2. SUPERVISED LEARNING

CURSO

Data Science Machine Learning & Deep Learning con Python

Naive Bayes

Naive Bayes

- Teorema de Bayes $$P(Y/X) = \frac{P(X/Y) * P(Y)}{P(X)}$$

Información conocida

$P(Y)$

$P(X/Y)$

Información inferida

$P(Y/X)$

$$P(\text{Clase} \mid \text{Datos}) = \frac{P(\text{Datos}|\text{Clase}) * P(\text{Clase})}{P(\text{Datos})}$$

Naive Bayes

- Se basa en el teorema de Bayes.
- Supuesto "Naive": Independencia entre las características, dada una clase.

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y) P(x_1, \ldots x_n \mid y)}{P(x_1, \ldots, x_n)}$$

Naive Bayes

- Dado el supuesto:

$$P(x_i | y, x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) = P(x_i | y),$$

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y) \prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, \ldots, x_n)}$$

Naive Bayes

- La condición de optimización es:

$$P(y \mid x_1, \ldots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i \mid y)$$

$$\Downarrow$$

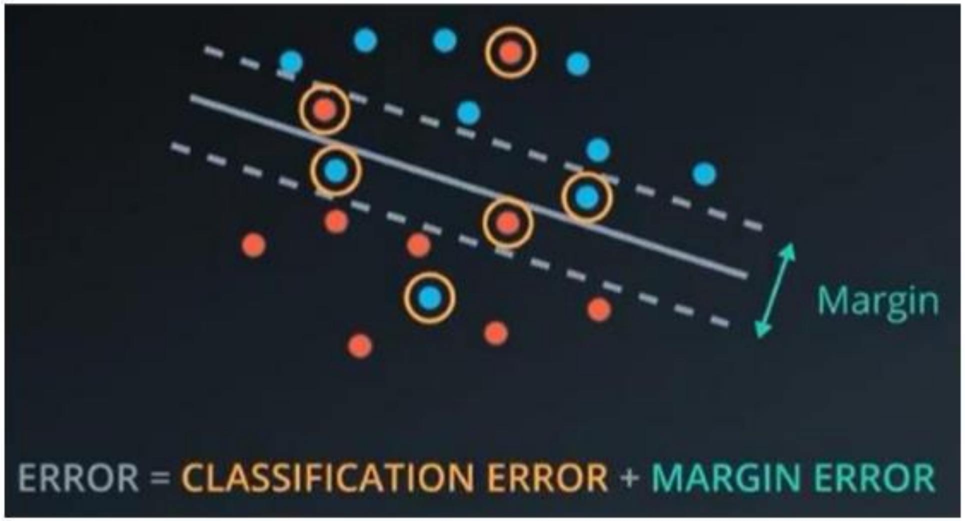
$$\hat{y} = \arg\max_y P(y) \prod_{i=1}^{n} P(x_i \mid y),$$

Support Vector Machine

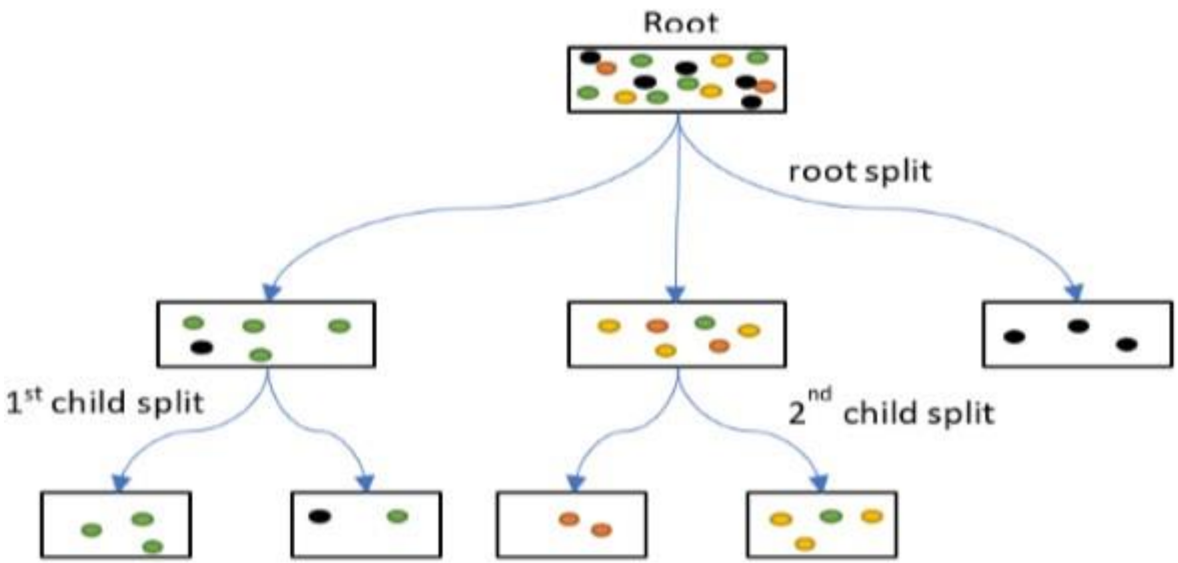
- Maximizar la distancia entre las categorías y la recta que las separa.

SVM

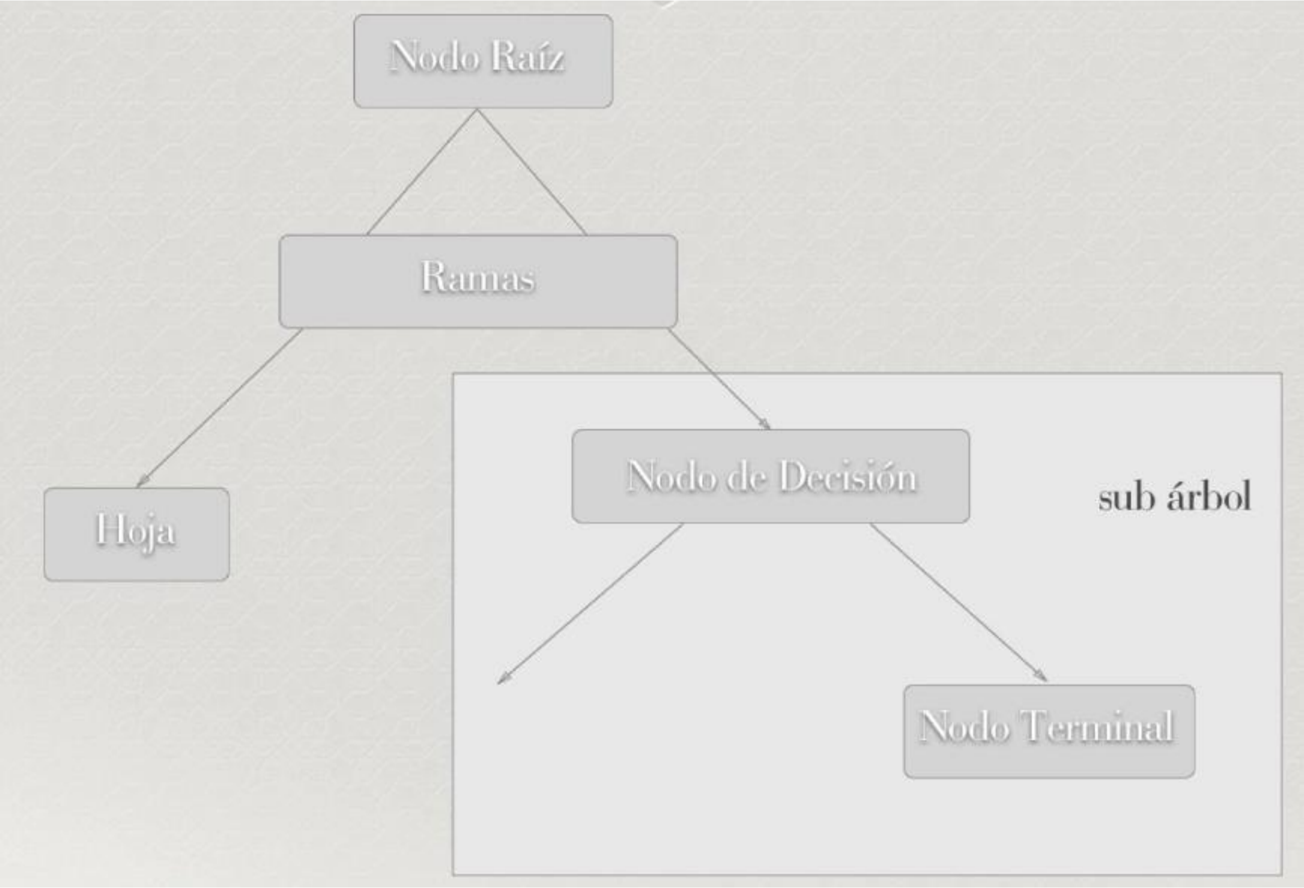
- Su función objetivo considera dos tipos de errores: Clasificación y margen.



Árboles de Decisión


Root
root split
1st child split
2nd child split

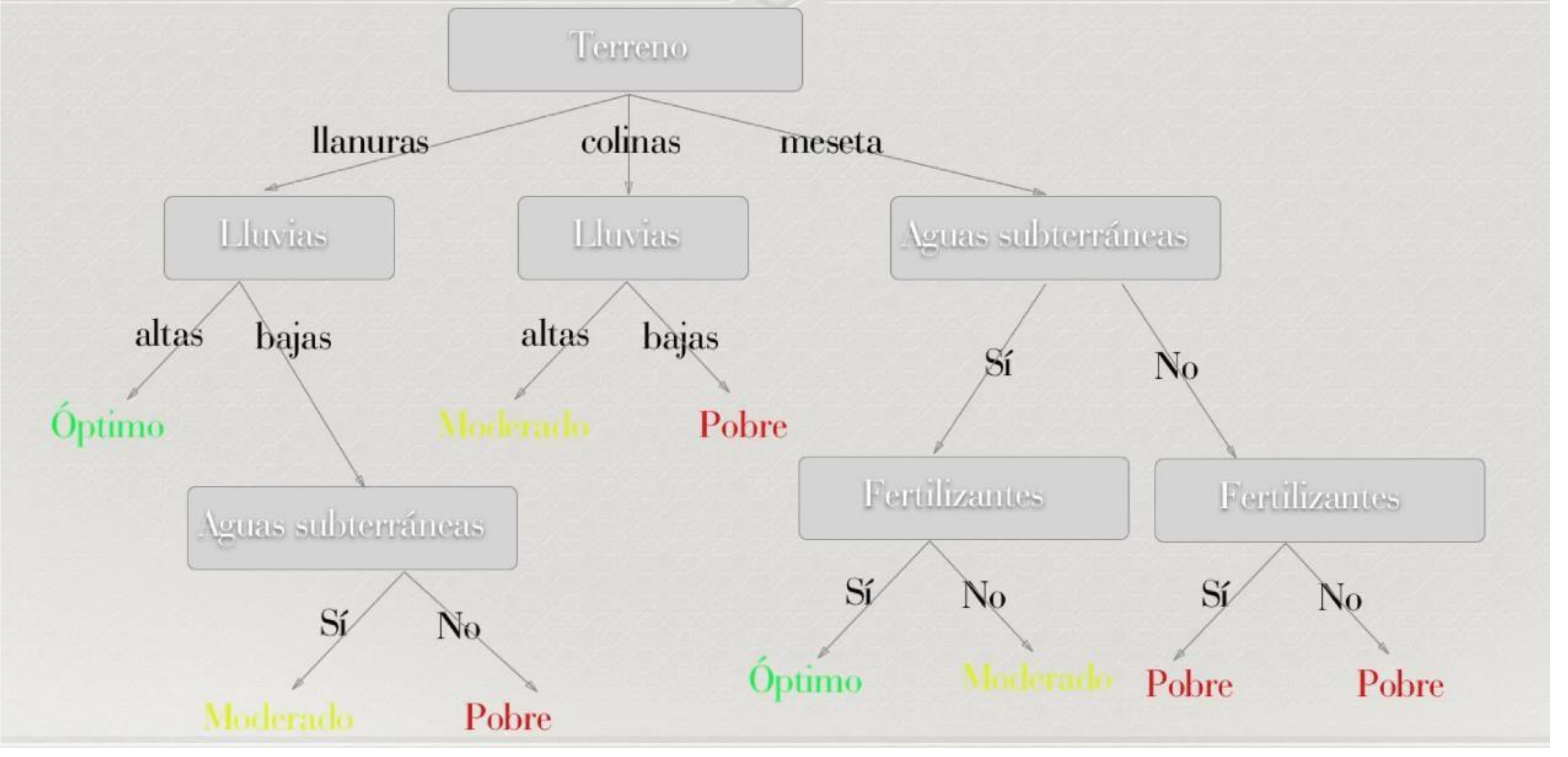
Componentes de un arbol



Diplomados Perú

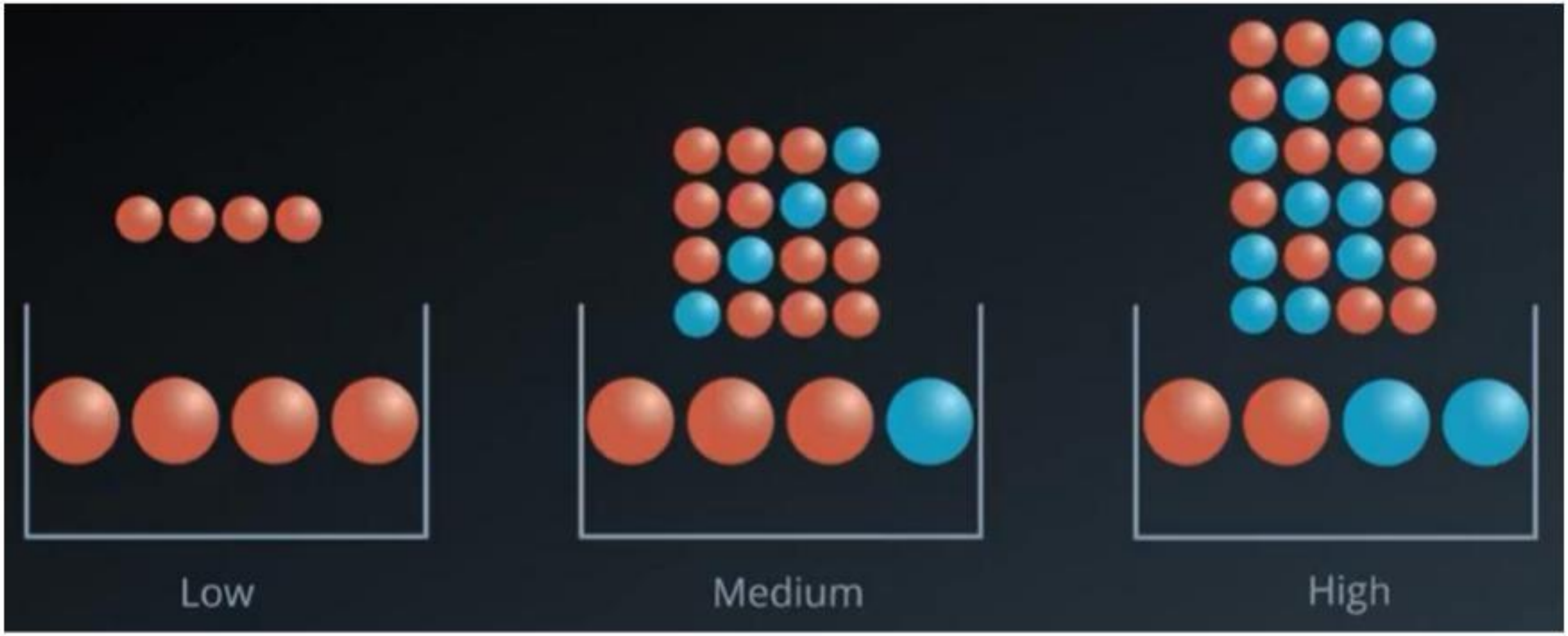
Lugar	Llúvias	Terreno	Fertilizantes	Aguas Subterráneas	Cosecha
P1	Altas	Llanura	Si	Si	Óptima
P2	Bajas	Colinas	No	Si	Pobre
P3	Bajas	Meseta	No	Si	Moderada
P4	Altas	Meseta	No	Si	Moderada
P5	Altas	Llanura	Si	No	Óptima
P6	Bajas	Colinas	No	No	Pobre
P7	Bajas	Meseta	No	No	Pobre
P8	Medias	Meseta	No	No	Pobre
P9	Altas	Colinas	Si	Si	Moderada
P10	Medias	Meseta	Si	Si	Óptima
P11	Altas	Meseta	Si	No	Óptima
P12	Medias	Meseta	Si	No	Moderada
P13	Altas	Colinas	Si	No	Moderada
P14	Bajas	Llanura	Si	Si	Moderada
P15	Medias	Llanura	Si	No	Moderada
P16	Bajas	Llanura	No	No	Pobre
P17	Bajas	Colinas	Si	No	Pobre
P18	Medias	Meseta	No	No	Pobre
P19	Altas	Llanura	No	Si	Moderada
P20	Medias	Colinas	Si	Si	Moderada

Decisiones de un árbol



Árboles de Decisión

- Entropía



- Entropía: Controla la partición de los datos.
- Information Gain: Cantidad de información ganada por una variable debido a la observación de otra variable.

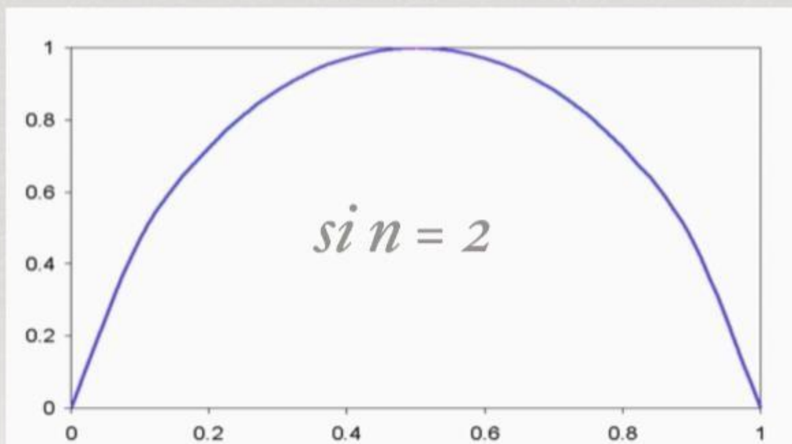
En las versiones más básicas, los árboles solo utilizaban variables explicativas categóricas.

Entropía

$$H(S) = \sum_{i=1}^{n} -p_i log_2(p_i)$$

n categorías para una variable objetivo a predecir del dataset

$H(S)$



Diplomados Perú

Cálculo de la entropía

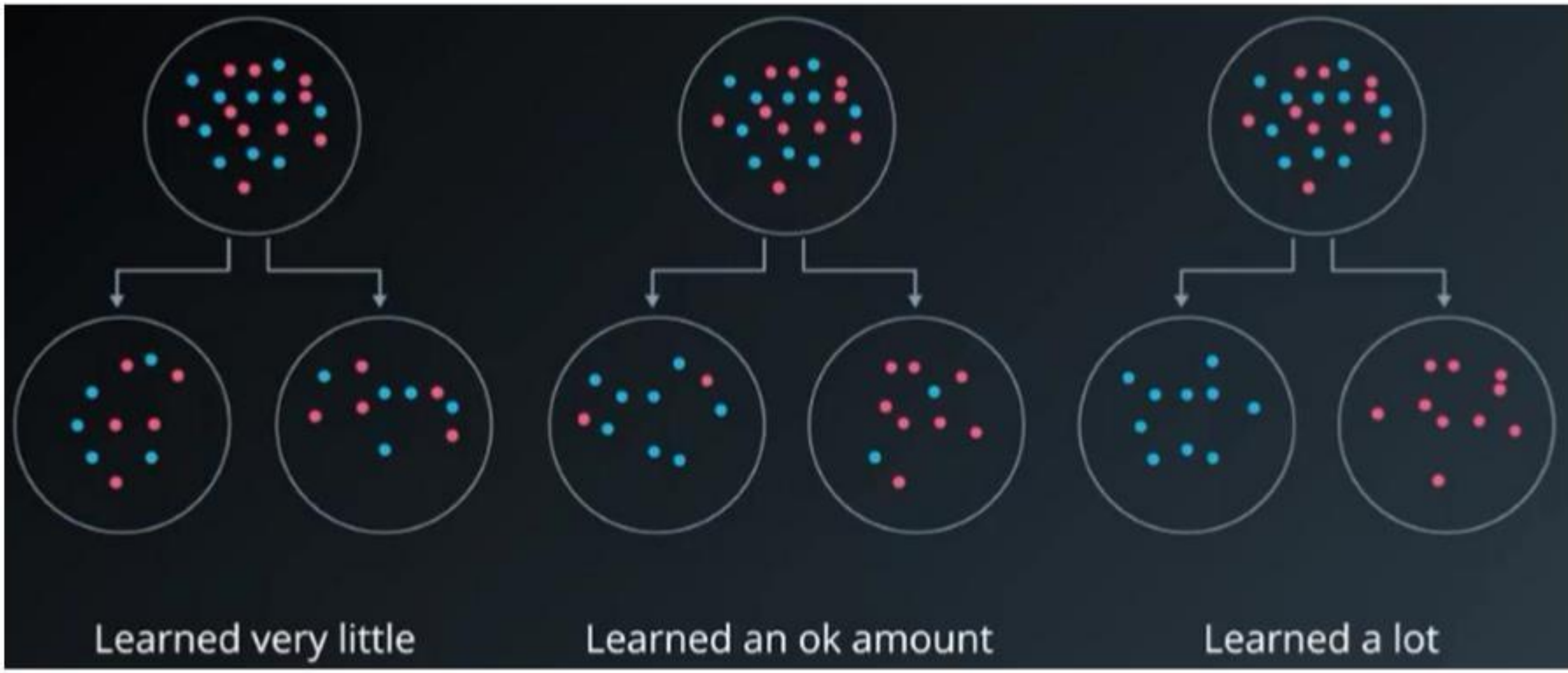
$$p_o = \frac{4}{20}$$

$$p_m = \frac{9}{20}$$

$$p_o = \frac{7}{20}$$

$$H(S) = -\frac{4}{20} \cdot log_2\left(\frac{4}{20}\right) - \frac{9}{20} \cdot log_2\left(\frac{9}{20}\right) - \frac{7}{20} \cdot log_2\left(\frac{7}{20}\right) = 1.5$$

- Information Gain



Cálculo de gain

$$\Delta H(S,V) = H(S) - \sum_{c \in V} \frac{|V=c|}{|V|} H(V=c)$$

$$Terreno = \{llanura, meseta, colina\}$$

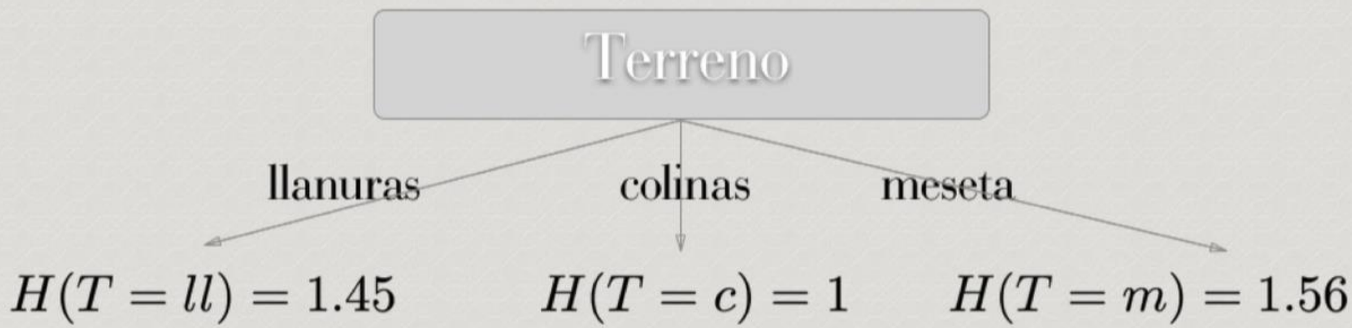
Terreno	Óptima	Moderada	Pobre
Colinas	0	3 (50%)	3 (50%)
Llanuras	2 (33%)	1 (17%)	3 (50%)
Mesetas	2 (25%)	3 (37.5%)	3 (37.5%)

$$H(T=c) = -\left(\frac{0}{6} log_2\left(\frac{0}{6}\right) + \frac{3}{6} log_2\left(\frac{3}{6}\right) + \frac{3}{6} log_2\left(\frac{3}{6}\right)\right) = 1$$

$$H(T=ll) = -\left(\frac{2}{6} log_2\left(\frac{2}{6}\right) + \frac{1}{6} log_2\left(\frac{1}{6}\right) + \frac{3}{6} log_2\left(\frac{3}{6}\right)\right) = 1.45$$

$$H(T=m) = -\left(\frac{2}{8} log_2\left(\frac{2}{8}\right) + \frac{3}{8} log_2\left(\frac{3}{8}\right) + \frac{3}{8} log_2\left(\frac{3}{8}\right)\right) = 1.56$$

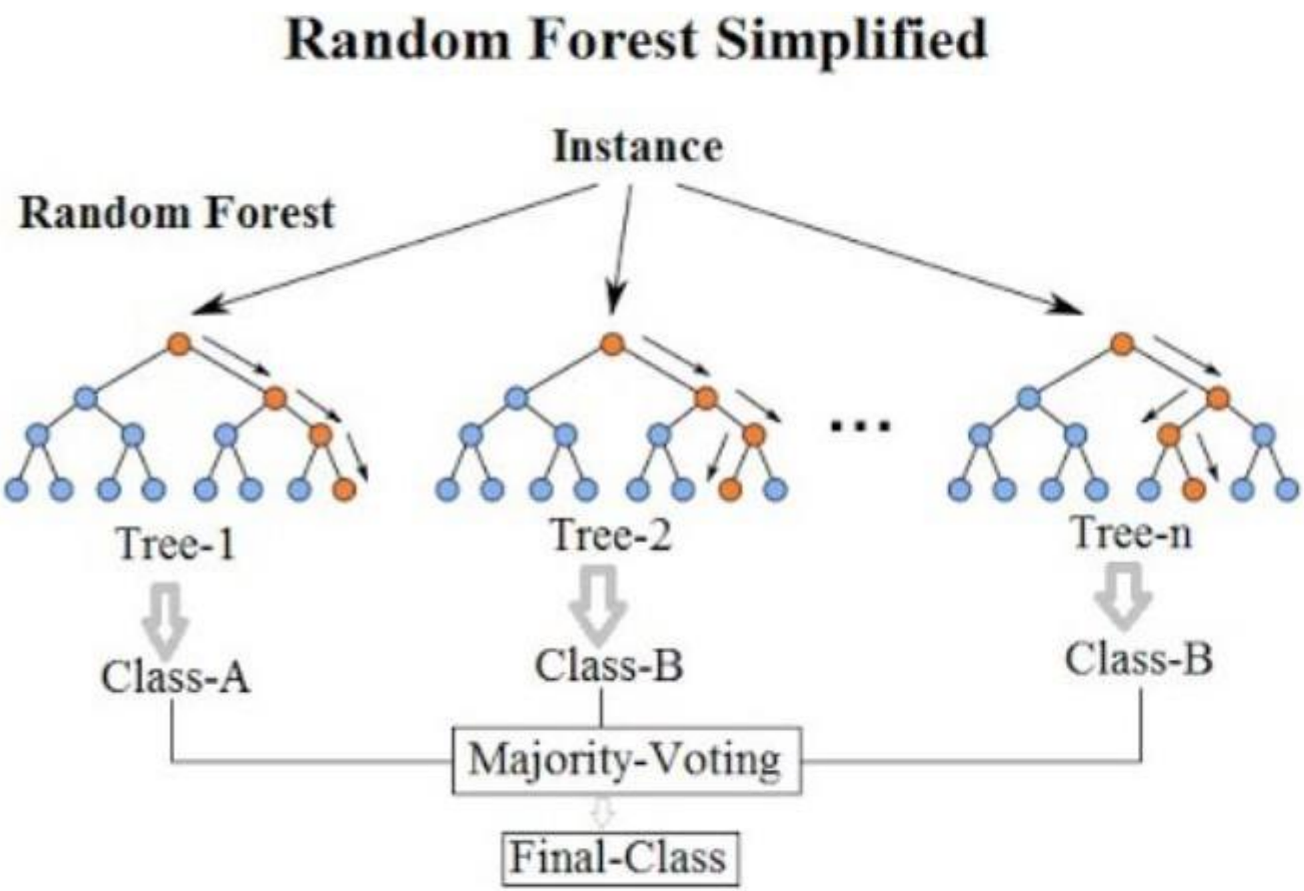
Cálculo de gain



$$\Delta H(S,T) = 1.5 - \left(\frac{6}{20} \cdot H(T = ll) + \frac{6}{20} \cdot H(T = c) + \frac{8}{20} \cdot H(T = m)\right) = 0.14$$

Random Forest

- Genera diversos árboles de decisión con submuestras, la regla de decisión final toma el promedio de los árboles generados.
- Usualmente el tamaño de las submuestras es el mismo que el de la muestra total.
- La ganancia de Random Forest, con respecto a los árboles de decisión tradicionales, es mayor precisión y evita el overfitting.


Random Forest Simplified
Instance
Random Forest
Tree-1
Tree-2
...
Tree-n
Class-A
Class-B
Class-B
Majority-Voting
Final-Class

XGBoost

- El criterio de optimización considera dos componentes: Trainning Loss y Regularization.

$$\text{obj}(\theta) = L(\theta) + \Omega(\theta)$$

$$L(\theta) = \sum_i (y_i - \hat{y}_i)^2$$

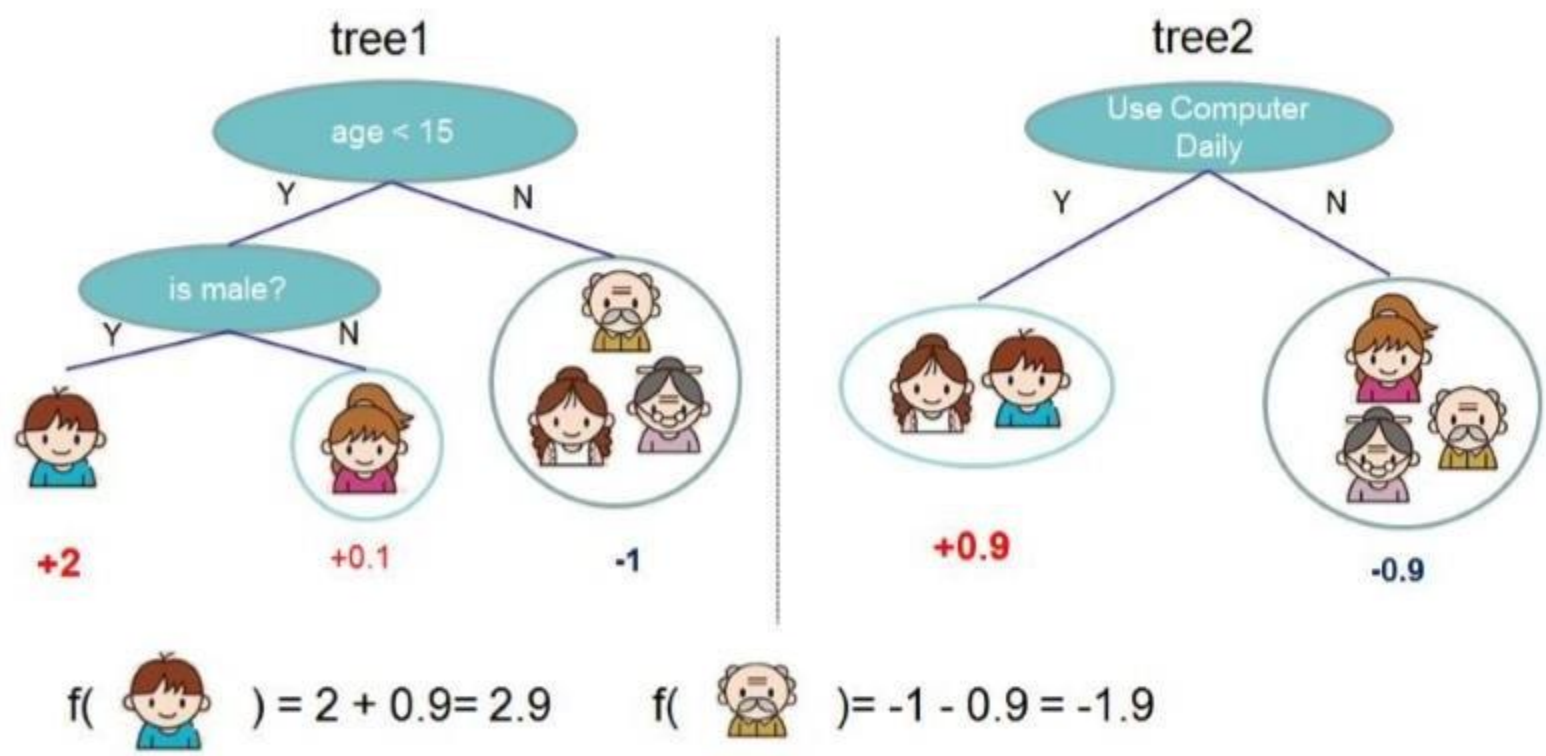
$$L(\theta) = \sum_i [y_i \ln(1 + e^{-\hat{y}_i}) + (1 - y_i) \ln(1 + e^{\hat{y}_i})]$$

$L(\theta) =$ Función que se desea optimizar

*L*a predicción es la suma de la predicción de varios arboles.

- Complejidad:
- $$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

T: cantidad de hojas del árbol
Wj: valor de la j-ésima hoja
Gamma y Lambda son hiper-parámetros



Las predicciones son la suma de scores por cada árbol www.diplomadosperu.com.pe