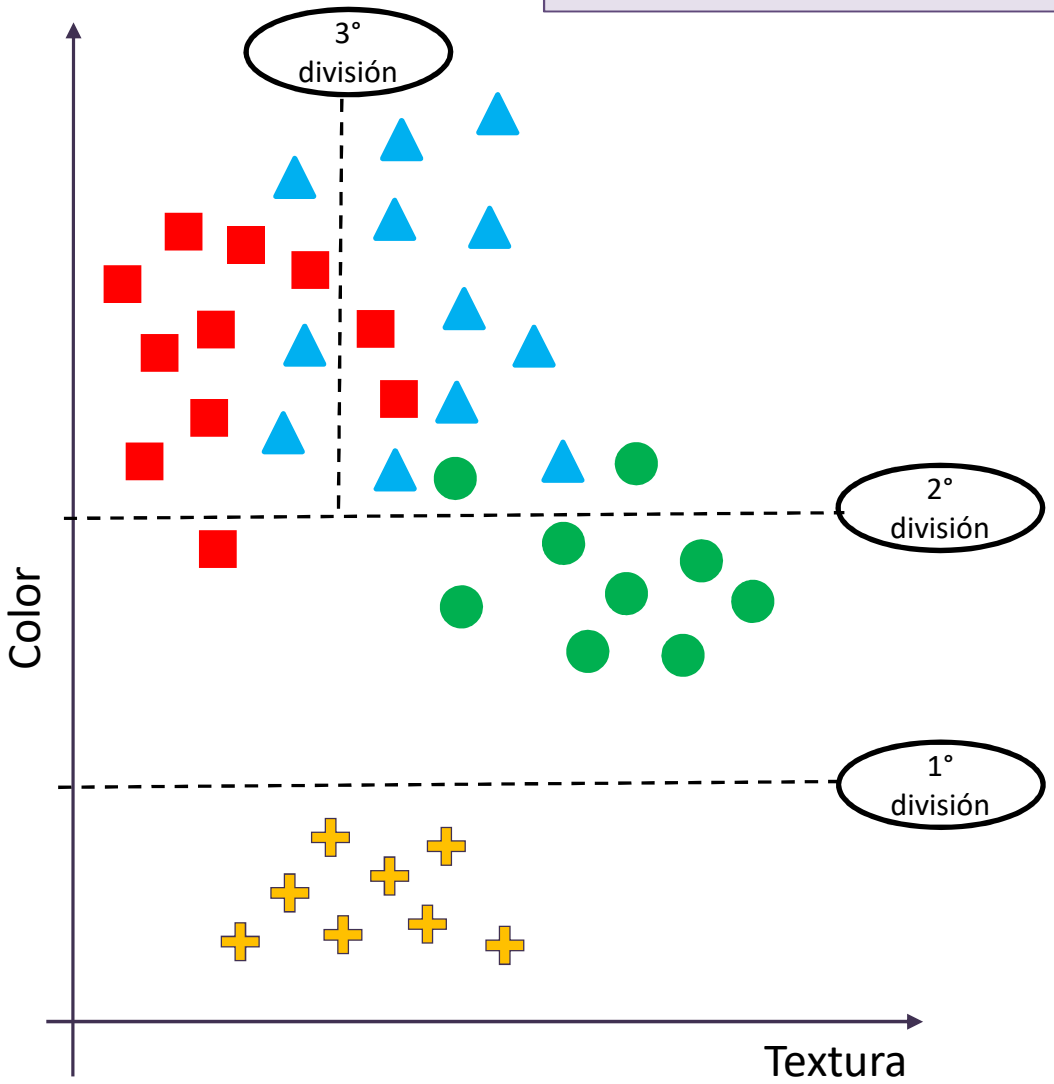
Árbol de Decisiones

Aprendizaje Automático

Tecnológico de Monterrey

Dr. Luis Eduardo Falcón Morales

Árbol de Decisiones Probabilístico



Mora (blackberry)

Arándano rojo (cranberry)

Fresa (strawberry)

Frambuesa (raspberry)

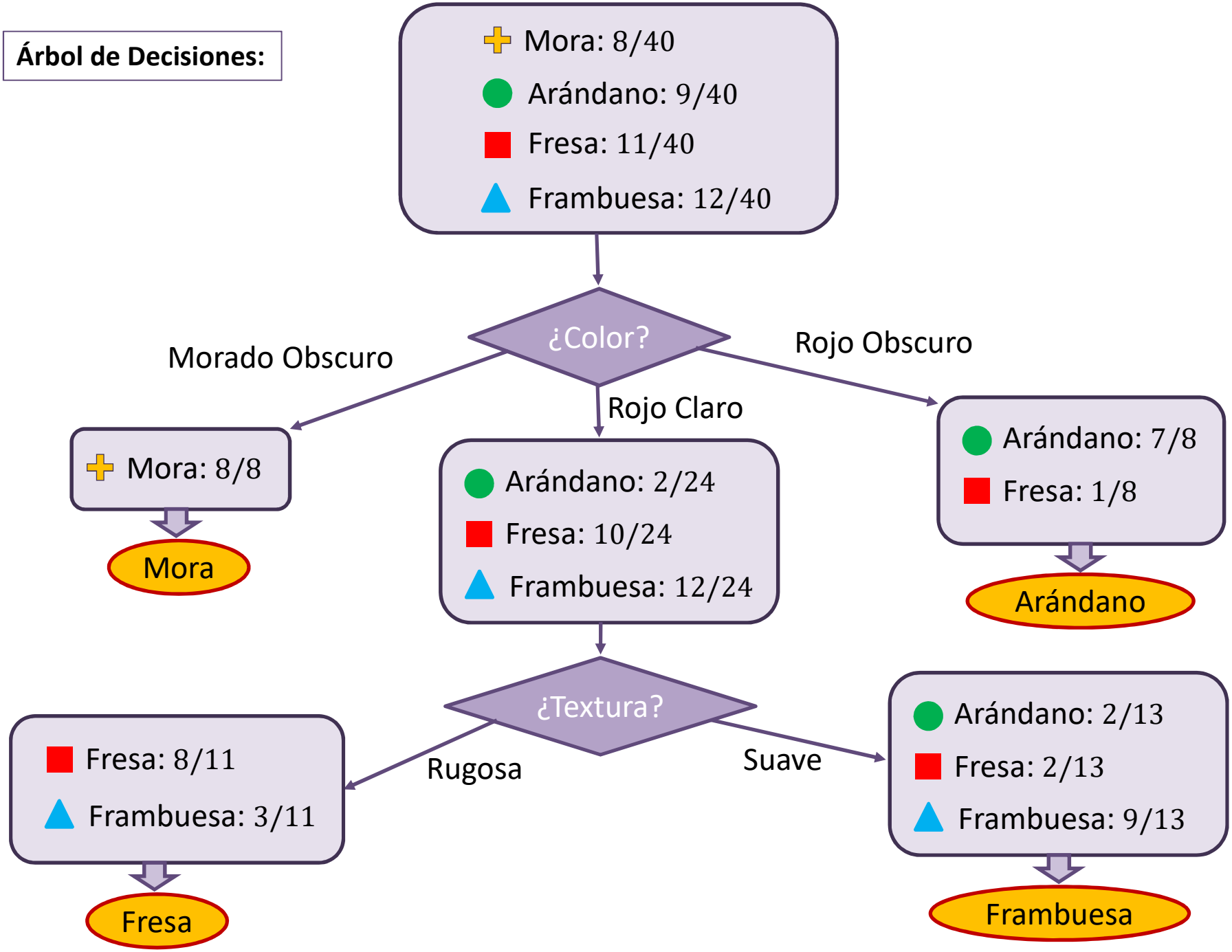
Foto de Ir Solypexelsanaya: https://www..com/es-es/foto/moras-en-la-mesa-892808/

Foto de Simon Berger: https://www.pexels.com/es-es/foto/frutas-de-cereza-966416/

Foto de David J. Boozer: https://www.pexels.com/es-es/foto/foto-de-primer-plano-de-fresa-1125122/

Foto de Viktor Talashuk: https://www.pexels.com/es-es/foto/foto-en-primer-plano-de-frambuesas-2682661/

Árbol de Decisiones:



Árbol de Decisiones

Divide y Vencerás:

- Dividir los datos en subconjuntos cada vez más pequeños, de manera que los datos de cada clase sean muy similares; pero evitando el sobreentrenamiento.

- Continuar con la división hasta que suceda alguna de las siguientes opciones:
  - Todos los casos de un nodo pertenecen a una misma clase.
  - Ya no hay más factores que dividir.
  - El árbol creció a un tamaño máximo predeterminado.

- **En cada paso se elegirá el factor con mayor entropía.** Así, se usan solamente los factores más importantes.

- Puede usarse con factores numéricos o nominales.

- El modelo/árbol resultante puede interpretarse relativamente fácil, es decir, sin conceptos matemáticos complejos.

Consideraciones:

- La división o separación de los subconjuntos en el sistema coordenado utiliza solamente rectas paralelas a los ejes coordenados, a diferencia por ejemplo del algoritmo SVM que puede recurrir a rectas con cualquier inclinación.

- Un inconveniente de este algoritmo es cuando existen muchos factores y cada uno con muchos niveles, ya que las ramificaciones crecen exponencialmente.

- Usualmente en primera instancia se consideran cuando muchos 3 ramificaciones en cada nodo, para evitar que crezca demasiado la complejidad del modelo.

- El árbol puede reducir (**pruning**) hacia los factores con mayor cantidad de niveles.

- Si no se eligen de manera adecuada las ramificaciones en cada nodo, el modelo puede resultar sub o sobre-entrenado.

- **Pequeños cambios en los datos de entrenamiento puede resultar en grandes cambios en las reglas de decisión del modelo.**

Árbol de Decisión para Problemas de Clasificación y Regresión (CART: Classification And Regression Tree)

- Los árboles de decisión pueden aplicarse tanto a problemas de clasificación, como de regresión.

- En general no es necesario introducir variables dummy o aplicar one-hot-encoding. Quedará a criterio del analista o la librería a utilizar.

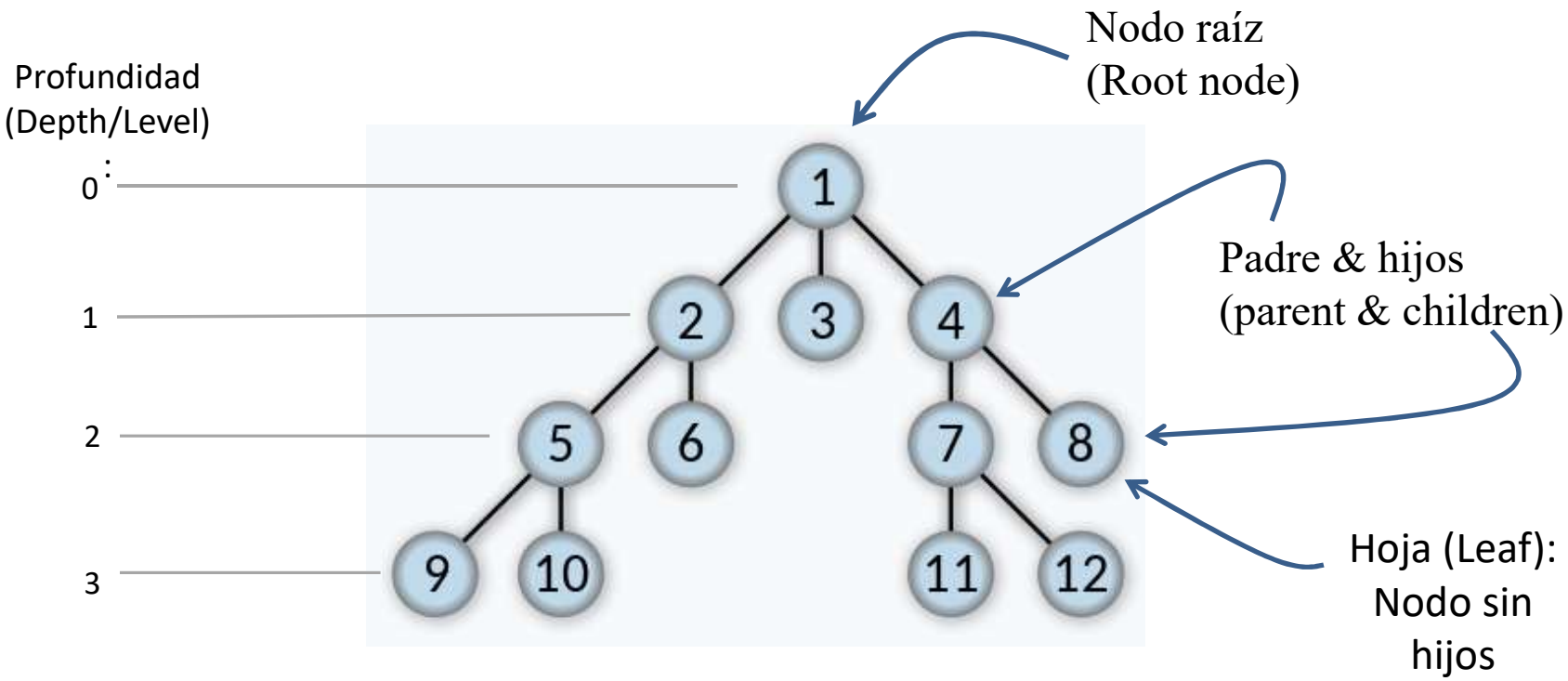
- Son métodos no paramétricos, es decir, no se requiere suponer algún tipo de distribución particular.

- El impacto de los valores extremos (outliers) se disminuye.

- **En un problema de clasificación**, la predicción de un dato nuevo se puede llevar a cabo por votación en cada hoja y donde el buen desempeño se mida con alguna métrica que nos diga qué tan homogeneo o que tanta entropía hay en cada hoja .

- **En un problema de regresión**, la predicción de un dato nuevo se puede llevar a cabo promediando las predicciones de los datos de entrenamiento que se encuentran en la misma ruta del árbol que el dato nuevo. El error puede medirse por ejemplo con MSE (mean squared error), donde cada valor real de salida en cada hoja se compara con el valor promedio de los datos en dicho nodo.

Grafo de un árbol de decision:



Inconvenientes de los árboles de decisión

- Son muy propensos al sobreentrenamiento y subentrenamiento.
- Si un factor categórico tiene una gran cantidad de niveles, el árbol es propenso a sesgarse al nivel de mayor frecuencia, sobre todo si la diferencia entre niveles es muy marcada. **Se recomienda por ello ajustar los niveles previamente.**
- Solamente lleva a cabo divisiones con rectas paralelas a los ejes coordenados (hiperplanos e hiperplanos coordenados, respectivamente).
- Pequeños cambios en el conjunto de entrenamiento, pueden implicar grandes cambios en el árbol obtenido y por lo tanto en las decisiones o reglas lógicas finales. **RandomForest disminuye este problema.**
- Árboles muy grandes son difíciles de interpretar o entender de manera intuitiva.
- Árboles con ramificaciones mayores a la binaria pueden llegar a ser muy costosos computacionalmente.

Maestría en Inteligencia Artificial Aplicada

Métricas para la mejor ramificación en un Árbol de Decisión

Inteligencia Artificial y Aprendizaje Automático

Tecnológico de Monterrey

Dr. Luis Eduardo Falcón Morales

Para construir un árbol de decisiones, necesitamos encontrar un criterio que nos ayude a determinar cuál factor dividir en cada nodo y con qué valor de umbral. Existen varias métricas y criterios que iremos viendo, pero para ello definamos algunos conceptos previos.

- Un subconjunto de datos se considera **puro (pureza)** si contiene solamente datos de una sola clase.

- La **entropía** es uno de los criterios más usados para medir la impureza o pureza de un subconjunto.

- **Valores cercanos a 0 nos hablan de un subconjunto muy puro o muy homogéneo: mínima entropía**. Un subconjunto de mínima entropía no es un buen conjunto de entrenamiento; pero en caso de ser un subconjunto en una de las hojas de un árbol de decisión, indica un buen proceso previo de clasificación.

- **Valores cercanos a 1 nos hablan de una mayor cantidad de desorden: máxima entropía.** Un subconjunto de máxima entropía es un buen conjunto de entrenamiento.

Teoría de la Información

- Claude Shannon (1916-2001) introdujo el concepto de entropía en el área de teoría de la información.
- Demostró que el álgebra booleana puede construir cualquier relación lógico-numérica de los circuitos digitales, 1937.
- Así, ayudó a sentar las bases de la teoría del diseño del ordenador digital y los circuitos digitales.
- Fundador de la Teoría de la Información, 1948.
- Contribuyó durante la Segunda Guerra Mundial al campo del criptoanálisis.
- En el área de la Inteligencia Artificial realizó una de las primeras publicaciones donde se describía la programación para que una computadora jugara ajedrez, 1949.

Proceso de Bernoulli

Un proceso de Bernoulli es una sucesión finita o infinita numerable de variables aleatorias independientes binarias $X_1, X_2, \ldots$ tal que todas tienen la misma probabilidad p de éxito.

Por ejemplo:

X_1	X_2	X_3	$\cdots$	X_k	$\cdots$
1	1	0	$\cdots$	1	$\cdots$

donde $P(X_j = 1) = p, \ \forall\, j.$

Teoría de la Información: Entropía

Sistema de comunicación de datos:

Objetivo: que el receptor pueda recuperar el conjunto de datos emitido por el emisor a través de un canal.

La entropía nos proporciona un límite absoluto sobre la longitud promedio mínima que es posible compactar un mensaje del emisor, y que aún pueda ser recuperada por el receptor.

Así, si la capacidad de comunicación del canal de transmisión es mayor que la entropía del emisor (en bits), los datos transmitidos podrán ser recuperados por el receptor de manera confiable… al menos en teoría.

Uso de Logaritmos

Debido a la conocida propiedad aditiva de los logaritmos:

$$\log(xy) = \log(x) + \log(y)$$

esta función se utilizará como medida de la entropía, para ayudar a mejorar el tratamiento de las probabilidades de eventos independientes.

Así, para variables aleatorias independientes $X_1, X_2, \ldots, X_m$ con $P(X_j = x_j) = p_j$, donde $j \in \{1, 2, \ldots, m\}$, tendríamos para el caso del evento $X_1 \cap X_2 \cap \cdots \cap X_m$:

$$P(X_1 \cap X_2 \cap \cdots \cap X_m) = p_1 p_2 \cdots p_m$$

Y al aplicar el logaritmo:

$$\log(p_1 p_2 \cdots p_m) = \log(p_1) + \log(p_2) + \cdots + \log(p_m)$$

Y si $p_1 = p_2 = \cdots = p_m = \frac{1}{m}$

$$\log(p_1 p_2 \cdots p_m) = \log(p_1) + \cdots + \log(p_m) = m \log\left(\frac{1}{m}\right) = -m \log(m)$$

Recuerda**: la entropía** nos habla sobre la **longitud promedio mínima** que es posible reducir en un mensaje y aún así, poder ser recuperado por el receptor.

Ejemplo:

Analicemos cómo debiera ser entonces la Entropía en el caso de una moneda:

Consideremos una moneda con dos estados posibles equiprobables: H, T.

Entonces:

Si lanzamos 1 vez la moneda,
su entropía deberá ser de 1 bit:

$$\log_2(2^1) = 1$$

Si lanzamos 2 veces la moneda,
su entropía deberá ser ahora de 2 $bits$:

$$\log_2(2^2) = 2$$

Y así, en general:

Si lanzamos m veces la moneda,
su entropía deberá ser de m $bits$:

$$\log_2(2^m) = m$$

O bien, si $p = \frac{1}{2^m}$, entonces: $-\log_2(p) = m$

Así, el Valor Esperado (promedio) de esta cantidad de bits, nos llevará a la definición de la fórmula de la entropía.

Shannon define la **cantidad de información** (***self-information***) como:

$$I(X) = -\log_2(p)$$

es decir, la cantidad de "**bits**" de información que recibimos del evento X.

Ejemplos:

- Al lanzar una moneda "legal" 3 veces, la cantidad de información de este evento sería:

$$-\log_2\left(\frac{1}{2}\cdot\frac{1}{2}\cdot\frac{1}{2}\right) = -\log_2\left(\frac{1}{2^3}\right) = 3 \text{ bits}$$

  Así, decimos que la cantidad de información es 3 bits, lo cual significa en este caso que se tienen $2^3 = 8$ resultados posibles al lanzar 3 veces una moneda.

- Un mensaje que contiene un solo carácter ASCII (de los 256 posibles) tendrá la siguiente cantidad de información:

$$-\log_2\left(\frac{1}{256}\right) = \log_2(2^8) = 8 \text{ bits}$$

Entropía: Sistemas de Información

De acuerdo a la definición de Shannon, la entropía H de una variable aleatoria discreta X con función masa de probabilidad $f(X)$, está dada mediante el siguiente valor esperado:

$$\mathrm{H}(X) = E\big[-\log\big(f(X)\big)\big]$$

Entonces, si la v.a. X tiene un dominio discreto $\{x_1, x_2, \dots, x_n\}$, con la función masa $f(X = x_k) = p_k$, entonces por la definición de valor esperado, la entropía estará dada como:

$$\mathrm{H}(X) = \sum_{k=1}^{n} -p_k \log(p_k)$$

Las unidades de la Entropía H se definen de acuerdo a la base b del logaritmo $\log_b$ utilizado:

- Si $b = 2$, unidades de bit (binary digit)
- Si $b = e$, unidades nat (natural unit of information)
- Si $b = 10$, unidades ban o dit (decimal digit)

Entropía de Información en un conjunto de varias Clases

Definimos la Entropía (también llamada entripía cruzada) de un conjunto de datos S como sigue:

$$Entropía(S) = \sum_{j=1}^{n_c} -p_j \log_2(p_j)$$

donde:

n_c : número de clases.

p_j : proporción de datos que caen en la clase del nivel j.

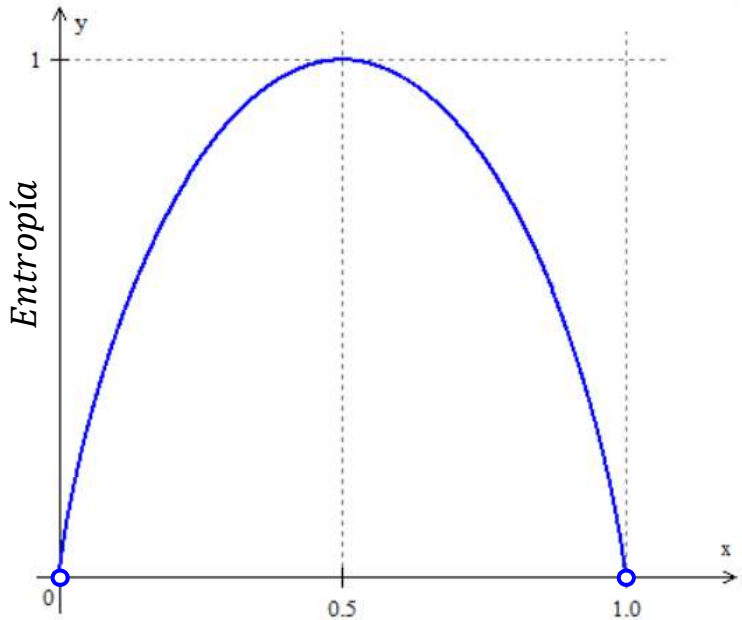
Nota:

- La entropía es no negativa: $Entropía \geq 0$
- La entropía está acotada superiormente: $Entropía(S) \leq \log_2(n_c)$

En particular, para un problema biclase, podemos graficar la entropía con la proporción x de uno de los factores, junto con la proporción $1 - x$ de su complemento:

$$\boldsymbol{Entropía}: \quad y = -x\log_2(x) - (1-x)\log_2(1-x)$$

donde $x \in (0, 1)$



Observamos que la máxima Entropía se obtiene con $x = 0.5$, es decir, cuando ambas clases tienen la misma probabilidad de suceder.

En cuanto una de las clases empieza a predominar, la Entropía se reduce.
Así, cuando la probabilidad de uno de los eventos es 0 o 1, la entropía es 0.

$$Entropía(S) = \sum_{j=1}^{n_c} -p_j \log_2(p_j)$$

- **Impureza máxima / Pureza mínima / Entropía máxima / Mayor incertidumbre**
- **Los subconjuntos de impureza máxima son adecuados para usarlos como conjuntos de entrenamiento.**

Ejemplo 1: Si A es un conjunto de dos clases equiprobables:

$$Entropía(A) = -\frac{1}{2}\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right) = \log_2(2) = 1$$

Ejemplo 2: Si B es un conjunto de tres clases equiprobables:

$$Entropía(B) = -\frac{1}{3}\log_2\left(\frac{1}{3}\right) - \frac{1}{3}\log_2\left(\frac{1}{3}\right) - \frac{1}{3}\log_2\left(\frac{1}{3}\right) = \log_2(3) = 1.5850$$

$$Entropía(S) = \sum_{j=1}^{n_c} -p_j \log_2(p_j)$$

- **Impureza mínima / Pureza máxima / Entropía mínima / Mayor información**
- **Los subconjuntos de impureza mínima NO son adecuados para usarlos como conjuntos de entrenamiento, ya que no tendrían suficiente información de las diferentes clases que se desean diferenciar.**

Ejemplo 3: Si C es un conjunto de una sola clase de elementos:

$$Entropía(C) = -1 \cdot \log_2(1) = 0$$

Ganancia de Información

Dado un conjunto de entrenamiento, ¿con qué factor debo iniciar la división de las ramas del árbol de decisión para obtener la mejor discriminación/diferenciación de los datos?

La ganancia de información de cada factor nos ayudará a determinar cual de ellos es el más importante para discriminar.

La Ganancia de Información GI al particionar un conjunto Ω con respecto a un factor (variable) F, está dado como sigue, si la partición es $\Omega = \bigcup_{j=1}^{n} \Omega_j$:

$$GI(F) = Entropía\left(S_{padre(\Omega)}\right) - EntropíaPonderada\left(S_{hijos}\right)$$

donde, si ω_j es la proporción de casos de Ω que caen en la partición Ω_j para cada hijo j, entonces:

$$EntropíaPonderada\left(S_{hijos}\right) = \sum_{j=1}^{n} \omega_j \, Entropía\left(\Omega_j\right)$$

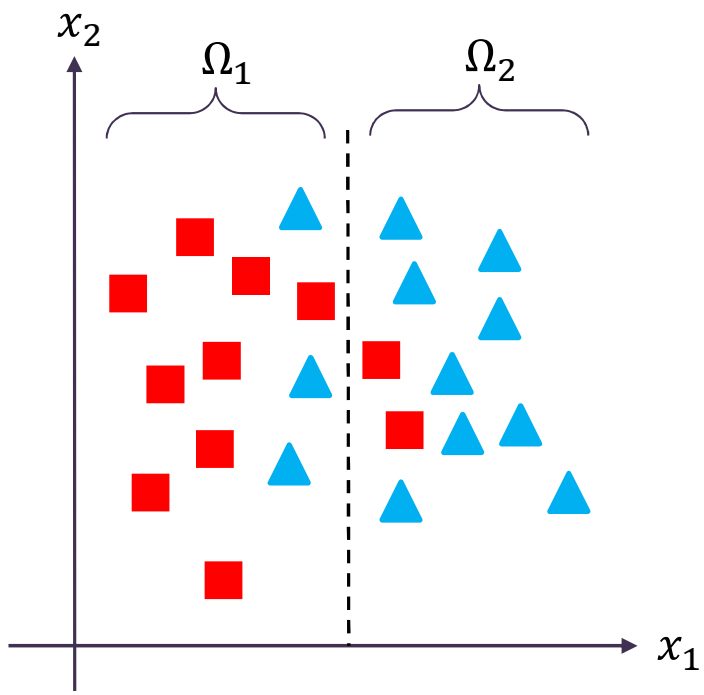
Es decir, la entropía de los hijos se obtiene sumando la entropía de cada una de las n particiones Ω_j, ponderadas por la proporción ω_j de casos en el conjunto de entrenamiento que caen en dicha partición.

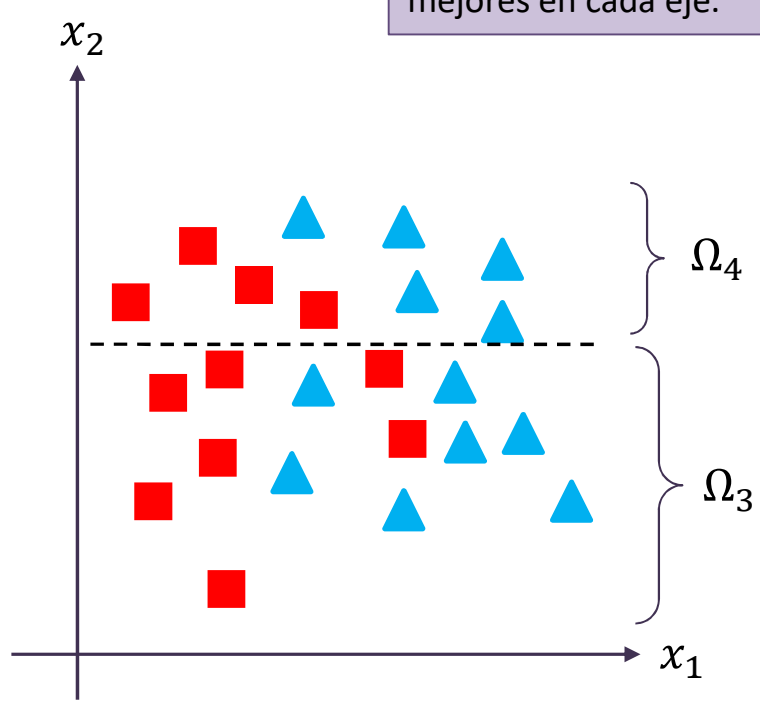
Ejemplo:

Supongamos que tenemos el conjunto de datos mostrado en la imagen y deseamos aplicar el método del árbol de decisiones para obtener una manera de clasificar ambos conjuntos.

Aunque también habría que valorar cuál de todas las rectas verticales u horizontales son las mejores en cada eje.

¿Cómo nos convendría iniciar la partición?
¿Con respecto al eje x_1 o con el eje x_2?



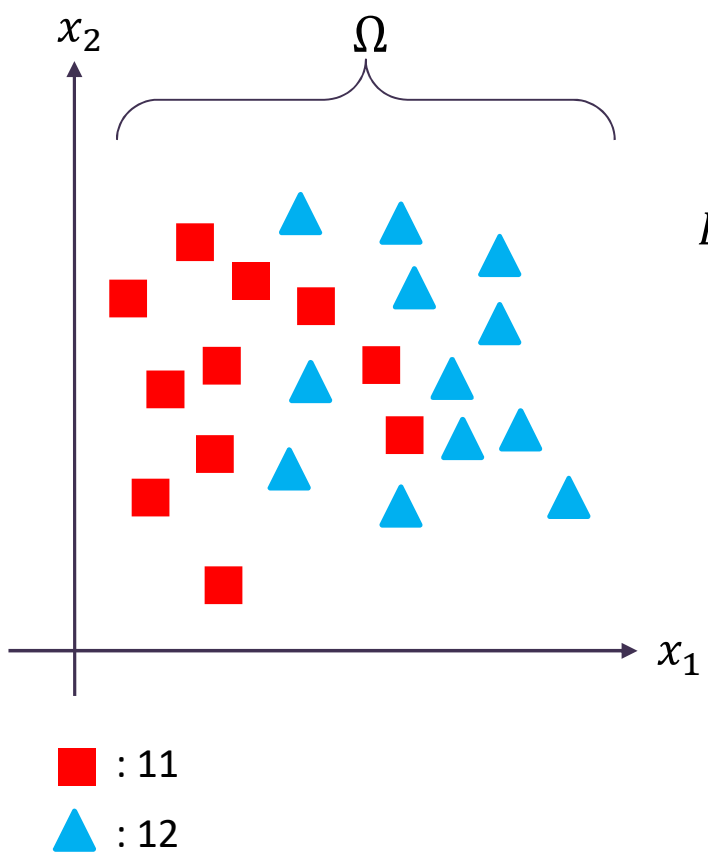


Supongamos que se eligen por el momento al azar las rectas indicadas en cada factor, ¿cuál de estas dos particiones lleva a cabo una mejor separación de las clases?

Ejemplo (cont):

Para decidir por cual de las dos particiones iniciar, debmos medir cuál de estas dos lleva mejor la separación de las dos clases y para ello nos apoyaremos en el valor de la Ganancia de Información (GI) de cada caso.

Calculemos primero el valor GI con respecto a la partición en el factor x_1, es decir, $\Omega = \Omega_1 \cup \Omega_2$:



Para ello requerimos obtener primero el valor de la Entropía del conjunto Padre, es decir, antes de la partición:

$$Entropía(S_{padre}) = \sum_{j=1}^{n_c} -p_j \log_2(p_j)$$

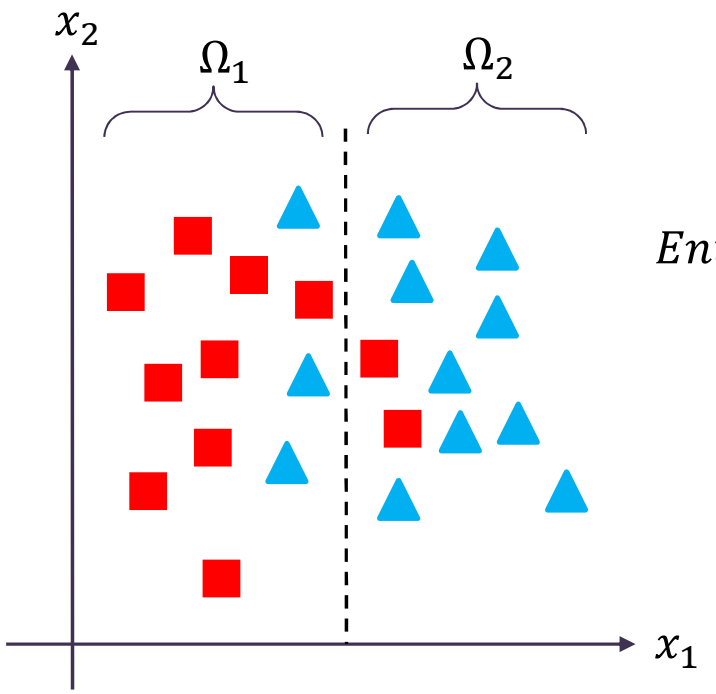
$$= -\frac{11}{23}\log_2\left(\frac{11}{23}\right) - \frac{12}{23}\log_2\left(\frac{12}{23}\right) = 0.9986$$

$$\therefore\ Entropía(S_{padre}) = 0.9986$$

Ejemplo (cont): Ahora calculamos la entropía después de la partición indicada en el eje del factor x_1, $\Omega = \Omega_1 \cup \Omega_2$:

$$Entropía(\Omega_1) = -\frac{9}{12}\log_2\left(\frac{9}{12}\right) - \frac{3}{12}\log_2\left(\frac{3}{12}\right) = 0.8113$$

$$Entropía(\Omega_2) = -\frac{2}{11}\log_2\left(\frac{2}{11}\right) - \frac{9}{11}\log_2\left(\frac{9}{11}\right) = 0.6840$$



De donde la Entropía Ponderada de los hijos es:

$$EntropíaPonderada(S_{hijos}) = \sum_{j=1}^{n} \omega_j \, Entropía(\Omega_j)$$

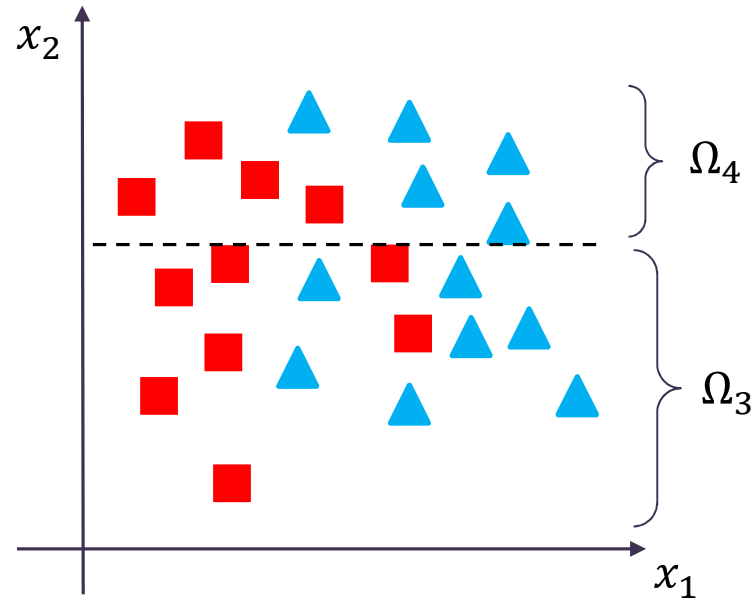
$$= \frac{12}{23}(0.8113) + \frac{11}{23}(0.6840) = 0.7504$$

Finalmente obtenemos la Ganancia de Información al realizar la división en x_1:

$$GI(F) = Entropía(S_{padre}) - EntropíaPonderada(S_{hijos}) = 0.9986 - 0.7504 = 0.2482$$

Ejemplo (cont):

Análogamente, calculemos ahora la Ganancia de Entropía con respecto al otro factor x_2 y de acuerdo a como se muestra en la figura:



- La entropía antes de la partición es la misma que el caso anterior:

$$Entropía(S_{padre}) = \sum_{j=1}^{n_c} -p_j \log_2(p_j) \approx 0.9986$$

- Ahora calculamos la entropía después de la partición:

$$Entropía(\Omega_3) = -\frac{7}{14}\log_2\left(\frac{7}{14}\right) - \frac{7}{14}\log_2\left(\frac{7}{14}\right) = 1$$

$$Entropía(\Omega_4) = -\frac{4}{9}\log_2\left(\frac{4}{9}\right) - \frac{5}{9}\log_2\left(\frac{5}{9}\right) \approx 0.9911$$

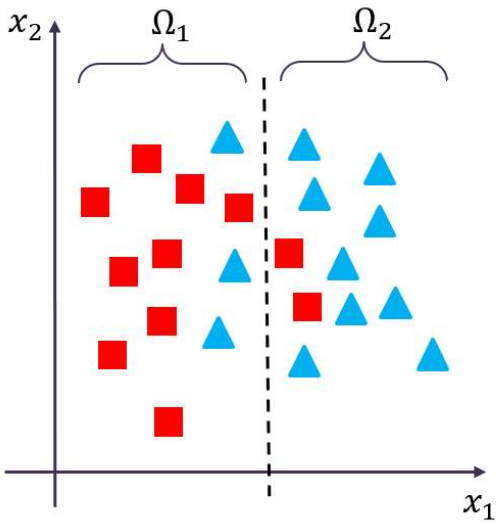
$$\begin{aligned} EntropíaPonderada(S_{hijos}) &= \sum_{j=1}^{n} \omega_j \, Entropía(\Omega_j) \\ &= \frac{14}{23}(1) + \frac{9}{23}(0.9911) \\ &= 0.9965 \end{aligned}$$

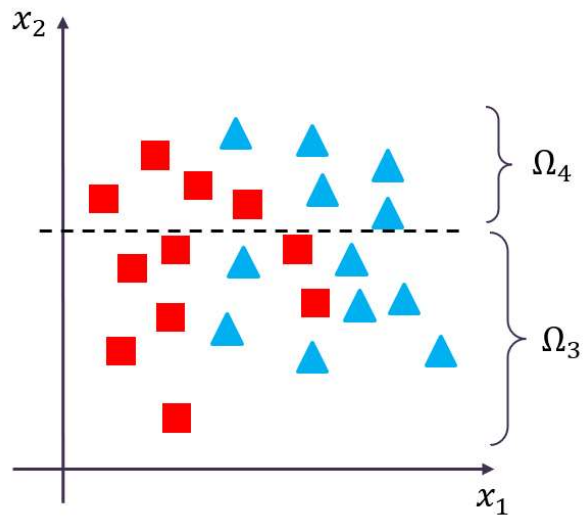
- Finalmente obtenemos la Ganancia de Información al realizar dicha división en x_2:

$$GI(F) = Entropía(S_{padre}) - EntropíaPonderada(S_{hijos}) = 0.9986 - 0.9965 = 0.0021$$

Ejemplo (cont): En resumen:

¿Cómo nos convendría realizar la partición?
¿Con respecto a la recta de x_1 o la recta de x_2?





En este caso, después de la partición sigue existiendo mucha entropía, es decir, no se gana casi nada con esta división.

Ganancia de Información al particionar con esta recta de x_1:

$$GI(x_1) = Entropía(S_{padre}) - Entropía(S_{hijos})$$

$$= 0.9986 - 0.7504$$

$$= 0.2482$$

Ganancia de Información al particionar con esta recta de x_2:

$$GI(x_2) = Entropía(S_{padre}) - Entropía(S_{hijos})$$

$$= 0.9986 - 0.9965 = 0.0021$$

$$= 0.0021$$

De los dos casos vemos entonces que conviene realizar la partición con respecto al factor x_1.

Maestría en Inteligencia Artificial Aplicada

Entropía vs Gini vs χ^2

Árbol de Decisión

Inteligencia Artificial y Aprendizaje Automático

Tecnológico de Monterrey

Dr. Luis Eduardo Falcón Morales

Otras métricas de la Ganancia de Información

Además del uso de la entropía con la Ganancia de Información existen otros criterios que se pueden utilizar para construir los árboles de decisiones.

Algunos de dichos criterios son el estadístico chi-cuadrada, las métricas Gini y la razón del error de clasificación.

Veamos algunas de ellas. Todas ellas se pueden usar en conjunto con el concepto de ganancia de información (GI).

Errores en modelos de Clasificación

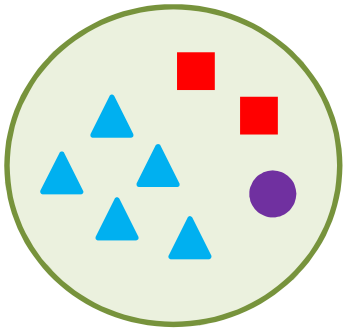
- **Razón del Error de Clasificación (Classification Error Rate):**

$$E_m = 1 - \max_k \{\hat{p}_{mk}\}$$

donde $\hat{p}_{mk}$ representa la proporción de observaciones del conjunto de entrenamiento en el nodo m, que pertenecen a la clase k.

Es una métrica relativamente sencilla e intuitiva, pero en general no es suficiente para generar buenos árboles de decisión.

Nodo m



$$\hat{p}_{m\blacktriangle} = \frac{5}{8} = 0.625 \qquad \hat{p}_{m\bullet} = \frac{1}{8} = 0.125$$

$$\hat{p}_{m\blacksquare} = \frac{2}{8} = 0.25$$

$$E_m = 1 - \max_k \{\hat{p}_{mk}\} = 1 - 0.625 = 0.375$$

- **Índice Gini (Gini Index):**

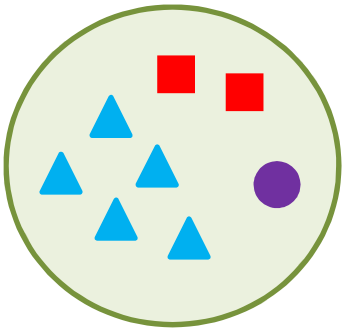
Esta métrica es una de las más utilizadas y se puede decir que mide la pureza o impureza de cada nodo m con respecto a las K clases del problema:

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

donde $\hat{p}_{mk}$ representa la proporción de observaciones del conjunto de entrenamiento del nodo m, que pertenecen a la clase k.

Observamos que en cuanto un nodo sea más puro (i.e., que contiene observaciones mayormente de una sola clase), su valor Gini es más pequeño.

Nodo m



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

$$= \frac{5}{8}\left(1 - \frac{5}{8}\right) + \frac{2}{8}\left(1 - \frac{2}{8}\right) + \frac{1}{8}\left(1 - \frac{1}{8}\right) = 0.53125$$

NOTA: Los nombres Gini "index" e "impurity" no son universales, y tanto autores como librerías las llaman de esta manera o al revés.

- **Impureza Gini (Gini Impurity):**

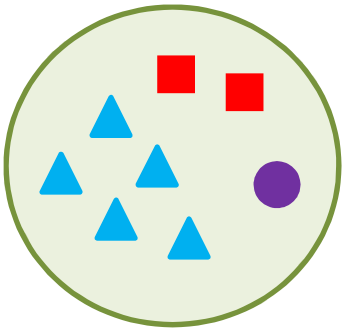
Una variante del índice Gini es mediante la llamada impureza de Gini, la cual de manera análoga mide la pureza o impureza de cada nodo m con respecto a las K clases del problema:

$$Gini_m = \sum_{k=1}^{K} (1 - (\hat{p}_{mk})^2)$$

donde $\hat{p}_{mk}$ representa la proporción de observaciones del conjunto de entrenamiento del nodo m, que pertenecen a la clase k.

Observamos que en cuanto un nodo sea más puro (i.e., que contiene observaciones mayormente de una sola clase), su valor Gini es más pequeño. Puede estar normalizada de 0 a 0.5, o bien de 0 a 1.

Nodo m



$$Gini_m = \sum_{k=1}^{K} (1 - (\hat{p}_{mk})^2)$$

$$= \left(1 - \left(\frac{5}{8}\right)^2\right) + \left(1 - \left(\frac{2}{8}\right)^2\right) + \left(1 - \left(\frac{1}{8}\right)^2\right) = 2.53125$$

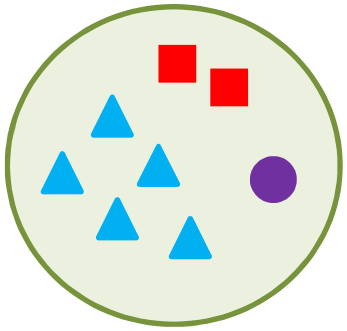
NOTA: Los nombres Gini "index" e "impurity" no son universales, y tanto autores como librerías las llaman de esta manera o al revés.

- **Entropía, Entropía Cruzada (Cross-Entropy) o log-loss:**

Finalemente como ya habíamos definido tenemos la entropía cruzada, o simplemente entropía, que basa su definición en el concepto de entropía de un sistema dada por Claude Shannon en sistemas de información: a mayor desorden/entropía en un nodo, mayor es su impureza.

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

Nodo m



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

$$= -\frac{5}{8} log_2\left(\frac{5}{8}\right) - \frac{2}{8} log_2\left(\frac{2}{8}\right) - \frac{1}{8} log_2\left(\frac{1}{8}\right)$$

$$= 1.2988$$

- **Chi-cuadrada : χ^2**

Se aplica una prueba de hipótesis para probar si existe diferencia significativa entre las observaciones del nodo padre y las observaciones del nodo hijo.

$$\chi^2 = \sum_k \frac{(observado_k - esperado_k)^2}{esperado_k}$$

Aplica solo para variables categóricas de salida, es decir, solo problemas de clasificación.

La hipótesis nula es que no existe diferencia significativa entre las observaciones del nodo padre y del hijo. Y la hipótesis alternativa es de que sí existe.

Así, **en cuanto mayor es el valor del estadístico χ^2, mayor evidencia de que existe diferencia significativa entre los nodos padre e hijo**. Es decir, que la partición realizada sí generó diferencia y conviene irse en esa dirección. Cuando no hay diferencia, significa que la partición realizada básicamente no separó los datos y no se generó ganancia de información.

Los árboles basados en esta métrica se denominan **CHAID**: Chi-square Automatic Interaction Detector.

Ejemplo con Chi-cuadrada

El uso de chi-cuadrada procede de manera análoga a la ganancia de información (GI), en la cual compara lo "observado" en los nodos hijos, en relación a lo "esperado" en esos mismos nodos, de acuerdo a la información del nodo padre. Veamos un pequeño ejemplo:

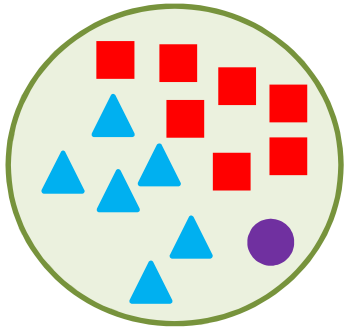
Porcentajes observados en el Padre:

$\frac{7}{14} \rightarrow 50\%$

$\frac{6}{14} \rightarrow 43\%$

$\frac{1}{14} \rightarrow 7\%$

Nodo Padre m_0



$$\chi^2 = \sum_k \frac{(observado_k - esperado_k)^2}{esperado_k}$$

$$= \left\{\frac{(2-4)^2}{4} + \frac{(5-3.44)^2}{3.44} + \frac{(1-0.56)^2}{0.56}\right\} +$$

$$+ \left\{\frac{(5-3)^2}{3} + \frac{(1-2.58)^2}{2.58} + \frac{(0-0.42)^2}{0.42}\right\}$$

$$= 2.053 + 2.721 = 4.774$$

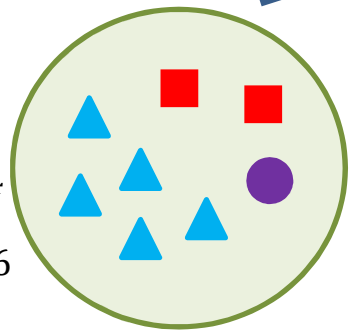
En cuanto mayor el valor de χ^2, mayor evidencia de que está separando mejor las clases. Entonces, si se hace otra partición diferente, elegir la que tenga el mayor valor de χ^2.

Esperado en el hijo m_1 :

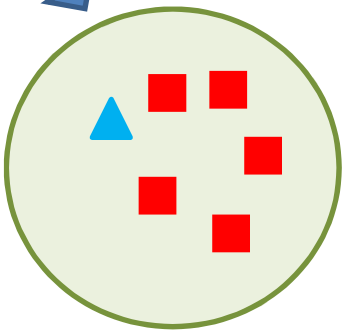
$0.5 * 8 = 4$

$0.43 * 8 = 3.44$

$0.07 * 8 = 0.56$



Nodo hijo m_1



Nodo hijo m_2

Esperado en el hijo m_2 :

$0.5 * 6 = 3$

$0.43 * 6 = 2.58$

$0.07 * 6 = 0.42$

Gini vs Entropía vs Chi-cuadrada :
¿Cuál es mejor o cuál utilizar en un problema de clasificación?

Existen diversos estudios sobre las ventajas y desvantajas de cada una de estas métricas para los problemas de clasificación. Mencionemos algunas de ellas:

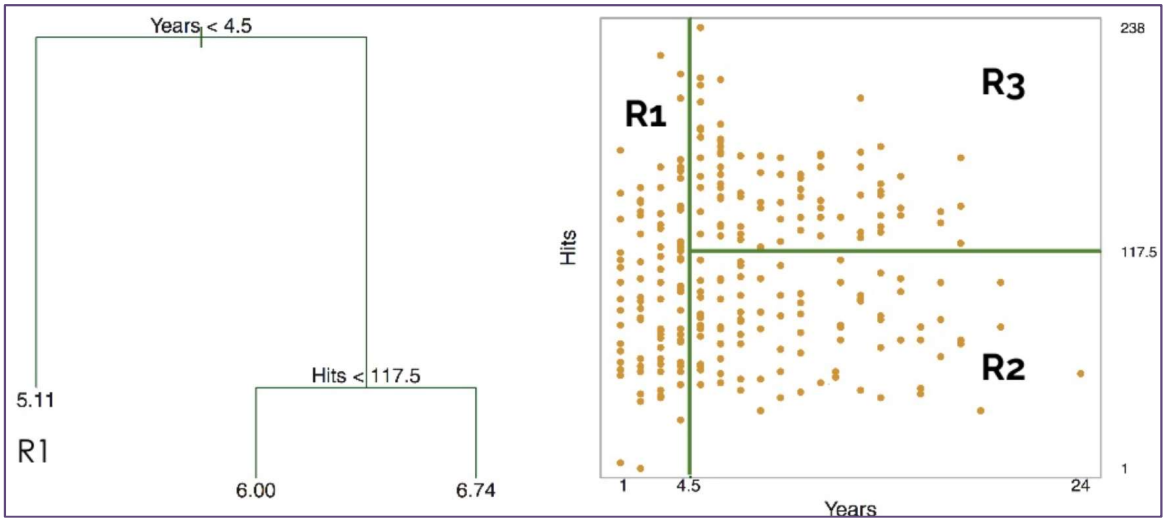
- Gini se comporta mejor con factores continuos
- Entropía cruzada se comporta mejor con factores categóricos
- Gini generalmente se decide por la clase/nivel mayoritario. Y se comporta mejor cuando se tiene una buena cantidad de registros.
- Entropía cruzada generalmente se decide por la agrupación de niveles que constituyan un 50% aproximadamente de los casos.
- Análisis muestran que las métricas Gini y Entropía generan usualmente resultados muy parecidos, con variaciones de un 2%.
- La Entropía es un poco más cosotos computacionalmente, por ello conviene usarlo cuando no se tienen muchos datos y el árbol no es muy profundo. Por lo tanto, Gini se puede usar en estos casos de muchos datos o árboles profundos.
- Chi-cuadrada es mejor cuando los factores tienen una distribución gaussiana.

Error en un Árbol de Decisión de Regresión

En un árbol de regresión el error de las predicciones se puede obtener mediante la suma de los cuadrados de los errores con respecto a las K regiones $R_1, R_2, \dots, R_K$ obtenidas en dicho árbol:

$$\min SSE = min \sum_{k=1}^{K} \sum_{j \in R_k} \left(y_j - \hat{y}_{R_k}\right)^2$$

donde $\hat{y}_{R_k}$ es la media de las predicciones de la variable de salida en la región R_k.



R_2 R_3

Pruning & Overfitting

Para evitar el sobre-entrenamiento (overfitting), una alternativa es buscar el sub-árbol T que reduzca el valor de la siguiente expresión (**cost-complexity pruning**):

NOTA: $|T|$ indica la cardinalidad o cantidad de nodos del árbol T.

$$SSE = \sum_{k=1}^{|T|} \sum_{j \in R_k} \left(y_j - \hat{y}_{R_k}\right)^2 + \alpha |T|$$

Scikit-Learn lo llama ccp_alpha

donde $|T|$ es el número de nodos terminales en el sub-árbol T y $\alpha \geq 0$ es el parámetro de ajuste.

A mayor cantidad de nodos terminales $|T|$ en el sub-árbol T, la suma de cuadrados disminuye, pero el sobre-entrenamiento (varianza) se incrementa. El sumando $\alpha|T|$ ayuda a controlar la cantidad de nodos terminales y evitar el sobre-entrenamiento.

El parámetro de ajuste α es otro hiperparámetro que deberá ser ajustado, usualmente con Cross-Validation.

Usualmente se inicia expandiendo el árbol a que todas las hojas sean de una sola clase o con un solo dato en cada hoja y a partir de ahí, reducir/podar (pruning) los nodos para obtener un sub-árbol que reduce el error.

Observa la analogía de dicha expresión con la regularización L_1.

Tomar en cuenta los siguientes puntos en los métodos basados en árboles de decisión:

- No extrapolan fuera del rango de valores del conjunto de entrenamiento en las variables de entrada.
- En general, las variables continuas se ven favorecidas a generar una mejor partición en un nodo con respecto a las variables categóricas.
- Al mismo tiempo las variables continuas pierden generalización al ser discretizadas durante las decisiones tomadas en cada nodo.
- Son sensibles a datos con clases no balanceadas o cuando se utilizan muy pocos árboles.
- La decisión tomada en un nodo, no considera el impacto que pueda tener (positivo o negativo) en los nodos posteriores.

Bosque Aleatorio
Random Forest

Tecnológico de Monterrey

Dr. Luis Eduardo Falcón Morales

- Random Forest es usado en problemas tanto de clasificación (RFClassifier) como de predicción (RFRegressor).

- RF está compuesto de árboles de decisión, cada uno generado de manera diferente:

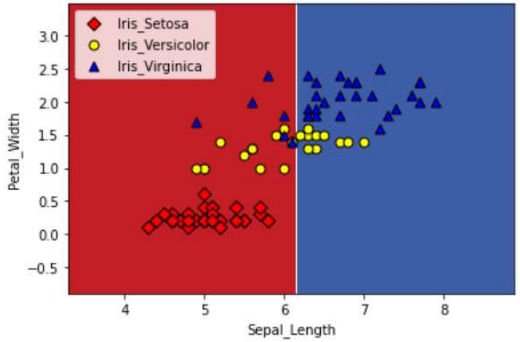
  - En particular, cada árbol de decisión del RF se genera seleccionando aleatoriamente un conjunto diferente de características (features).

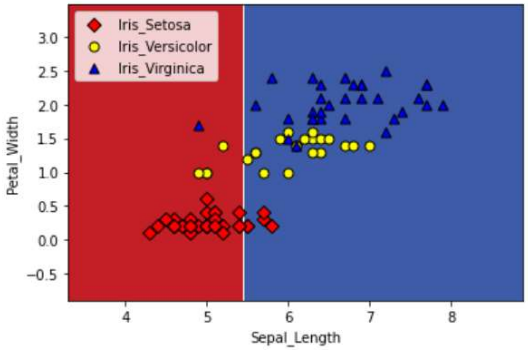
  - Existe una variante donde además en cada árbol se genera llevando a cabo un muestreo con reemplazo de los renglones. Es decir, se muestrean los renglones (observaciones) mediante la técnica llamada “Bootstrap”.

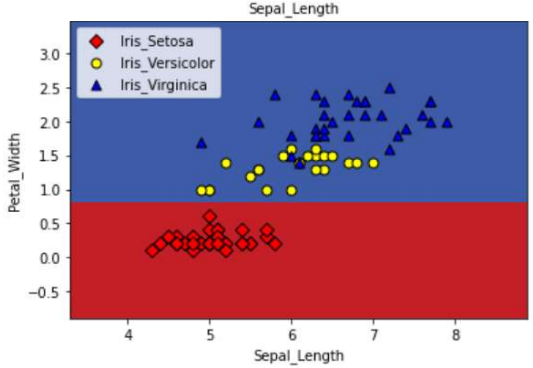
  - En general, como valor predeterminado la cantidad de árboles en un RF es de 10. A partir de ahí se puede buscar ajustarlo. En general puede variar a hasta cientos o miles.

  - Por lo general, se recomienda iniciar con una cantidad pequeña de árboles (“n_estimators” en sklearn-RandomForestClassifier) para ajustar los mejores hiperparámetros del modelo. Una vez obtenido los mejores hiperparámetros incrementar finalmente la cantidad de árboles en el RF, ya que esto siempre podrá mejorar o es su defecto, no afectarlo o incrementar el overfitting. Sin embargo, se debe cuidar el tiempo de cómputo que en general es muy costoso.

Por ejemplo, con los datos de Iris, considerando árboles con depth=1, algunos resultados posibles son:

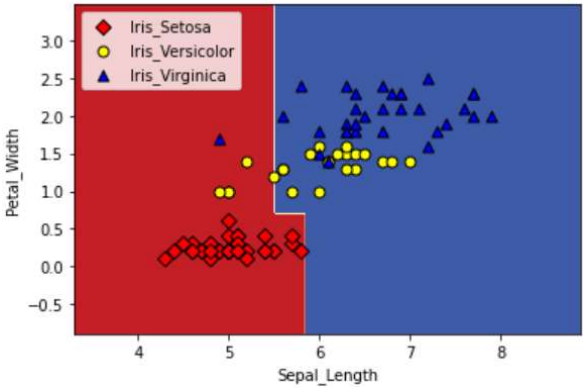






En los dos primeros casos, “Sepal_Length” fue considerada la mas imporante (la de mayor entropía) y en el tercer caso Petal_Width fue considerada la más importante.

Random-Forest írá consideranado la intersección de todos los árboles de decisión que se vayan generando:



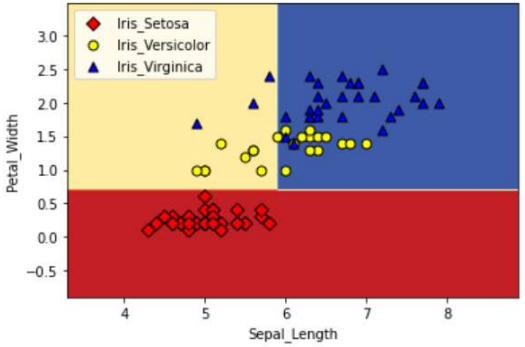
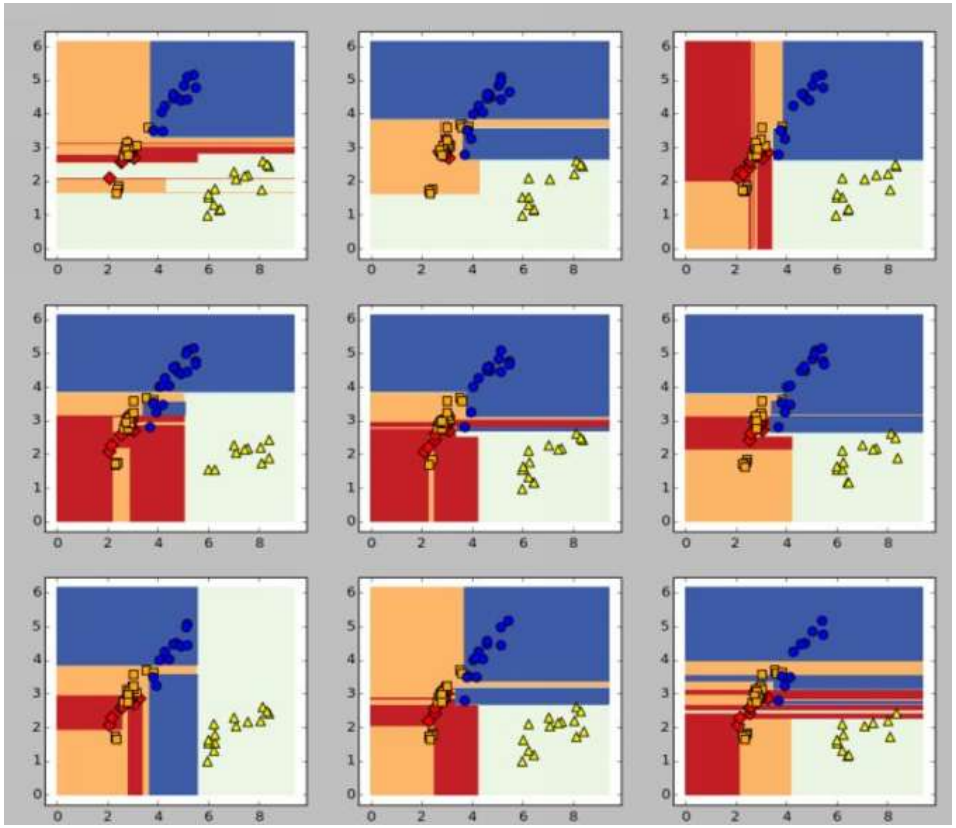
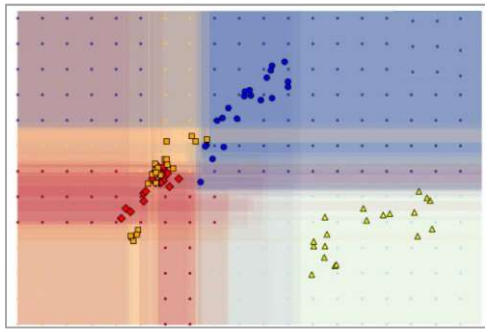


Ilustración de 9 árboles de decisión con 9 conjunto de datos diferentes en cada caso (bootstrap). Random-Forest tomará la intersección de todos ellos.

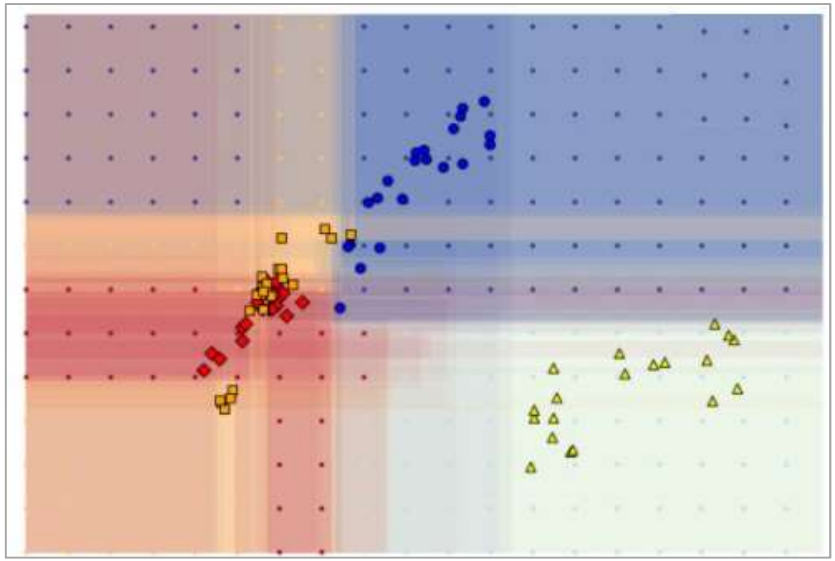
Esto hace que los datos extremos (outliers) o anómalos tengan un menor impacto en el modelo final, ya que los datos más estables son los que estarán en la mayoría de ellos y por lo tanto en la intersección del modelo final.



Random-Forest:
La intersección de todas las soluciones:



Al final, la mayoría de las hojas de cada árbol del RF o la intersección de todas ellas, no serán completamente puras (puede haber por ejemplo un 80% de la clase principal y 15% y 5% de las otras dos en el caso de Iris-dataset con tres clases), entonces cada región del plano queda asociado a tres valores de probabilidad, siendo la mayor probabilidad la que se usará de manera práctica para determinar la clase a la que pertenece un punto de prueba dado.



Out of Bag: O.O.B.

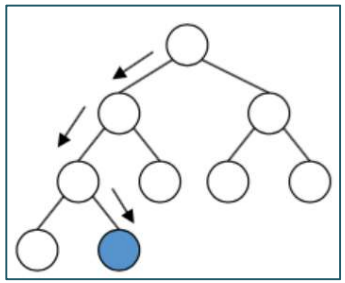
En cada árbol que genera RF, se selecciona un conjunto de datos diferentes con reemplazo (bootstrapping) para realizar el entrenamiento y por lo tanto dejará de seleccionar otros del conjunto original. Los datos no seleccionados, llamados out-of-bag, se utilizarán para evaluar ese árbol en particular.

Es decir, aún cuando no se tenga una partición de los datos originales, RF estará usando un conjunto de datos para entrenamiento y otros (O.O.B.) para validar.

De manera general, aproximadamente el 36.8% de los datos usados de entrenamiento estarían formando el conjunto OOB. Esto por lo siguiente:

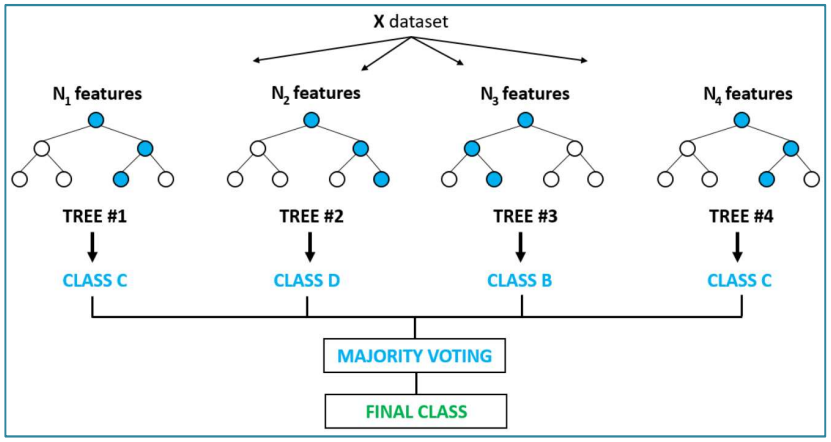
$$\underbrace{\frac{R-1}{R}}_{\text{Probabilidad de no seleccionar uno de los R renglones del conjunto de entrenamiento en una extracción. Ya que la probabilidad de seleccionarlo es 1/R.}} \Rightarrow \underbrace{\left(\frac{R-1}{R}\right)^{R}}_{\text{Usando muestreo con reemplazo, la probabilidad de no seleccionar R renglones en R repeticiones aleatorias}} \Rightarrow \underbrace{\lim_{R\to\infty}\left(\frac{R-1}{R}\right)^{R} = \frac{1}{e} \approx 0.368}_{\text{Y el límite al llevar a cabo “una gran cantidad” de repeticiones de muestreos con reemplazo.}}$$

Árbol de Decisión
(Decision Tree)



Bosque Aleatorio
(Random Forest)

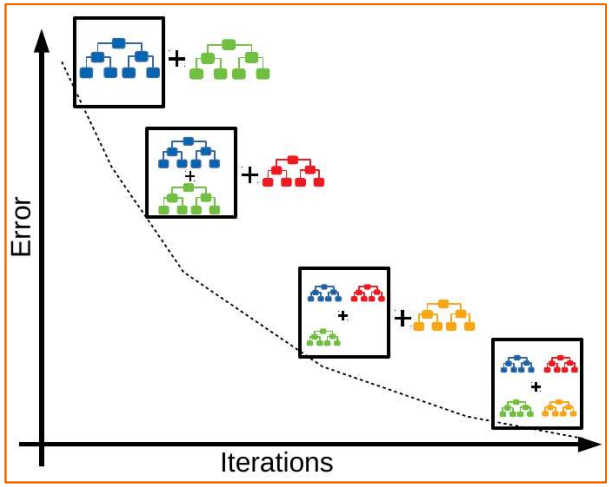
Muestreos con reemplazo



- Bagging (Bootstrap Aggregating) -

Voto por mayoría de
todos los árboles

Potenciación del Gradiente
(Gradient Boosting)



Entrenados de forma secuencial, cada nuevo árbol trata de mejorar los errores de los anteriores.

- Boosting -

Combinar algoritmos débiles para obtener como resultante uno fuerte: AdaBoost, Gradient Boosting, Stochastic Gradient Boosting, etc