Árboles

ID3
Entropía
Ganancia de información

Impureza de Gini

C4.5
Poda

Random Forest
Bootstrap aggregating
Attribute bagging (o random subspace)

ID3

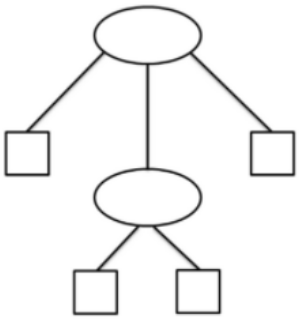
ID3: Iterative Dichotomiser 3 (tree -> árbol)

- Genera un árbol de decisión a partir de un conjunto de ejemplos.

Representación de un Árbol de decisión

→ se llama asi porque nos permite tomar decisiones, a partir de ciertas variables, valores de entrada los cuales vamos a tenerlos representados en los nodos. Y en las aristas vamos a tener las respuestas

La salida del algoritmo ID3 se representa como un grafo en forma de árbol



Sus componentes son:

- Un **nodo principal** llamado raíz en la parte superior
- **Nodos terminales** → son nodos donde termina el flujo y que ya no son raíz de ningún otro nodo. Estos nodos terminales deben contener una respuesta, o sea, la clasificación a que pertenece el objeto que ha conducido hasta él.
- Los demás nodos representan preguntas con respecto al valor de uno de los atributos.
- Las líneas representan las posibles respuestas que los atributos pueden tomar

Entropia (de la información)

La medida del desorden o la medida de la pureza. Básicamente, es la medida de la impureza o aleatoriedad en los datos

Entropía: Para calcular la entropía de n clases se utiliza la fórmula:

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

Dónde:

- S: es una lista de valores posibles.
- Pi: es la probabilidad de los valores.
- i: Cada uno de los valores.

Las n clases de un problema de clasificacion

Si usamos el ejemplo iris ⇒ clases posibles 3 (setosa, virginica, versicolor)

Importante

- Para una muestra homogénea la entropía es igual a 0 → que todos los registros son iguales
- La máxima entropía viene dada por Log2(n), n son los posibles valores de salida.

Si n = 2 (TRUE o FALSE) entonces, la máxima entropía es 1.

O sea es la máxima incertidumbre

Ejemplo: Imaginemos que queremos saber cual es la ***entropia de Shannon*** de tirar un dado.

Estados posibles

$$H(X) = 6 * \frac{1}{6} log_2(6) = 2,58$$

El 6 es por la sumatoria de los 6 valores que tienen igual prob que es 1/6.

En el log esta invertido por el menos

Ejemplo: Supongamos que el numero de estados de un mensaje es igual a 3, M1, M2, y M3. Donde la probabilidad de M1 es 50%, la de M2 25% y la de M3 es 25%.

$$H(M) = \frac{1}{2} log_2(2) + \frac{1}{4} log_2(4) + \frac{1}{4} log_2(4) = 1,5$$

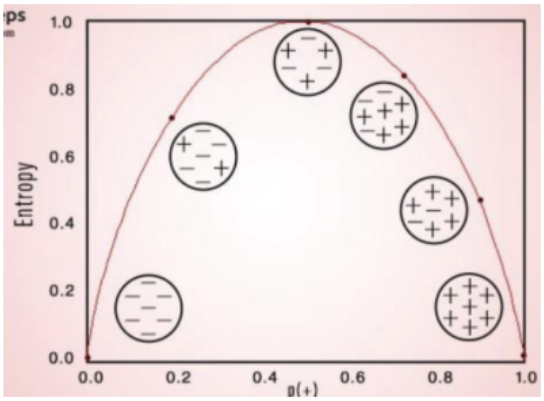
La entropia aumenta o disminuye segun que tan homogenea es la muestra

n=2 para casos positivos y negativos

Si todos son - ⇒ h=0
No hay ningun desorden de info, la muestra es completamente homogenea

Si es 50/50 ⇒ h es absoluta

Al invertir, vuelve a decrecer la entropia hasta cuando todos son positivos donde h=0



Ganancia de información

La ganancia de información se aplica para cuantificar qué característica, de un conjunto de datos dados, proporciona la máxima información sobre la clasificación.

Si tuviésemos que elegir una sola característica, para clasificar. ¿Cuál sería?

$$\textbf{Gan Inf}(S, A) = \textbf{Entropia}(S) - \sum_{v \in V(A)} \frac{|Sv|}{|S|} \textbf{Entropia}(Sv)$$

Dónde:

- S: es una lista de valores posibles para un atributo dado: A
- A: Uno de los atributos, en la lista de ejemplo.
- V(A) : Conjunto de valores que A puede tomar
- Sv/S = Probabilidad de un valor, para el atributo A.
- Entropía (Sv), entropía calculada para el valor "v" de A.

Esto es lo que nos interesa calcular, queremos saber que atributo nos da más información sobre el conjunto de datos, sobre la variable target, cual es el que mejor clasifica.

Algoritmo básico:

1. Calcular la entropía para todas las clases.
2. Calcular la entropía para cada valor posible de cada atributo.
3. Seleccionar el mejor atributo basado en la reducción de la entropía. Usando el cálculo de Ganancia de la Información
4. Iterar, para cada sub-nodo. Excluyendo el nodo raíz, que ya fue usado.

Ejemplo

Animal	ATRIBUTO			Clase
	Vuela	Piel	Nacimiento	
vaca	no	pelo	placenta	Mamifero
cerdo	no	pelo	placenta	Mamifero
paloma	sí	pluma	huevo	Ave
murcielago	sí	pelo	placenta	Mamifero
gallina	no	pluma	huevo	Ave
iguana	no	escamas	huevo	Reptil
cocodrilo	no	escamas	huevo	Reptil

Animal es un id, una forma de identificar cada registro

Vamos a calcular la probabilidad de cada clase (en el conjunto de entrenamiento)

Son 7 animales

Animal	Probabilidad	-P * Log2(P)
Mamifero	3/7 = 0.43	0.52
Ave	2/7 = 0.29	0.52
Reptil	2/7 =0.29	0.52

$$\text{Entropia}(S) = \sum_{i=1}^{n} -p_i \log_2 p_i$$ **= 1,56**

Tomamos primero Atributo Vuela = NO
Tachamos lo que no va

Animal	ATRIBUTO			Clase
	Vuela	Piel	Nacimiento	
vaca	no	pelo	placenta	Mamifero
cerdo	no	pelo	placenta	Mamifero
gallina	no	pluma	huevo	Ave
iguana	no	escamas	huevo	Reptil
cocodrilo	no	escamas	huevo	Reptil

Ahora tenemos 5 registros para los que no y 2 para los que si:

p1= 5/7 = 0.71 p2 = 2/7 = 0.29

Probabilidades del atributo Vuela NO				0,71
Mamifero	2/5	0,40	0,53	
Ave	1/5	0,20	0,46	
Reptil	2/5	0,40	0,53	
			1,52	Entropia(S,A)

Probabilidades del atributo Vuela Sí				0,29
Mamifero	1/2	0,50	0,50	
Ave	1/2	0,50	0,50	
Reptil	0	0,00	0,00	
			1,00	Entropia(S,A)

Probabilidades del atributo Piel: PLUMA				0,29
Mamifero	0/2	0,00	0,00	
Ave	2/2	1,00	0,00	
Reptil	0/2	0,00	0,00	
			0,00	Entropia(S,A)

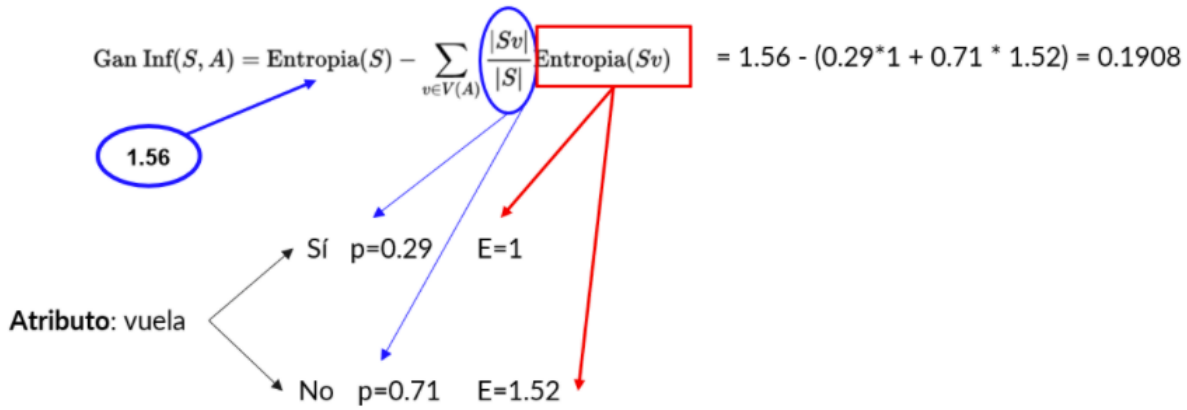
Probabilidades del atributo Piel: PELO				0,43
Mamifero	3/3	1,00	0,00	
Ave	0	0,00	0,00	
Reptil	0	0,00	0,00	
			0,00	Entropia(S,A)

Probabilidades del atributo Piel: Escamas				0,29
Mamifero	0/2	0,00	0,00	
Ave	0/2	0,00	0,00	
Reptil	2/2	1,00	0,00	
			0,00	Entropia(S,A)

Probabilidades del atributo Nacimiento: Huevo				0,57
Mamifero	0	0,00	0,00	
Ave	2/4	0,50	0,50	
Reptil	2/4	0,50	0,50	
			1,00	Entropia(S,A)

Probabilidades del atributo Nacimiento: Placenta				0,43
Mamifero	3/3	1,00	0,00	
Ave	0	0,00	0,00	
Reptil	0	0,00	0,00	
			0,00	Entropia(S,A)

Calculamos la entropia para el conjunto en gral y para cada uno de los atributos con sus diferentes valores

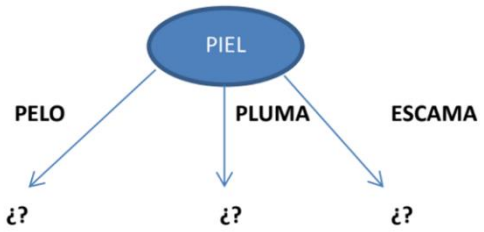


Repetimos con los otros atributos

Atributos	Ganancia de información
Vuela	0.191
Piel	1.557
Nacimiento	0.985

La piel da mayor ganancia de info ⇒ tomamos este atributo para hacer el nodo raiz
Con la piel puedo predecir mejor que tipo de animal es

Animal	ATRIBUTO			Clase
	Vuela	Piel	Nacimiento	
vaca	no	pelo	placenta	Mamifero
cerdo	no	pelo	placenta	Mamifero
~~paloma~~	~~sí~~	~~pluma~~	~~huevo~~	~~Ave~~
murcielago	sí	pelo	placenta	Mamifero
~~gallina~~	~~no~~	~~pluma~~	~~huevo~~	~~Ave~~
~~iguana~~	~~no~~	~~escamas~~	~~huevo~~	~~Reptil~~
~~cocodrilo~~	~~no~~	~~escamas~~	~~huevo~~	~~Reptil~~



Probabilidades de la clase PELO		Probabilidad	"-P*Log2(P)"
Mamifero	3/3	1	0
Ave	0/3	0	0
Reptil	0/3	0	0

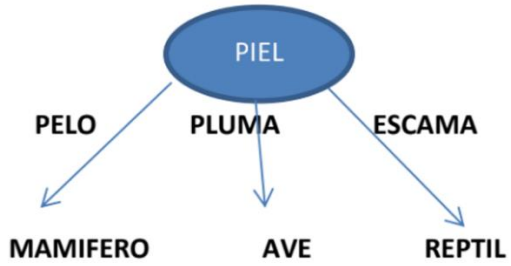
Entropía de la clase PELO: 0

Probabilidades de la clase PLUMA		Probabilidad	"-P*Log2(P)"
Mamifero	0/2	0	0
Ave	2/2	1	0
Reptil	0/2	0	0

Entropía de la clase PLUMA: 0

Probabilidades de la clase ESCAMA		Probabilidad	"-P*Log2(P)"
Mamifero	0/2	0	0
Ave	0/2	0	0
Reptil	2/2	1	0

Entropía de la clase ESCAMA: 0



La muestra es completamente homogenea en los tres casos.
Como la entropia es cero, no tengo que agregar otro nodo.

Este modelo así como esta no contempla datos que estén por fuera de lo que estaba en el conjunto de entrenamiento.

Si entra un perro funciona la predicción porque al ver la piel que es pelo -> predice que es mamifero.
Si entra un animal con otra característica, el modelo no sirve.

Un **nodo de decisión** está asociado a uno de los atributos y tiene 2 o más ramas que salen de él, cada una de ellas representando los posibles valores que puede tomar el atributo asociado.

Un **nodo-respuesta** está asociado a la clasificación que se quiere proporcionar, y nos devuelve la decisión del árbol con respecto al ejemplo de entrada.

Impureza de Gini

La impureza de Gini es una medida de cuán a menudo un elemento elegido aleatoriamente del conjunto sería etiquetado incorrectamente si fue etiquetado de manera aleatoria de acuerdo a la distribución de las etiquetas en el subconjunto

Algunas implementaciones de árboles de decisión utilizan la impureza de Gini en lugar de la ganancia de información, ya que **es más fácil de calcular (computacionalmente menos costosa)**
Ejemplo: Scikit-learn

$$Gini = 1 - \sum_{i=1}^{C} (p_i)^2$$

Ejemplo:

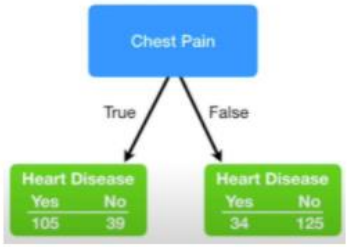
Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...

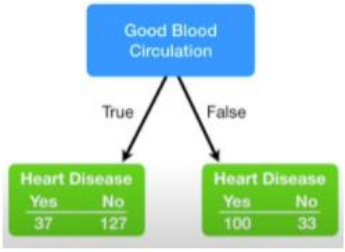
- Chest Pain (dolor de pecho)
- Good Blood Circulation (buena circulación sanguínea)
- Blocked Arteries (arterias bloqueadas)

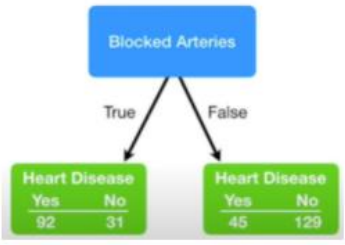
La clase, lo que queremos obtener, lo que queremos clasificar es saber si hay o no una enfermedad del corazon

¿Cúal debería ser el mejor atributo para el nodo raíz?

Voy contando la cantidad de casos en cada categoria







¿Qué tan bien clasifica cada uno de estos árboles?

En Chest Pain → True, tiene la mayoría de "Yes", y "False" la mayoría de "No". Pero no clasifica correctamente al 100% de los casos. En los otros árboles tampoco clasifican correctamente al 100% de los casos.

- El total no siempre coincide, ya que para valores faltantes (nulos), se omite el registro
  Si sumamos todos los casos puede que no sea el total de los valores

Como ninguno de los nodos hoja es 100%: "Yes Heart Disease" o bien 100% "No Heart Disease". Ninguno es puro, es decir que son **nodos impuros**

Que tan impuros son?

Para saber cual clasifica mejor, tenemos que medir la impureza de cada nodo y de cada árbol.

Hay varias formas de medir esto, una muy popular es llamada: **Gini**

Para esta hoja, la impureza es:

Gini = 1 - (probabilidad de yes)^2 - (probabilidad de no)^2

$$Gini = 1 - (\frac{105}{105+39})^2 - (\frac{39}{105+39})^2$$
$$= 0.395$$

En la otra hoja:

$$Gini = 1 - (\frac{34}{34+125})^2 - (\frac{125}{34+125})^2$$
$$= 0.336$$

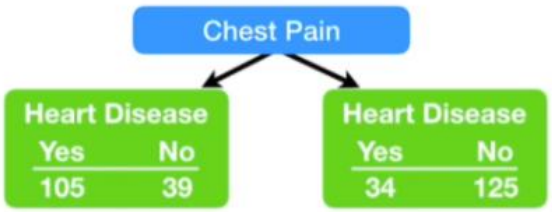
Ahora que tenemos la impureza de Gini para cada nodo hoja, podemos calcular la impureza de Gini total, para el nodo raíz: "Chest Pain".
Se trata de el **promedio ponderado**.

105+39=144 y 34+125=159

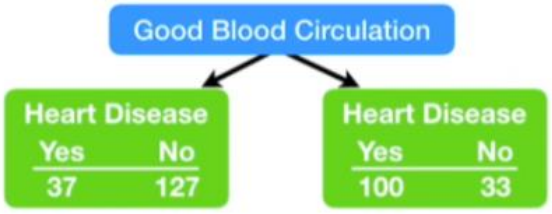
$$Gini\ ChestPain = (\frac{144}{144+159})\ 0.395 - (\frac{159}{144+159})\ 0.336 = 0.364$$

Hago eso en todas:

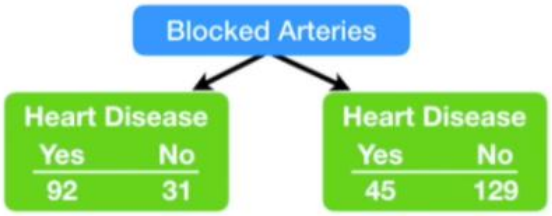
Gini impurity for Chest Pain = 0.364



Gini impurity for Good Blood Circulation = 0.360



Gini impurity for Blocked Arteries = 0.381



Cómo "Good blood circulation" tiene la menor impureza, utilizamos este atributo para el nodo raíz.

Luego procedemos de igual forma que en el caso de la ganancias de información pero calculando la impureza de Gini

C4.5

Mejoras a ID3:

- Soporta campos numéricos, rangos continuos
- Soporta datos faltantes
- Se agrega la Poda → para evitar el sobreajuste. Los arboles tienden al overfitting

Datos faltantes:

Manejo de los datos de formación con valores de atributos faltantes

C4.5 permite valores de los atributos para ser marcado como "?" para faltantes.
Los valores faltantes de los atributos simplemente no se usan en los cálculos de la ganancia y la entropía.

Campos numéricos, o rangos continuous:

Si un atributo A tiene un rango continuo de valores → El algoritmo puede, dinámicamente crear un campo Booleano tal que si A < C ⇒ **Ac = TRUE**, sino **Ac = FALSE**.

El rango originalmente numerico queda partido en 2 categorias

¿Cómo encontrar ese umbral C?
Vamos a cortar el rango de forma que C nos quede con la mayor ganancia de información.

Pasos:

- Ordenamos A de menor a mayor, por ejemplo.
- Identificamos los valores adyacentes (de la clase que es nuestra salida)
- Detectamos cuando hay un cambio de valor de salida, entonces en esos límites seguramente están nuestros Ci candidatos.
- Creamos varios Ci, que dividen en dos el rango. Para cada uno de estos rangos calculamos la ganancia de información. Nos quedamos con el que nos da el mejor resultado.
  (Se puede también quedarse con los N mejor.)

Poda

El método de poda del árbol consiste en:

- Generar el árbol tal y como hemos visto → Arbol completo con todos los nodos
- A continuación, analizar recursivamente, y desde las hojas, qué preguntas (nodos interiores) se pueden eliminar sin que se incremente el error de clasificación con el conjunto de test.

(Si hay ruido el árbol tendrá un error, es decir cantidad de casos mal clasificados)
Error= Casos bien clasificados / Casos Totales

1. Se elimina un nodo interior cuyos sucesores son todos nodos hoja.
2. Se vuelve a calcular el error que se comete con este nuevo árbol sobre el conjunto de test.
3. Si este error es menor que el error anterior, entonces se elimina el nodo y todos sus sucesores(hojas)
4. Se repite.

Random Forest

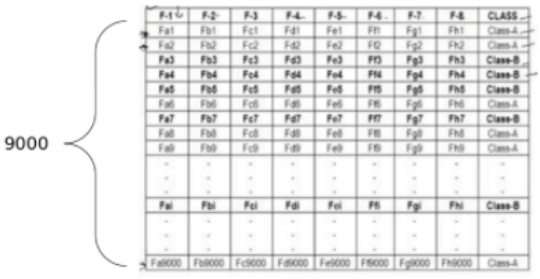
"Muchos estimadores mediocres, promediados pueden ser muy buenos"

Bootstrap aggregating:

Es una técnica, o meta-algoritmo que dice lo siguiente:
Dado un conjunto de entrenamiento D, de tamaño n, la técnica de bagging generará m nuevos conj de entrenamiento D1,... Di,..., Dm cada uno de tamaño n' tomando muestras aleatorias de D.
Y en general n'<n. Siendo n' aproximadamente un ⅔ de n.

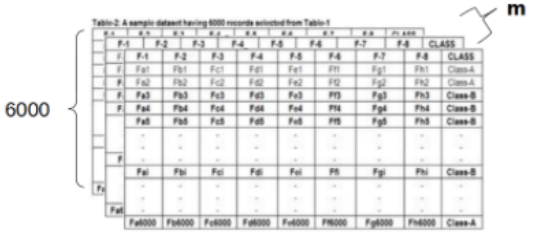
Suponiendo que tenemos un conjunto de datos, como el que sigue:

- 8 Atributos
- 1 Clase que queremos saber
- 9000 registros



Hacemos m sub-tablas tomando solo algunas filas, de forma aleatoria, ¿cuantas tomamos?
=> 2/3 es decir 6000

Nos quedamos con m datasets mas chicos



Attribute bagging (o random subspace):

Luego para cada una de las m tablas, escogemos sólo algunos atributos (COLUMNAS) de forma aleatoria también.

¿Con cuantas nos quedamos? => Con RAÍZ CUADRADA del número de atributos.

Cómo acá hay 8 atributos, tomamos Round(√8) = 3

(Esto es recomendable sobre todo cuando hay muchas columnas)

Table-3: A sample dataset having 6000 records, 3-features, selected from Table-2

F-1	F-5	F-7	CLASS
Fa1	Fe1	Fg1	Class-A
Fa2	Fe2	Fg2	Class-A
Fa3	Fe3	Fg3	Class-B
Fa4	Fe4	Fg4	Class-B
Fa5	Fe5	Fg5	Class-B
.	.	.	.
.	.	.	.
.	.	.	.
Fai	Fei	Fgi	Class-B
.	.	.	.
.	.	.	.
.	.	.	.
Fa6000	Fe6000	Fg6000	Class-A

Random Forest:

Ahora tenemos m tablas reducidas en atributos y para cada una de ellas entrenamos un árbol

- Para cada árbol calculamos su matriz de confusión.

n=165	**Predicted: NO**	**Predicted: YES**	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
	55	110	

Entonces calculamos la tasa de error de cada árbol, esto es:
FALSOS POSITIVOS + FALSOS NEGATIVOS SOBRE EL TOTAL DE EJEMPLOS CLASIFICADOS
(Sobre un conjunto de entrenamiento o sobre un porcentaje del conjunto utilizado)

Nos quedamos con el mejor árbol de los m, y repetimos el proceso K veces.
Al final vamos a tener K árboles.

Tasa de error = (FP + FN) / TOTAL →es el accuracy

¿Cómo clasificar una vez finalizado el proceso?

Luego para clasificar un nuevo ejemplo hacemos lo siguiente:

Ejecutamos el caso que queremos probar por cada uno de los K arboles
Si la mayoría de los casos devolvió que la categoría final es CATEGORIA-A, entonces nos quedamos con ese valor de salida.