“... los animales se dividen en
(a) pertenecientes al Emperador,
(b) embalsamados,
(c) amaestrados,
(d) lechones,
(e) sirenas,
(f) fabulosos,
(g) perros sueltos,
(h) incluidos en esta clasificación,
(i) que se agitan como locos,
(j) innumerables,
(k) dibujados con un pincel finísimo de pelo de camello,
(l) etcétera,
(m) que acaban de romper el jarrón,
(n) que de lejos parecen moscas.”

El Idioma Analítico de John Wilkins
Jorge Luis Borges

DEPARTAMENTO
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Laboratorio de datos

Clasificación

Verano 2025

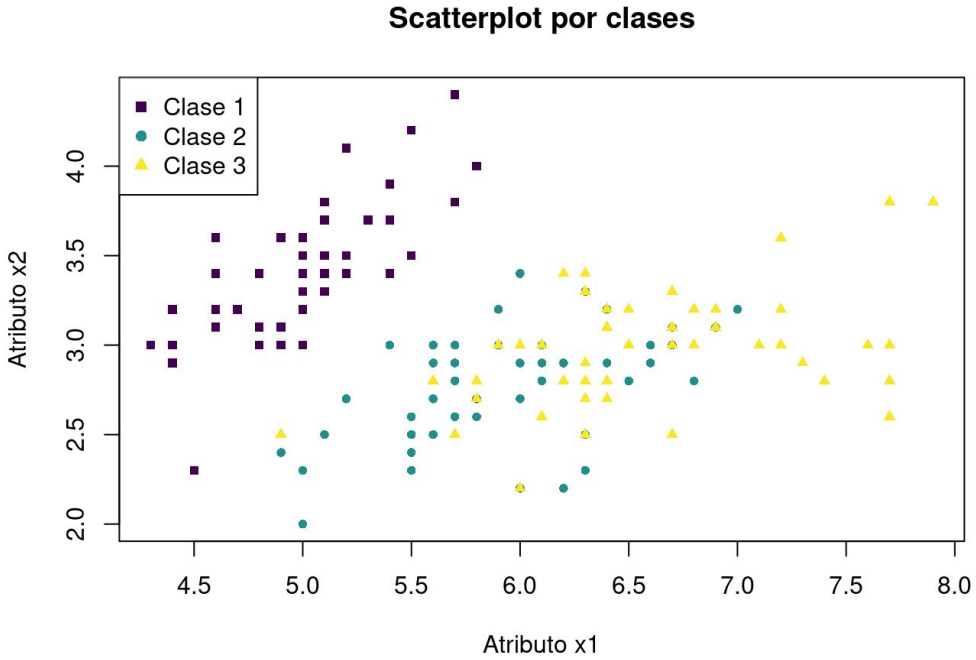
Clase de Paula Perez Bianchi, Viviana Cotik
y Manuela Cerdeiro

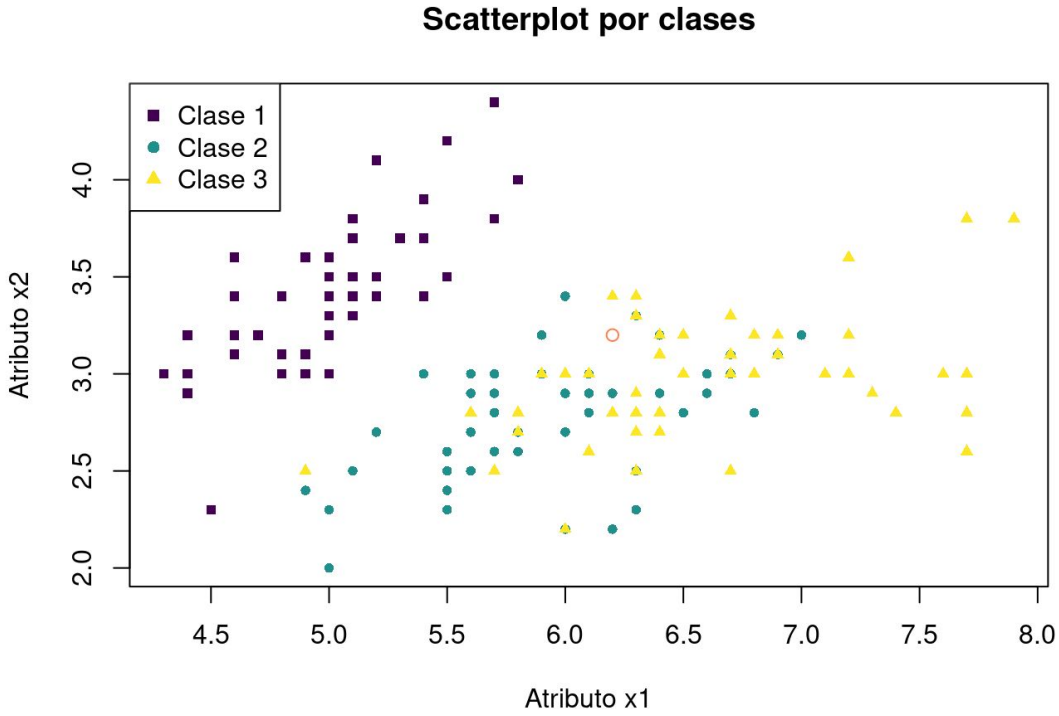
Ejemplo

Tenemos un conjunto de datos con variables *x1, x2, y*.

- Variables explicativas continuas *x1, x2*
- Variable a explicar *y* categórica

Graficamos *x1, x2* en los ejes.
La variable a explicar toma 3 valores: Clase 1, Clase 2, Clase 3 y se representa con símbolos/colores.





¿Qué clase le asignamos a la nueva observación?

Clasificación

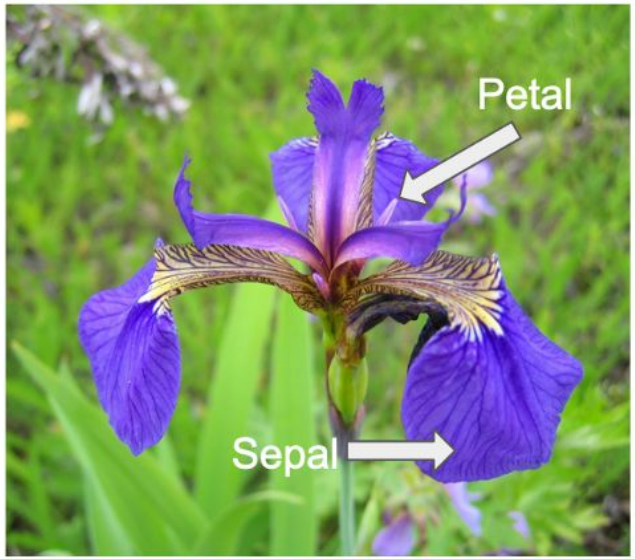
- A partir de los atributos (variables explicativas) queremos poder determinar la etiqueta - la variable categórica Y.
- Aprendizaje supervisado: contamos con un conjunto de entrenamiento en el cual conocemos las etiquetas - valores de la variable Y.
- Evaluación del modelo: medida relacionada con la cantidad de elementos bien o mal clasificados.

Métodos posibles (hay más)

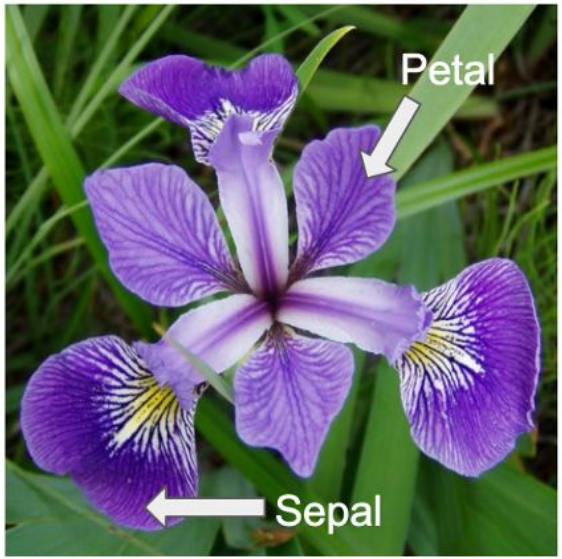
- Umbral ($x_1 > c$)
- Regresión logística
- Árboles de decisión
- Support Vector Machines (SVM)
- K-Nearest Neighbors (KNN)

Dataset de flores - Iris

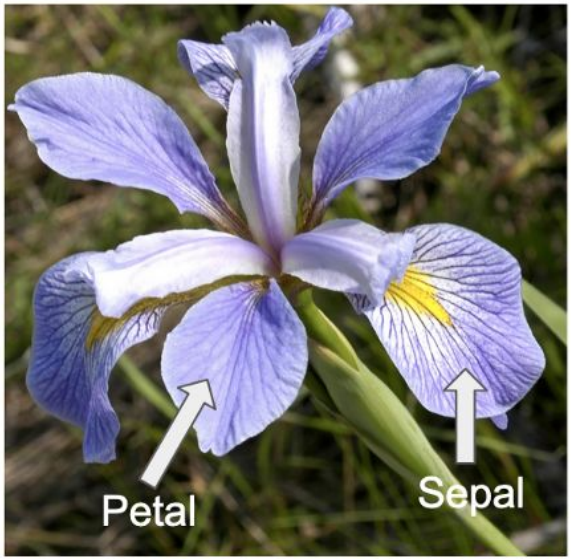
Iris setosa



Iris versicolor



Iris virginica



50 muestras de cada una de tres especies de flores *Iris: setosa, versicolor y virginica*. De cada flor se midieron 4 atributos: largo y ancho del sépalo y del pétalo.

Fisher - 1936

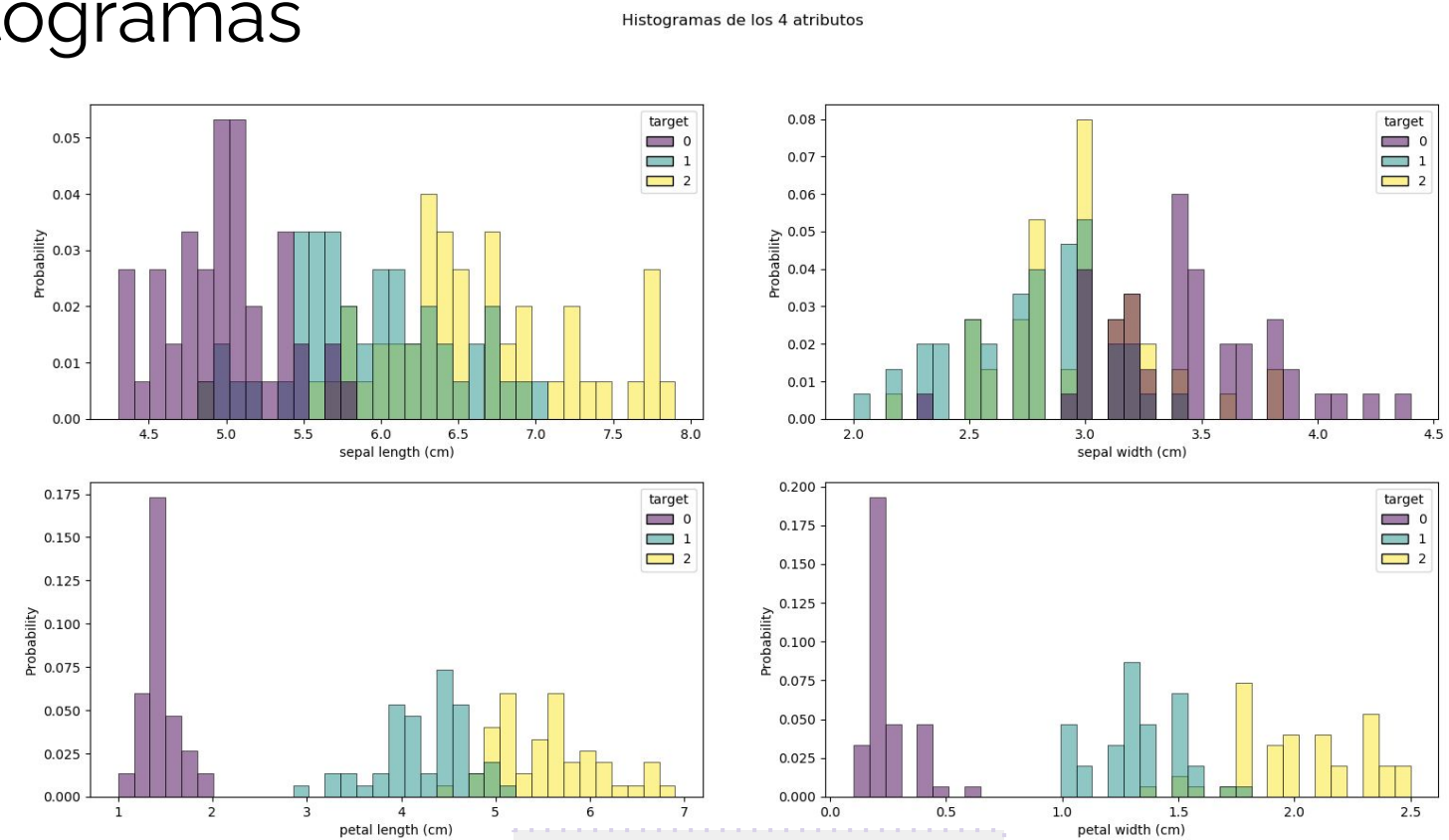
Clasificación de Flores

¿Ante las medidas de una nueva flor, cómo podríamos determinar su especie?

¿Cuál resulta el atributo más importante a la hora de distinguir entre especies?

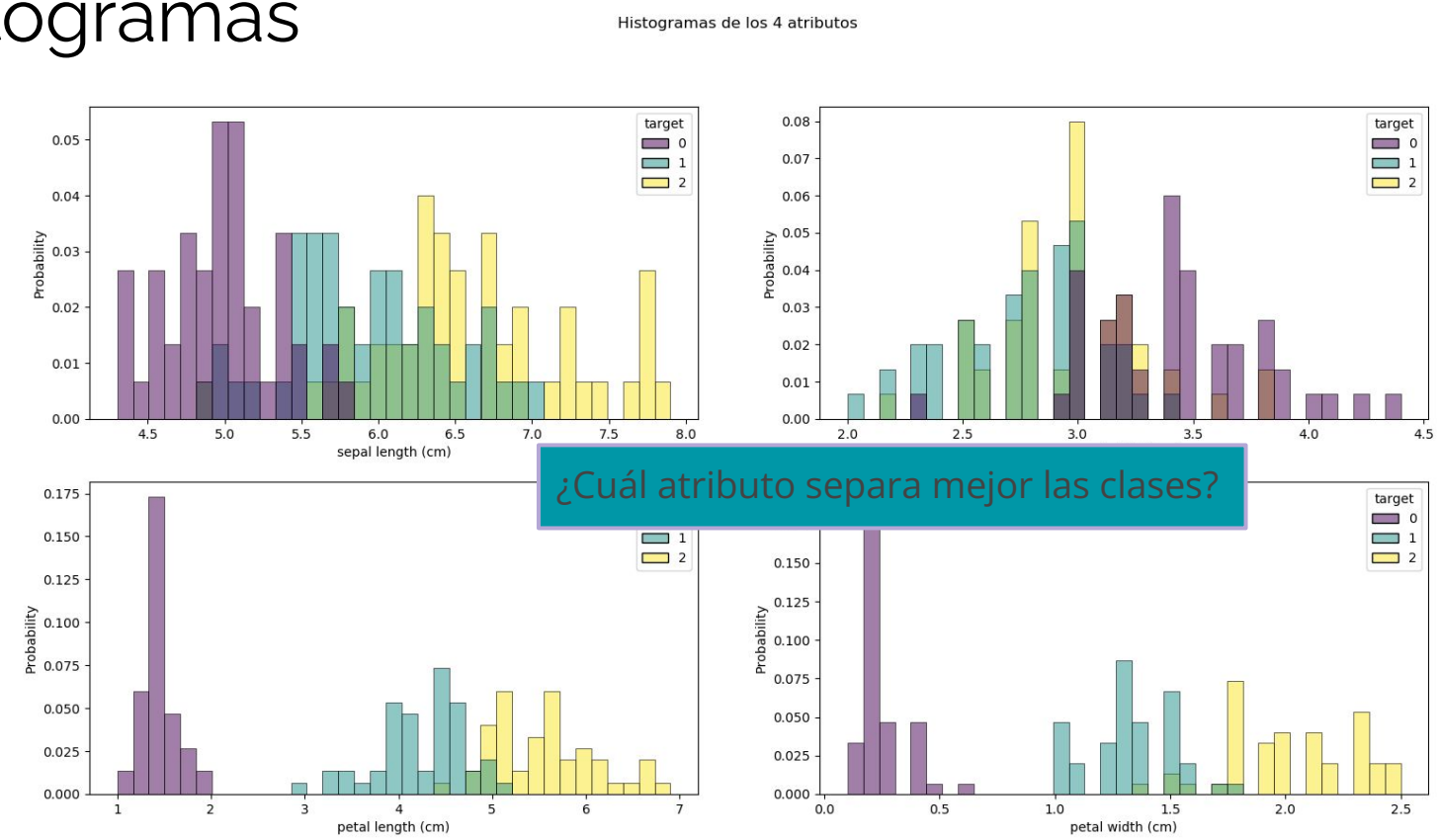
¿Qué herramienta puede ayudarnos a responder esto?

Histogramas



{0: 'setosa', 1: 'versicolor', 2: 'virginica'}

Histogramas



Clasificamos por largo del pétalo

```
def clasificador_iris
```

Clasificamos por largo del pétalo

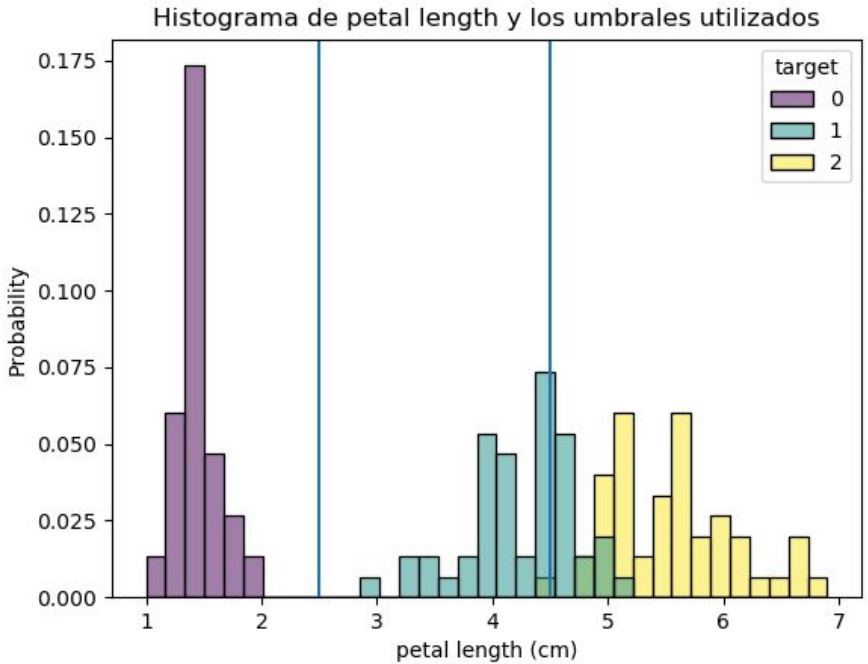
Ahora veamos cómo se comporta este clasificador.

¿Podemos mejorar el clasificador?
¿Cómo comparamos entre dos clasificadores?

Clasificamos por largo del pétalo

Ahora veamos cómo se comporta este clasificador.

Éstas son las líneas de corte.



Medidas para evaluar clasificadores

Matriz de confusión

Para cada clase i, nos fijamos cuántas observaciones de la clase fueron clasificadas en cada clase j.

Esto nos da una matriz cuadrada, con una fila y columna por cada clase.

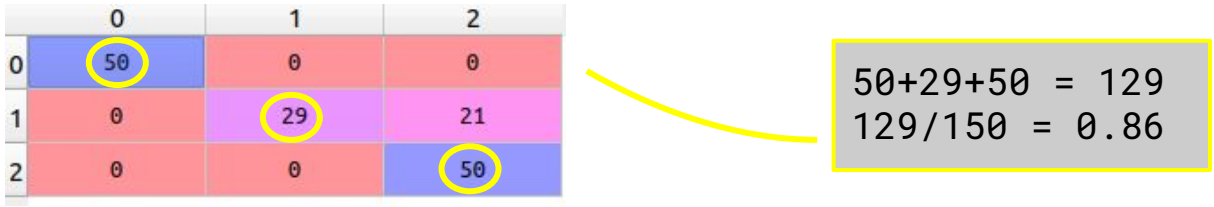
	0	1	2
0	50	0	0
1	0	29	21
2	0	0	50

{0: 'setosa', 1: 'versicolor', 2: 'virginica'}

Medidas para evaluar clasificadores

Exactitud

La exactitud o *accuracy* que es una medida numérica que cuenta la proporción de observaciones *bien* clasificadas.

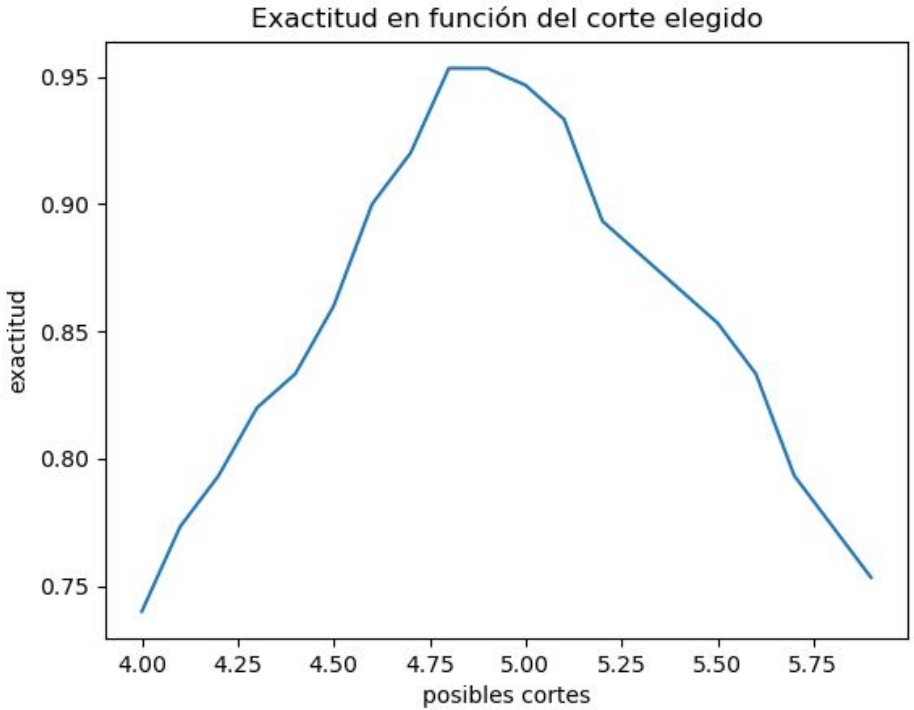


¿Buscamos el mejor clasificador?

Podemos

+ recorrer muchos posibles umbrales dentro de un rango
+ para cada umbral correr el clasificador y evaluarlo
+ comparar los clasificadores para seleccionar el mejor

Comparación de clasificadores



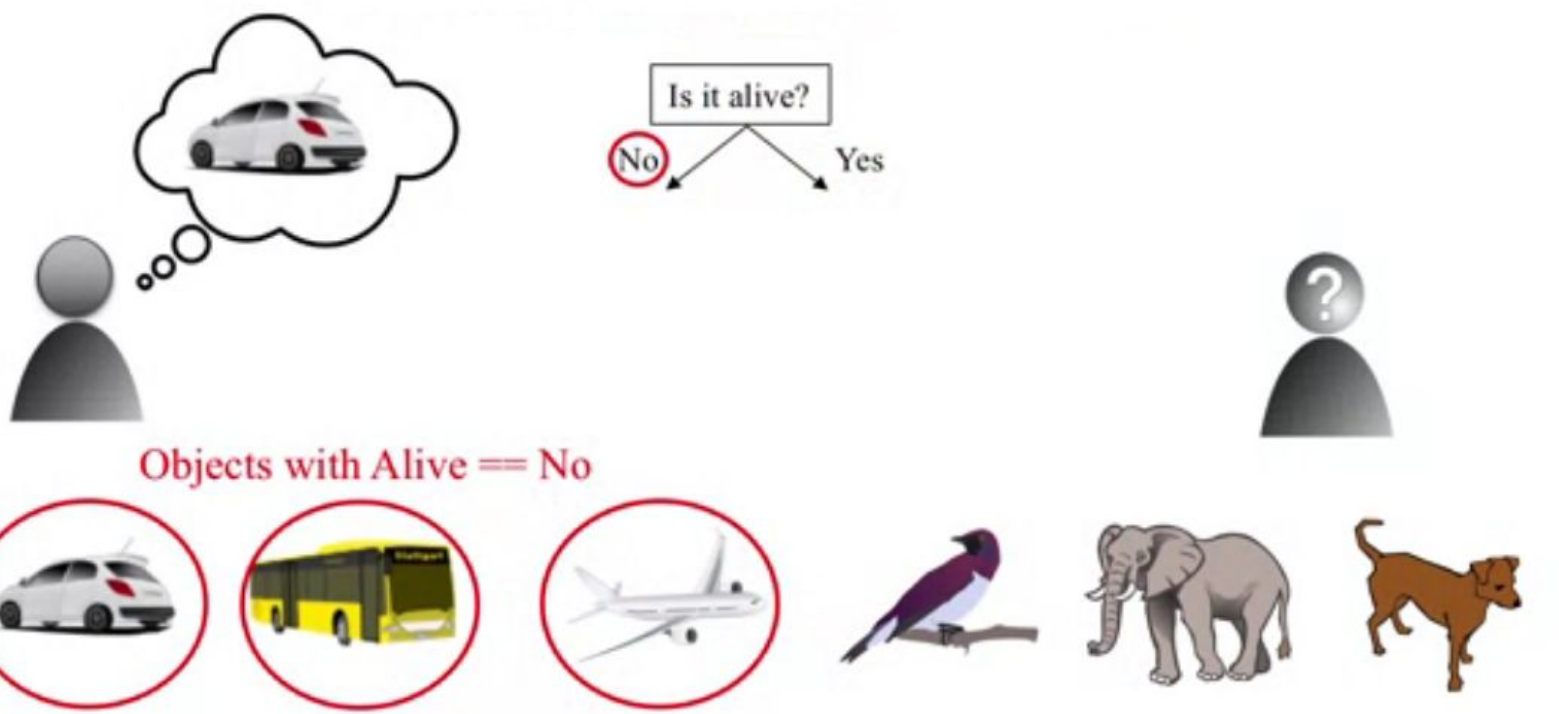

```
corte_selec =
posibles_cortes[np.argmax(exactitudes)]
```

Árboles de decisión

Árboles de decisión

- método para **inferencia inductiva**
- aprenden **reglas if-then** sobre los valores de los atributos. Predicen valor objetivo en función de las reglas.

Árboles de decisión - Ejemplo



Applied Machine Learning with Python. Universidad de Michigan

Árboles de decisión - Ejemplo

nodo: representa pregunta

aristas: representan posibles respuestas

hojas: nodos que representan decisiones

caminos desde la raíz

Árboles de decisión

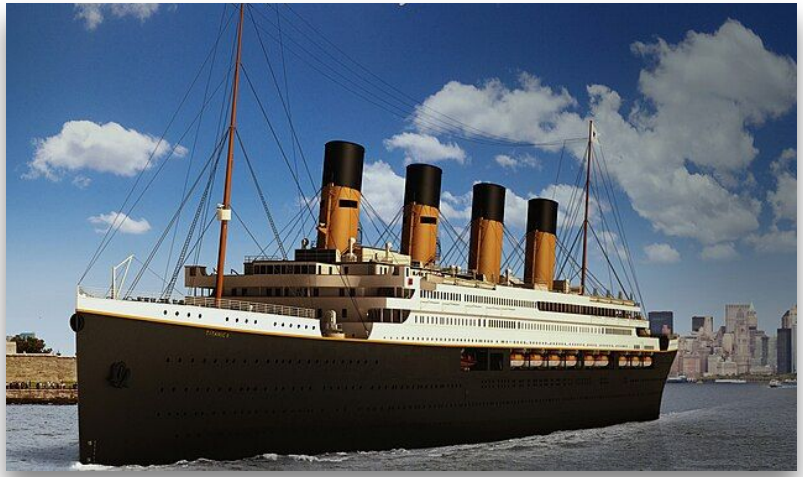
- Modelo de aprendizaje supervisado utilizado principalmente para **clasificación**.
- Se basa en el armado de una **jerarquía de reglas.** Estas las podemos expresar usando una fórmula lógica de ANDs y ORs.
- Es decir que el árbol representa una **disyunción de conjunciones sobre valores de atributos.**

Podemos pensar el armado del árbol como jugar al ¿Quién es quién?.

Los árboles son un ***modelo altamente interpretable***. Es decir que dada una predicción particular podemos entender por qué el modelo la generó. Sólo hay que mirar la rama de la hoja correspondiente a la predicción.

Dataset Titanic

El Titanic se hundió en Abril de 1912.
Tenía la fama se ser “Inhundible” pero chocó con un iceberg y no le alcanzó con la fama.
Como no había suficientes salvavidas 1502 de 2224 pasajeros y personal de a bordo murieron.



Dataset en:
https://www.kaggle.com/competitions/titanic

	PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare
1	2	1	female	38.0	1	0	71.2833
3	4	1	female	35.0	1	0	53.1000
6	7	1	male	54.0	0	0	51.8625
10	11	3	female	4.0	1	1	16.7000
11	12	1	female	58.0	0	0	26.5500

Objetivo: ¿Podemos predecir quienes sobrevivieron al titanic?

Actividad Titanic

Consigna 1: ¿Qué características tenían los pasajeros que sobrevivieron? ¡Exploren los datos y escríbanlas en un papel (o grafíquenlas)!

Algunas preguntas:

- ¿Cuántos pasajeros de primera clase había? ¿Y de segunda y tercera?
- ¿Cuál era la proporción de niños?
- ¿Quiénes les parece que tenían prioridad en los botes de rescate?

Consigna 2: Tenemos los datos de una pasajera,
¿pueden decir si sobrevivió o no?

	Pclass	Sex	Age	SibSp	Parch	Fare	Survived
184	1	female	27.0	1	1	247.5208	???

¿Adivinaron o usaron reglas? ¿Cuáles? ¿Se pueden generalizar?

Consigna 3: Competencia

PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare

Ahora les voy a dar los datos de **10 pasajeros**,
¿Pueden responder cuáles sobrevivieron?

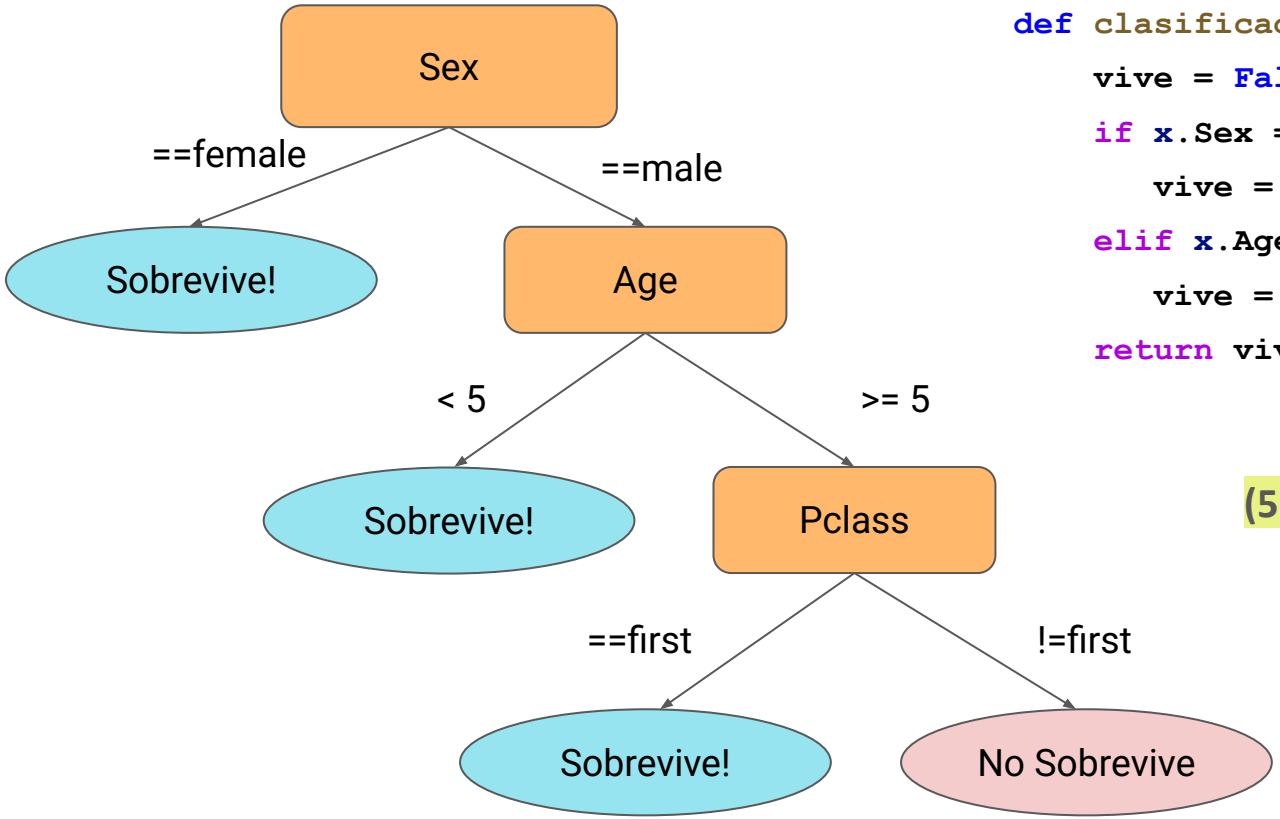
Armen un conjunto de reglas generales y generen etiquetas para estos 10 pasajeros.

```
def clasificador_titanic(x):
    vive = False
    if REGLA:
        vive = True
    return vive
```

Tienen 10 min!
https://www.online-stopwatch.com/timer/15minute/

Subir sus predicciones y el código que usaron para sus reglas a un google form.
https://forms.gle/er2MFDw3LA8GERGa9

🚣 → Sobrevivió (True)
💀 → No sobrevivió (False)



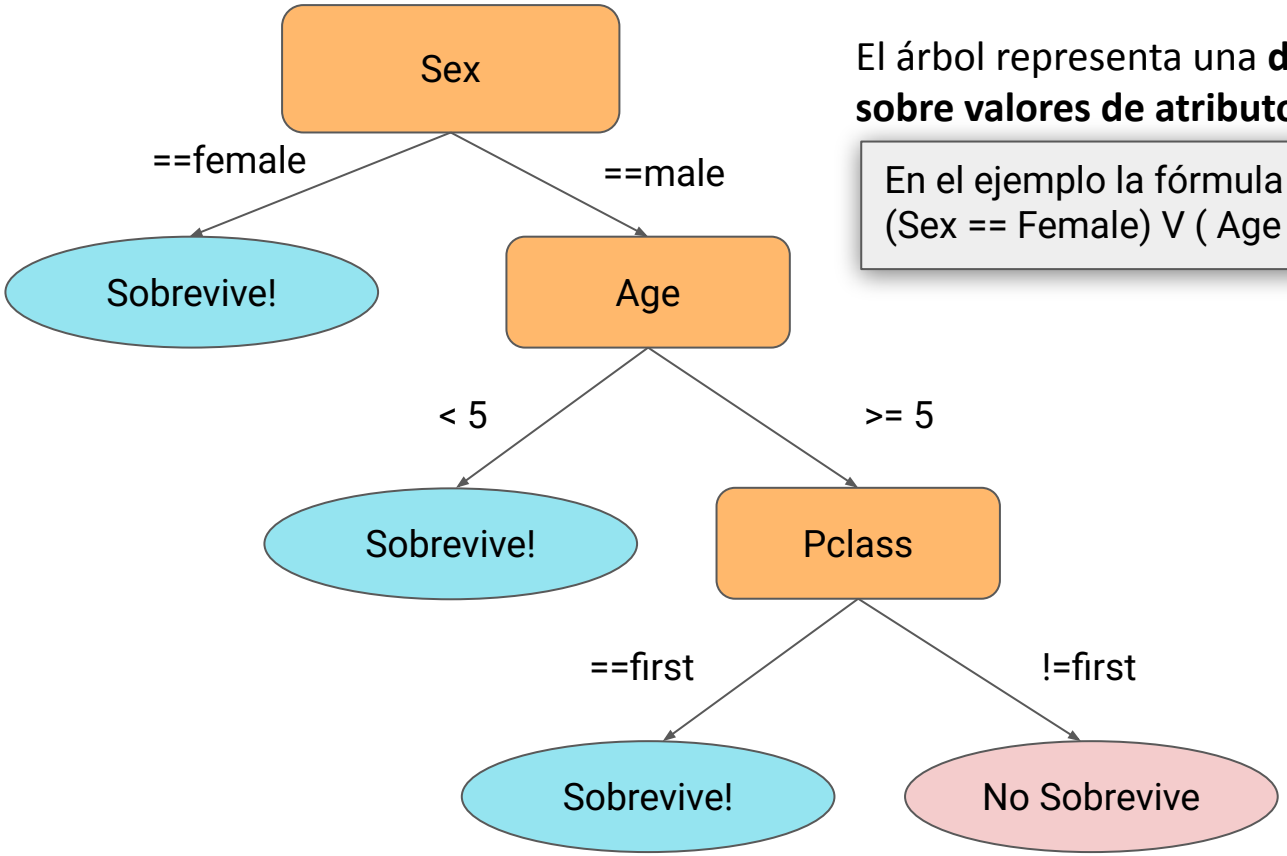

```
def clasificador_titanic(x):
    vive = False
    if x.Sex == "female":
        vive = True
    elif x.Age < 5 or x.Pclass == 1 :
        vive = True
    return vive
```

SCORE : 50%
(5 instancias bien clasificadas)

*Cada nodo interno evalúa un atributo discreto **a***
*Cada rama corresponde a **un valor/umbral** para **a***
*Cada hoja predice un valor de **Y***

Esto que hicimos a ojo se puede formalizar como un método de Aprendizaje Supervisado

Árboles de decisión



El árbol representa una **disyunción de conjunciones sobre valores de atributos.**

En el ejemplo la fórmula quedaría:
(Sex == Female) V (Age < 5) V (Age >= 5 Λ Pclass == 1)

Inducción Top-Down para árboles de decisión

(versión simplificada de ID-3 y C4.5 (Quinlan))

Input: **S** un conjunto de instancias con atributos **A**.

1. Elegimos **a ϵ A**, el atributo que produce el **mejor corte** de **S** para el nodo actual.
2. Para cada valor $\mathbf{v_i}$ posible de **a**, crear un nuevo hijo del nodo actual.
3. Clasificar (repartir) las instancias en los nuevos nodos, según el valor del atributo **a**.

   $$\mathbf{S_i \leftarrow \{ x \mid x \in S \wedge x[a] = v_i\}}$$

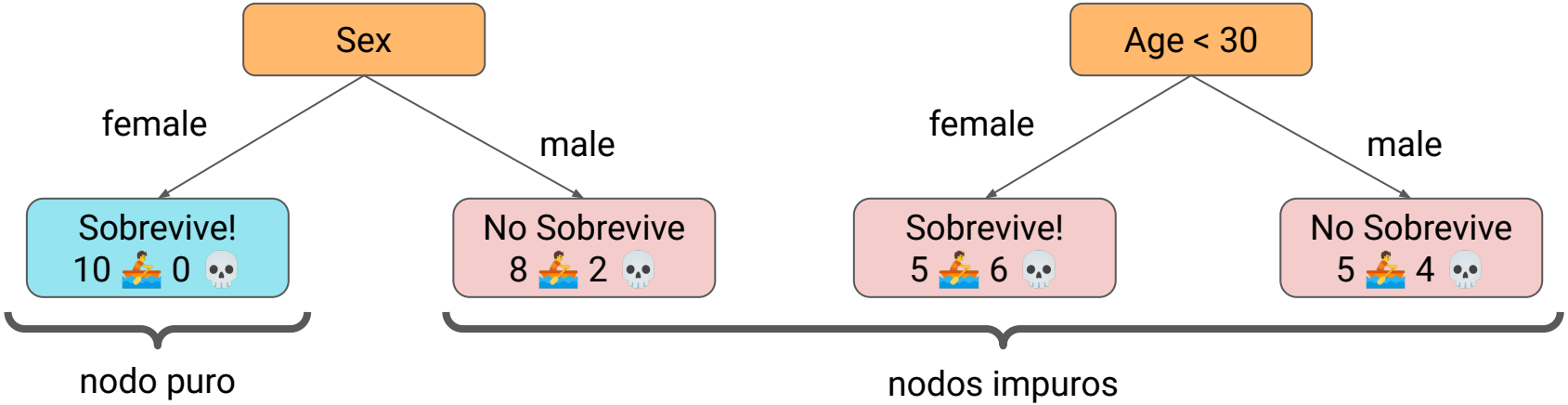
4. Repetir para cada hijo del nodo en el que haya instancias de más de una clase (salvo que se cumpla algún criterio de corte en cuyo caso terminamos).

¿Cómo definimos el mejor corte?
→ Necesitamos métricas!!

Para pensar: ¿Cómo hacemos el paso 2) para atributos continuos? ¿Cómo se definen las regiones del paso 3) ¿en ese caso?

Árboles de decisión - Medidas de impureza

🚣 → Sobrevivió
💀 → No sobrevivió



El primer árbol tiene un mejor corte que el segundo. *¿Qué lo hace mejor?*
¿Cómo medirían esto?

$$\Delta M(S, c) = M(S) - (Prop_{c,<=} * M(S_{c,<=}) + Prop_{c,>} * M(S_{c,>}))$$

*"cuánto gano si divido a **S** en regiones $S_{c,<=}$ y $S_{c,>}$ según la medida **M**" (caso binario)*

Competencia - Evaluamos los clasificadores.

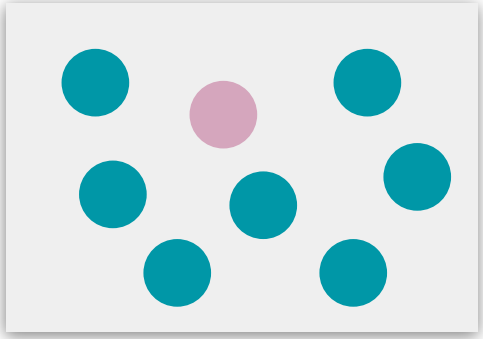
Vemos respuestas del form.
Evaluamos también con un conjunto de test.
Accuracy - cantidad de errores de cada tipo?

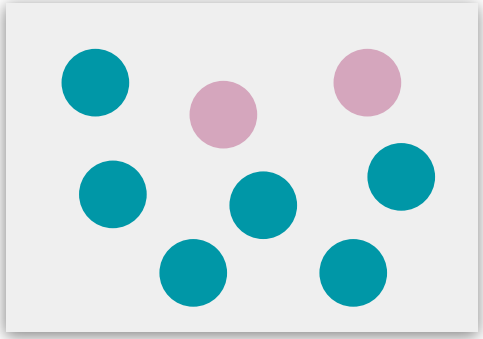
Árboles de decisión - Medidas de impureza

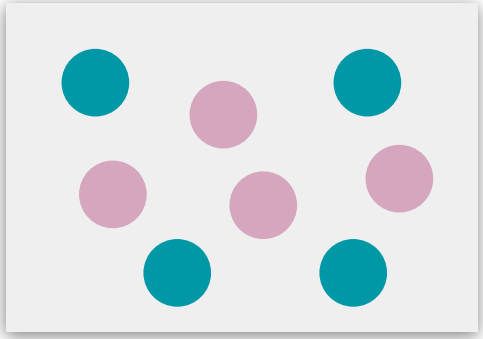
Entropía

Mide la incertidumbre inherente a los posibles resultados de una variable aleatoria.

Voy a sacar una bolita. ¿Cuánta incertidumbre hay?

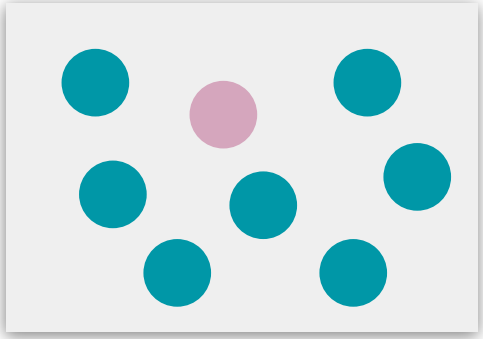




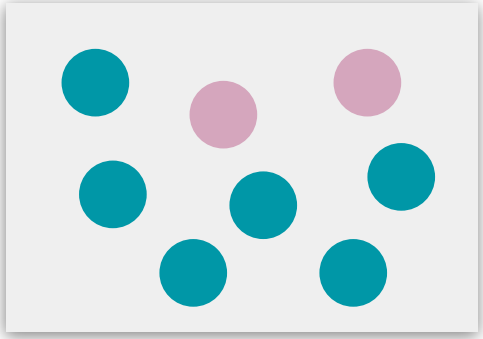


$$H = - \sum p_i \log_2(p_i)$$

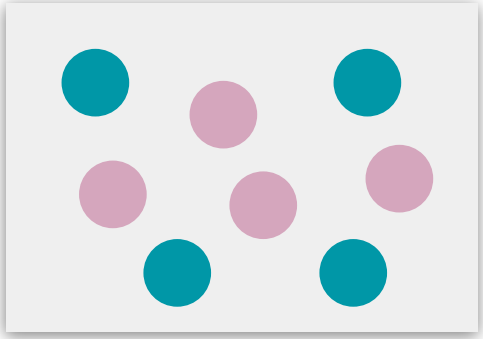
$p_A = 7/8, P_R = 1/8$



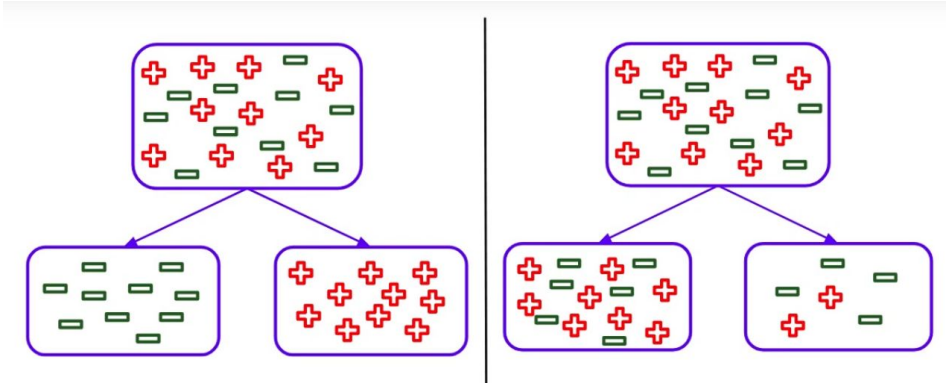
$p_A = 6/8, P_R = 2/8$

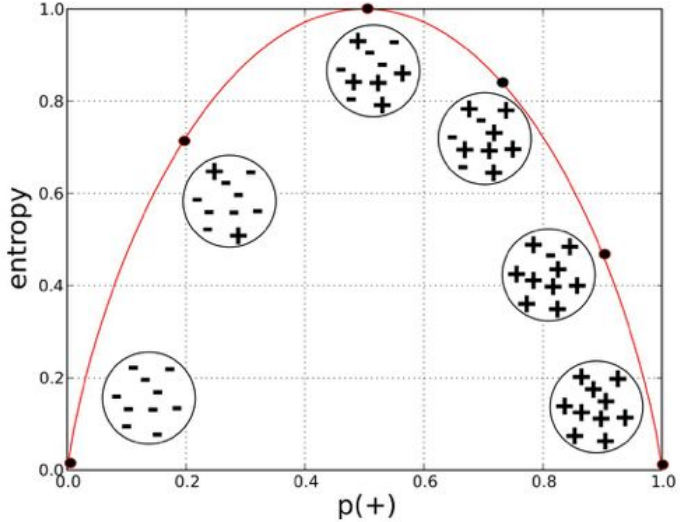


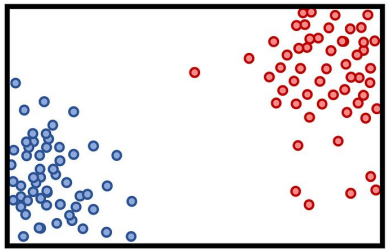
$p_A = 4/8, P_R = 4/8$



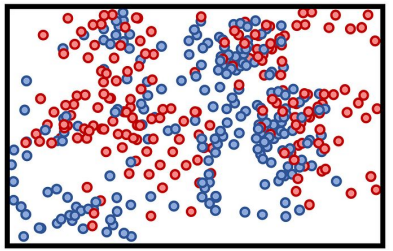
Árboles de decisión - Medidas de impureza







Low Entropy



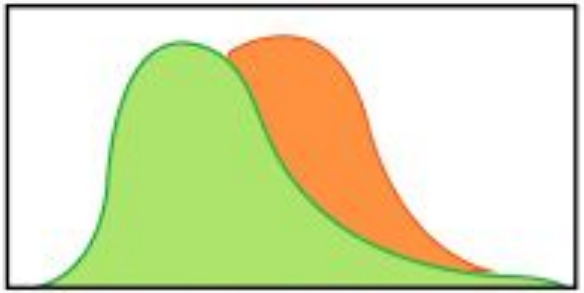
High Entropy

Árboles de decisión - Medidas de impureza

Info Gain (Ganancia de Información)

Mide Cuánta entropía removemos al hacer un corte.

$$InfoGain(S, < a, c >) = H(S) - (Prop_{\leq} \cdot H(S_{\leq}) + Prop_{>} \cdot H(S_{>}))$$



Low information gain High entropy

High information gain Low entropy

$$H(S) = -\sum_{k \in clases(S)} p_s(k) \log_2 p_s(k)$$

Árboles de decisión - Medidas de impureza

Gini Gain

La Impureza de Gini mide la probabilidad de que una instancia particular sea clasificada erróneamente si esta fuese etiquetada aleatoriamente de acuerdo con la distribución de clases dentro de la región.

$$Gini(S) = 1 - \sum_{k \in clases(K)} (p_s(k))^2$$

$$Gini\_Gain(S, < a, c >) = Gini(S) - (Prop_{\leq} \cdot Gini(S_{\leq}) + Prop_{>} \cdot Gini(S_{>}))$$

Sesgo inductivo

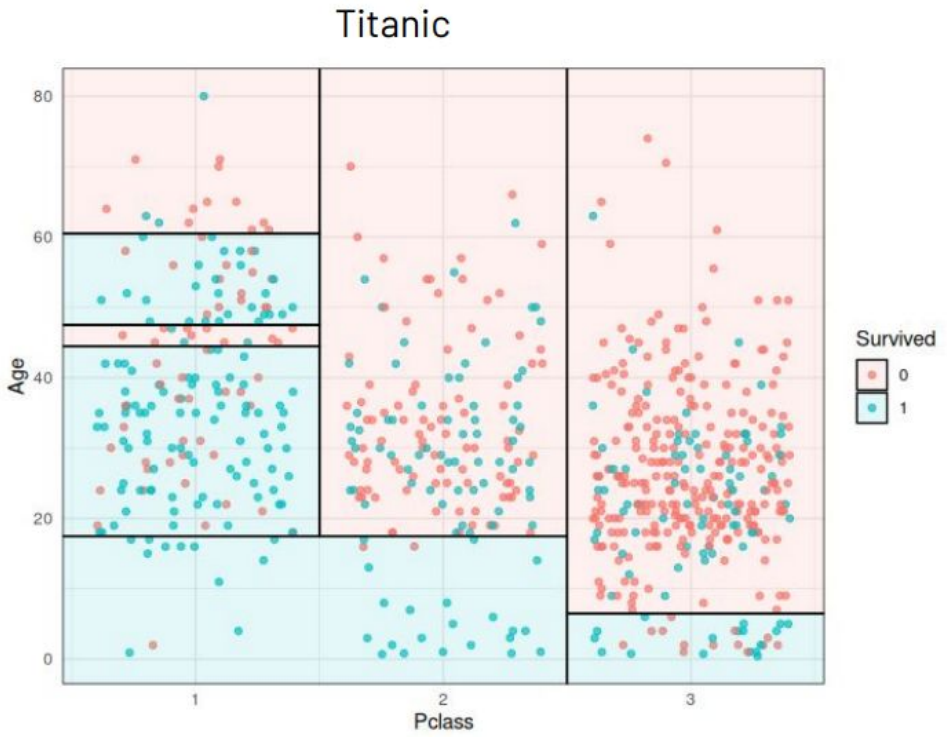
Se denomina sesgo inductivo de un algoritmo de aprendizaje automático al conjunto de afirmaciones que el algoritmo utiliza para construir un modelo.

- Incluye:
  - forma de las hipótesis (número y tipo de parámetros)
  - características de funcionamiento del algoritmo (cómo recorre el espacio de hipótesis hasta elegir un único modelo)

Sesgo inductivo

Hay muchos posibles árboles para un conjunto de datos de entrenamiento. ¿Cómo se elige uno?

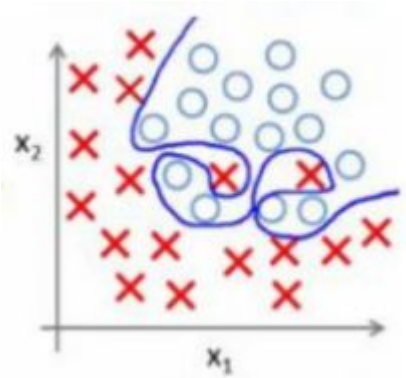
1) El tipo de regiones de decisión que puede generar tienen forma de rectángulos.
2) Las regiones que exploramos se determinan de manera Greedy. En cada paso optamos por mejorar la métrica de manera local.



Overfitting - Sobreajuste

En árboles de decisión, hay sobreajuste cuando el árbol es “demasiado” profundo

¿Qué pasa si hay **descripciones exactas de instancias únicas y aisladas**?



Fernando Berzal, DECSAI, Universidad de Granada

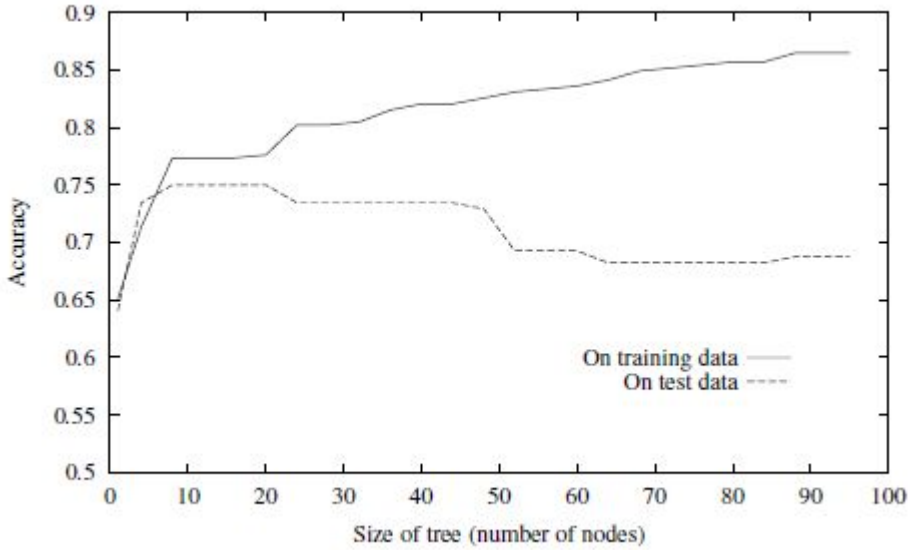
Overfitting

Datos:

- entrenamiento
- test (independiente)

Accuracy (exactitud):

(TP+TN)/(TP+TN+FP+FN)



Machine Learning, Tom M.Mitchel,McGrawHil,1997

Overfitting en Árboles de Decisión - Cómo evitarlo

Soluciones:

- **detener crecimiento del árbol** antes de que clasifique perfectamente a los datos
- hacer crecer el **árbol entero, luego podar (post-prune)**

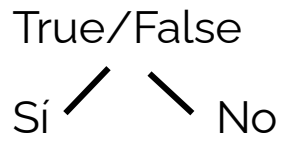
Atributos de valores continuos

Si tenemos un atributo A numérico, lo **discretizamos**. Definimos nuevos atributos discretos que particionan los valores de A en intervalos discretos.

Buscamos un umbral t y discriminamos en función de si A < t.

¿Cómo elegir t?:
1) se ordenan instancias de menor a mayor A
2) se busca forma de partir la lista, de forma tal que maximice la reducción de impureza (umbrales posibles: 6,11,25,33), ej. (22+28)/2

Temperatura	3	5	7	15	22	28	32	34
¿Corre?	Sí	Sí	No	Sí	Sí	No	No	Sí



Existen **extensiones** para particionar atributos contínuos en múltiples intervalos

Resumen

- aprendizaje supervisado.
- para clasificación y regresión
- fáciles de usar y de entender
- buen método exploratorio para ver qué atributos son importantes
- sesgo, overfitting

Ventajas:

- son altamente interpretables
- fácil visualización
- se pueden usar atributos binarios, categóricos o continuos

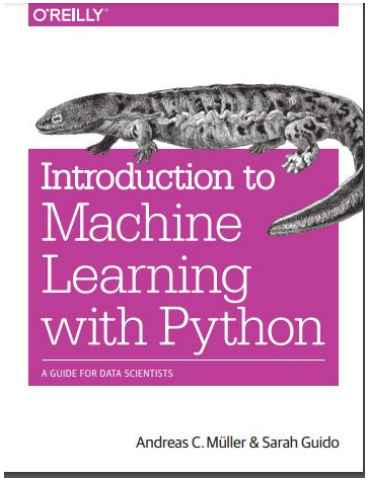
Desventajas:

- pueden tener sobreajuste
- suelen necesitarse ensambles de árboles para tener mejor performance

Bibliografía

Mitchell

Müller-Guido



Artículos:

Induction of Decision Trees. Quinlan. http://hunch.net/~coms-4771/quinlan.pdf

Simplifying Decision Trees. Quinlan.
https://www.sciencedirect.com/science/article/pii/S0020737387800536

En Python

class `sklearn.tree.`**`DecisionTreeClassifier`**(**, criterion='gini', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0, class_weight=None, ccp_alpha=0.0*) [source]

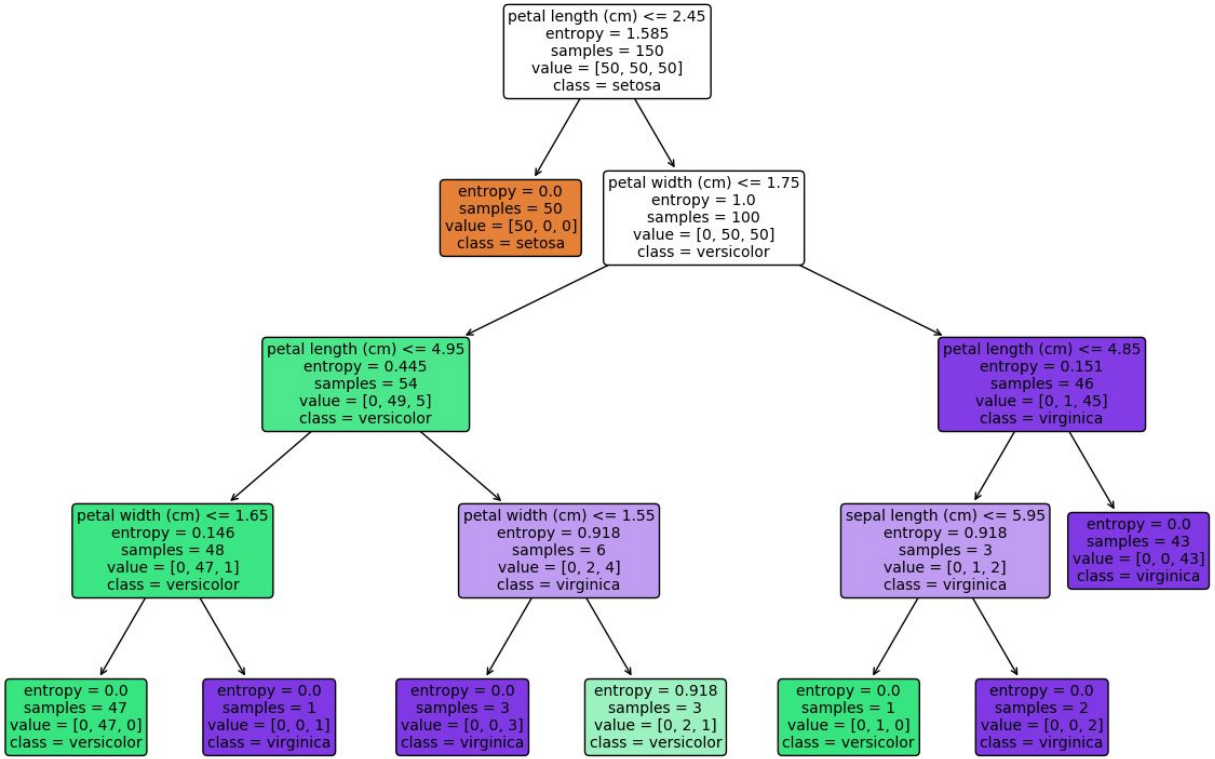
```python
from sklearn.tree import DecisionTreeClassifier

arbol = DecisionTreeClassifier()
arbol.fit(X, y) # Entrenamiento del modelo

prediction = arbol.predict(X) # Generamos las predicciones // llamamos al modelo
```

Hiperparámetros
→ Son los parámetros que nos permiten configurar el modelo.
Por ejemplo, la altura máx del árbol de decisión.

Ejemplo con Iris



Tarea

Resolver la guía de ejercicios de clasificación

Bibliografía

- Müller & Guido, "Introduction to Machine Learning with Python", O'Reilly, 2016.
- Mitchell, "Machine Learning", McGraw-Hill, 1997.
- James, Witten, Hastie, Tibshirani, & Taylor, "An Introduction to Statistical Learning with Applications in Python", Springer Nature, 2023.
- Alpaydin, “Introduction to Machine Learning”, 2010.