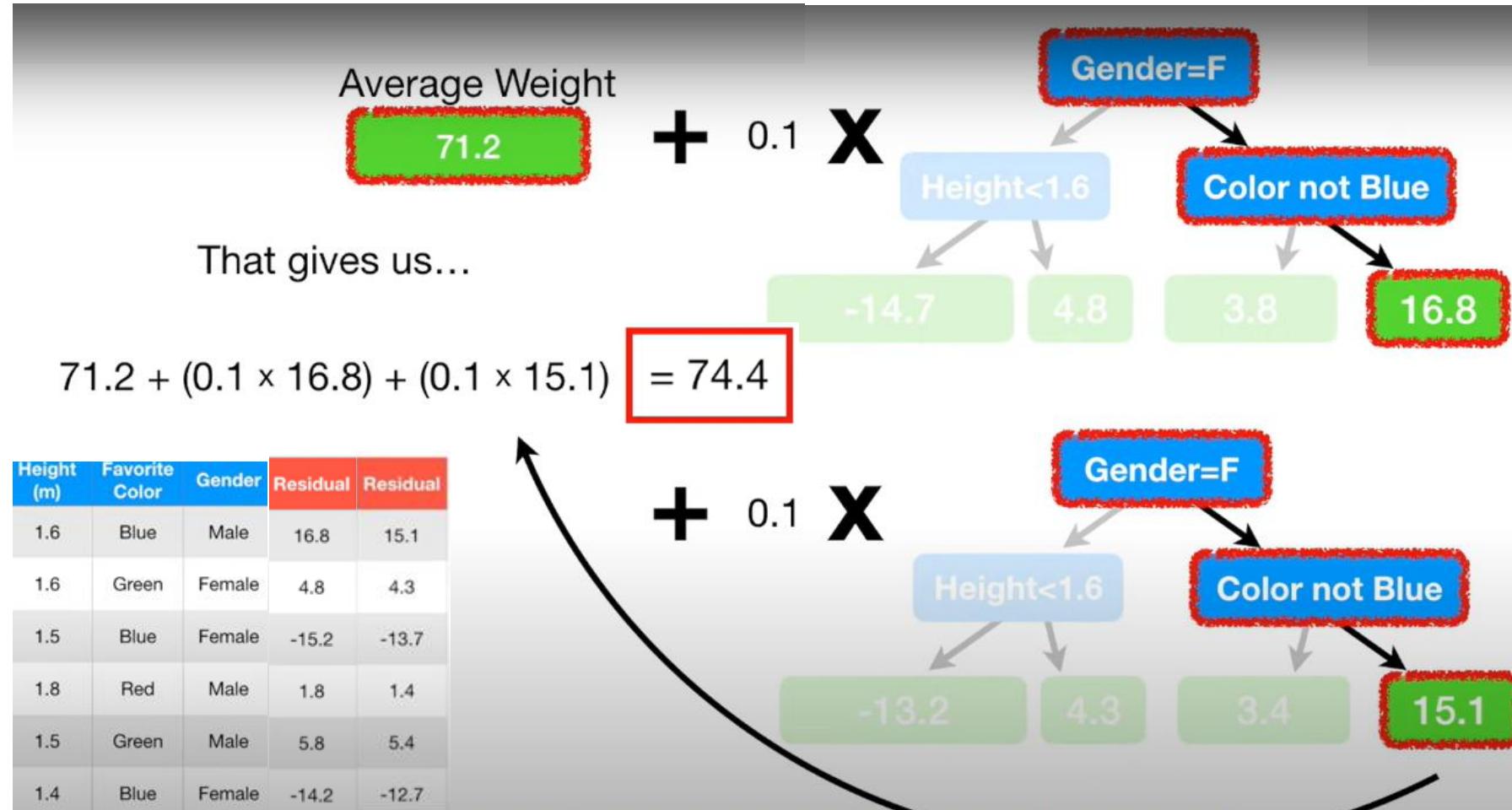
# Técnicas: 8. Gradient Boosting

Height (m)	Favorite Color	Gender	Weight (kg)	Residual
1.6	Blue	Male	88	15.1
1.6	Green	Female	76	4.3
1.5	Blue	Female	56	-13.7
1.8	Red	Male	73	1.4
1.5	Green	Male	77	5.4
1.4	Blue	Female	57	-12.7

And here's the new tree!



# Técnicas: 8. Gradient Boosting



# Técnicas: 8. Gradient Boosting

