



DEPARTAMENTO
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Laboratorio de datos

Clasificación

2do Cuatrimestre 2024

Laje, López-Rosenfeld, de Erausquin

(Agradecimientos: Manuela Cerdeiro - Paula Perez Bianchi - Viviana Cotik)

“... 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

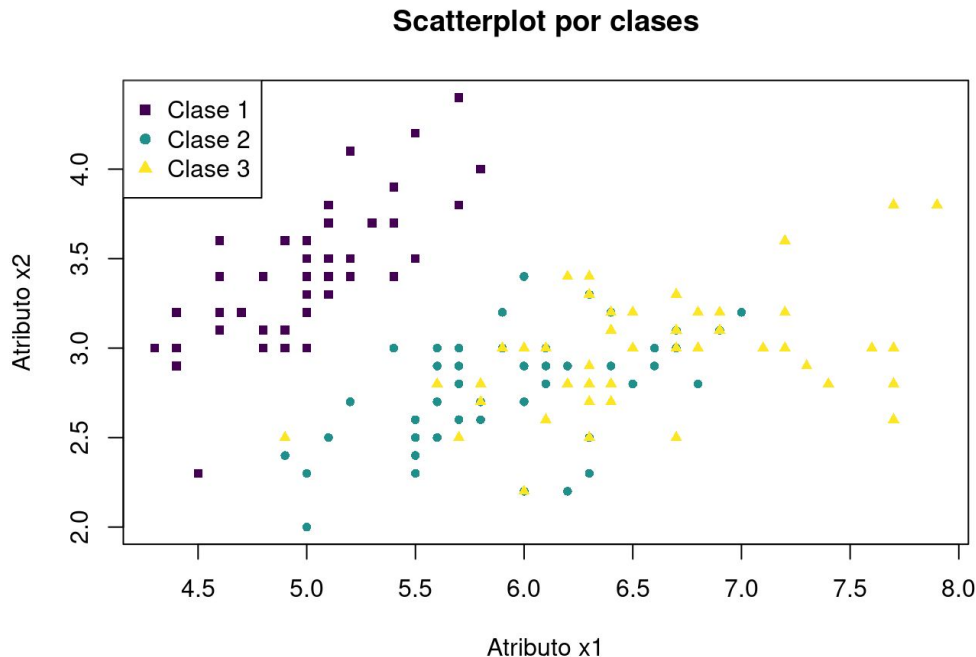
Ejemplo

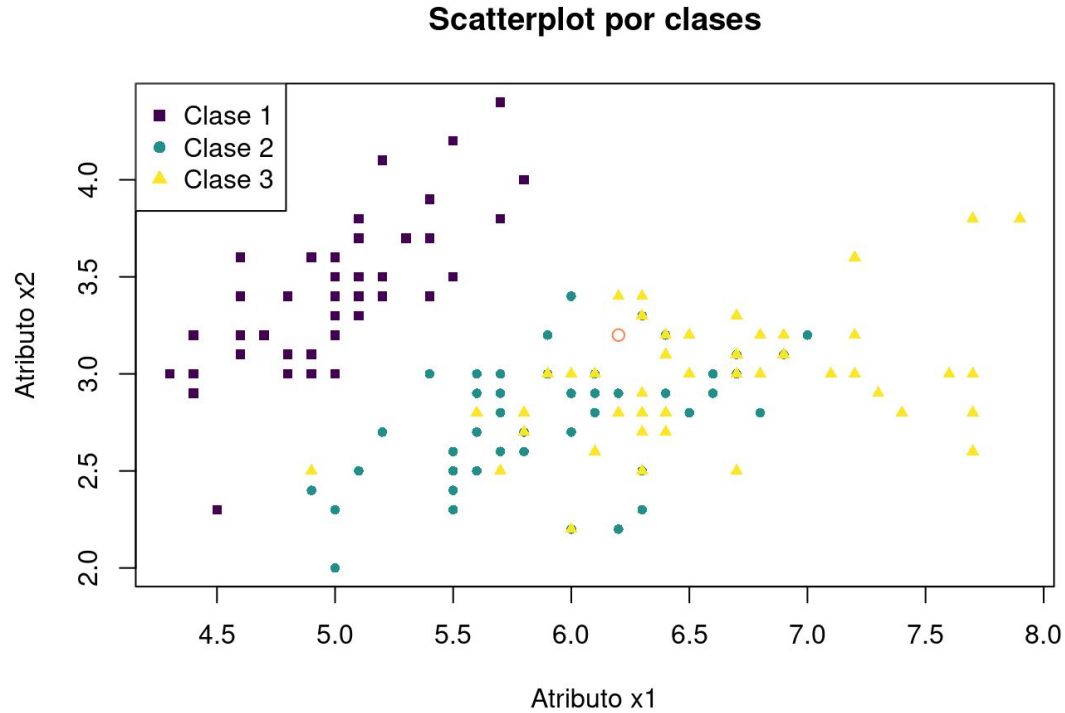
Tenemos un conjunto de datos con variables x_1 , x_2 , y .

- Variables explicativas continuas x_1 , x_2
- Variable a explicar y categórica

Graficamos x_1 , x_2 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), determinar “a ciegas” la etiqueta en 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.
- + En el caso de las variables sí/no, hablamos de falsos positivos, o falsos negativos.

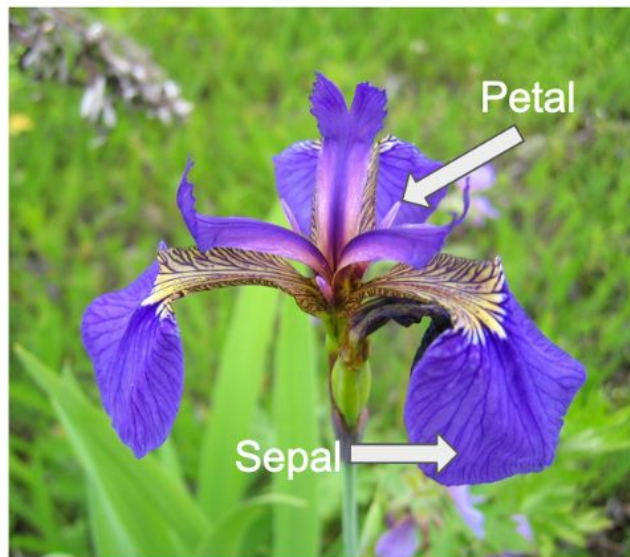
Métodos posibles (hay más)

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

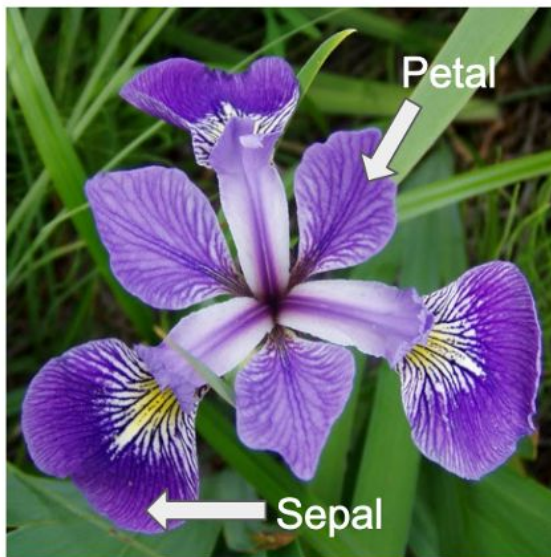
Clasificación por umbrales

Dataset de flores - Iris

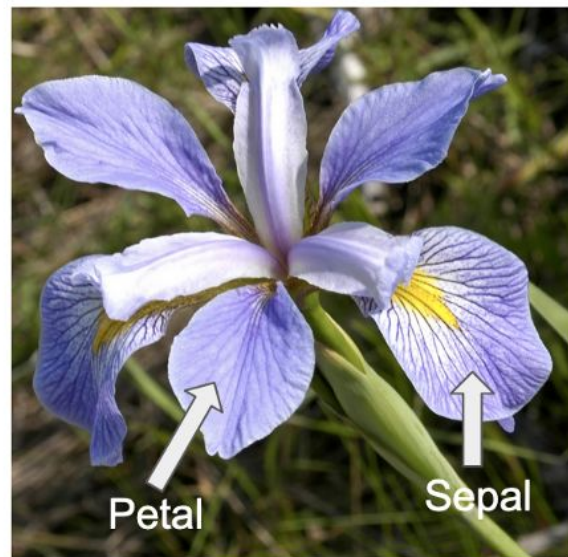
Iris setosa



Iris versicolor



Iris virginica

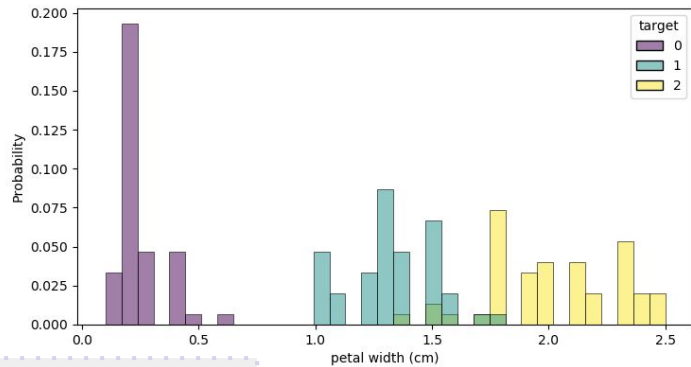
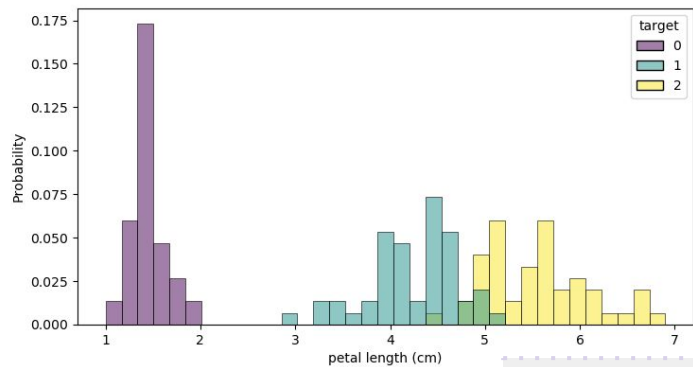
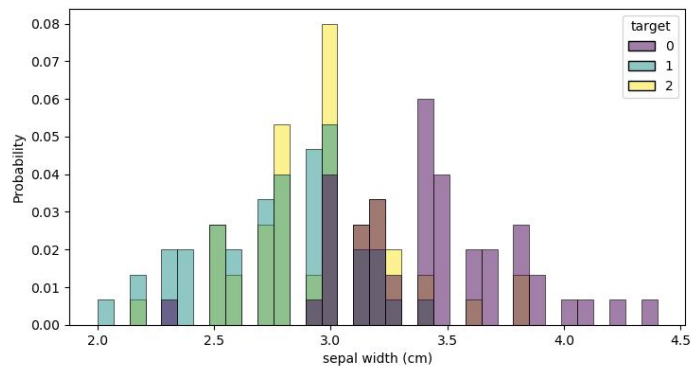
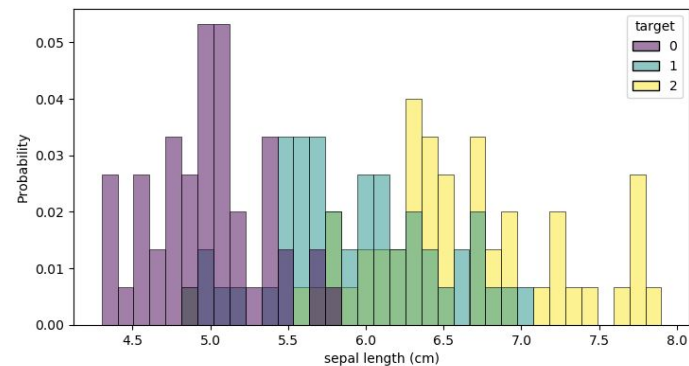


50 muestras de cada una de tres especies de flores *Iris*: *setosa*, *versicolor* y *virginica*. De cada flor se midieron 4 atributos: largo y ancho del sépalo y del pétalo.

Fisher - 1936

Histogramas

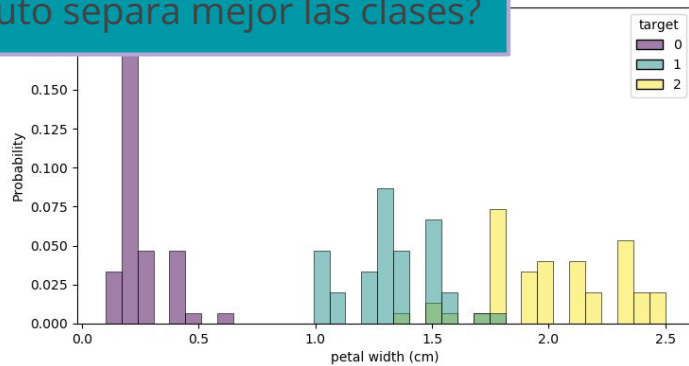
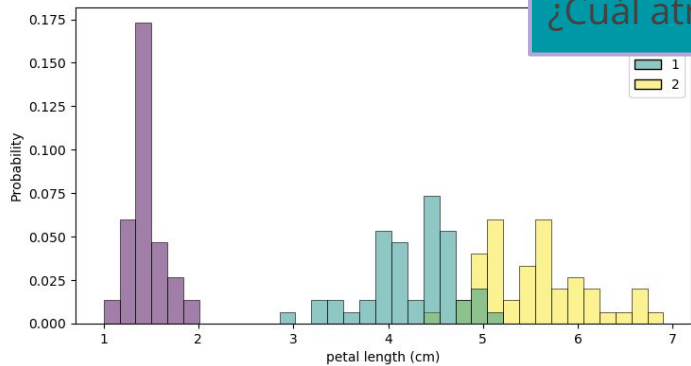
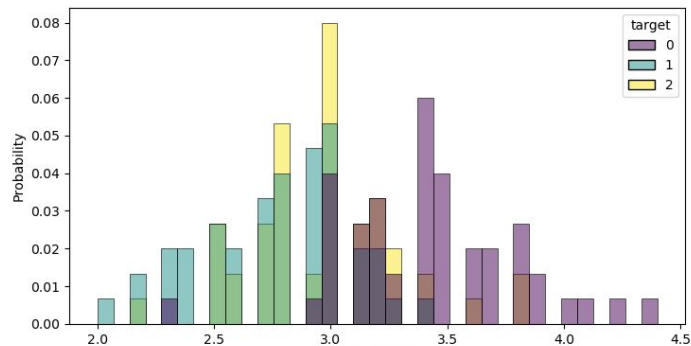
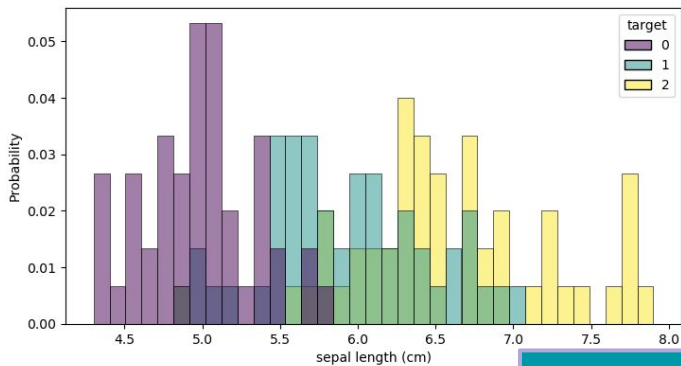
Histogramas de los 4 atributos



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

Histogramas

Histogramas de los 4 atributos



¿Cuál atributo separa mejor las clases?

Clasificamos por largo del pétalo

```
def clasificador_iris
```



Clasificamos por largo del pétalo

Ahora veamos cómo se comporta este clasificador.

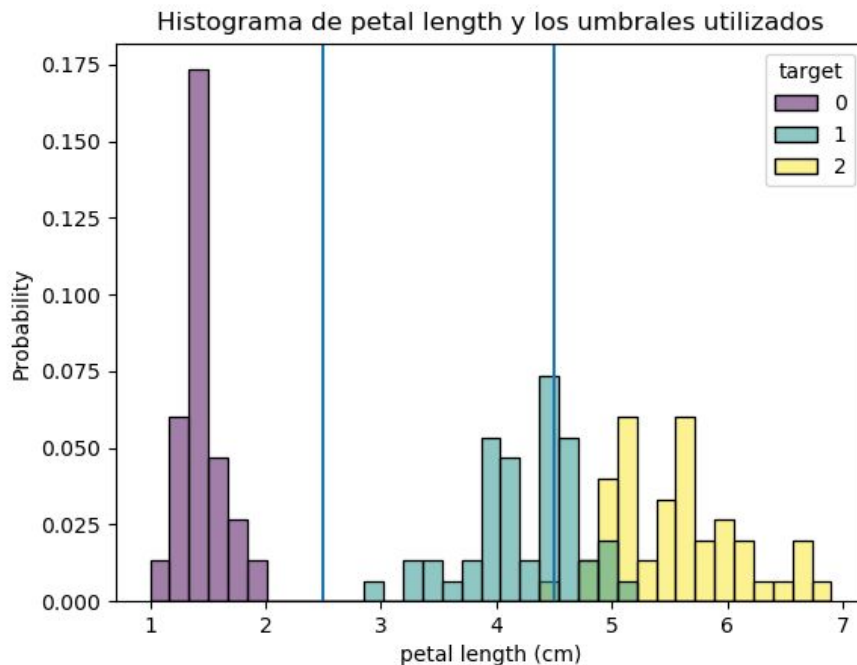


¿Podemos mejorar el clasificador?
¿Cómo comparamos entre dos clasificadores?

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.

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

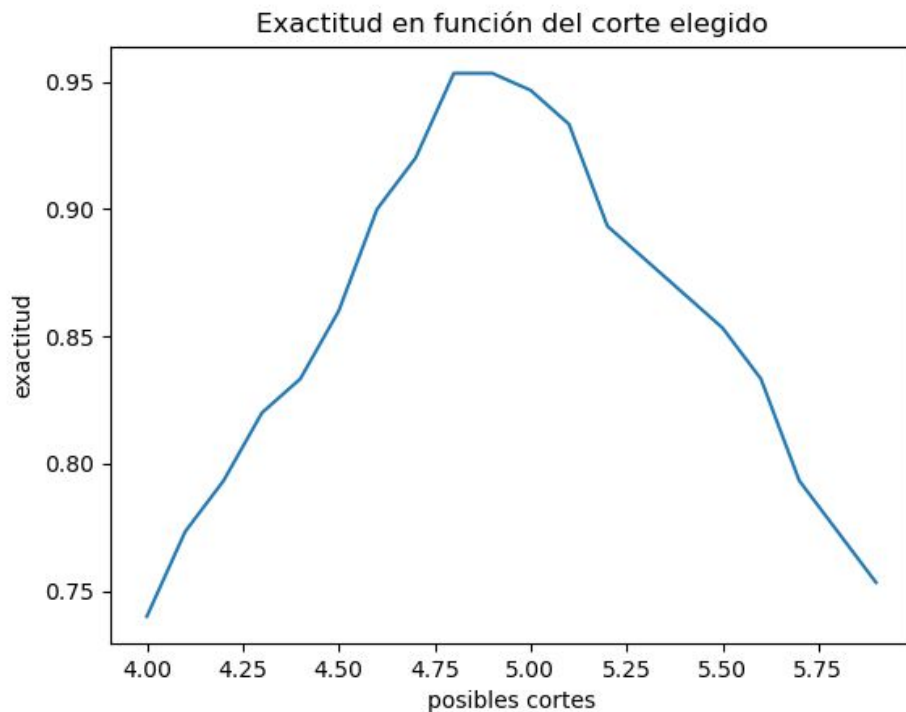
$$50 + 29 + 50 = 129$$
$$129 / 150 = 0.86$$

¿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

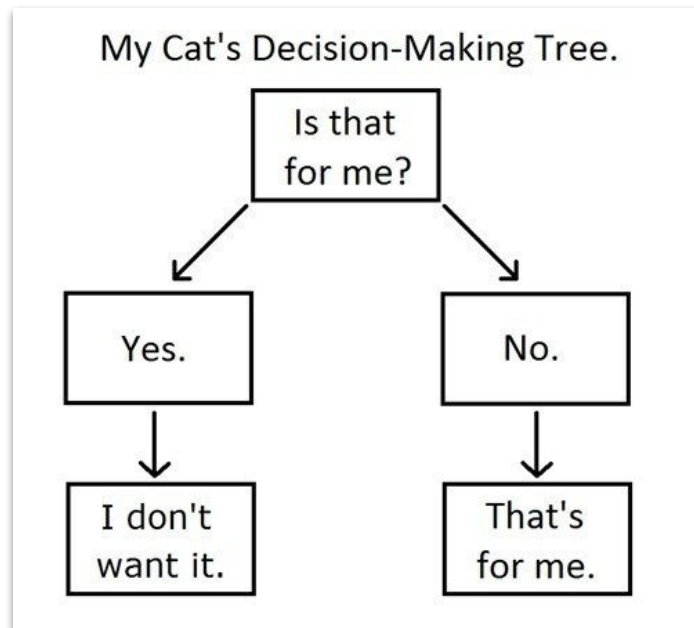
Comparación de clasificadores



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


Árboles de decisión

- 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



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 áltamente 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

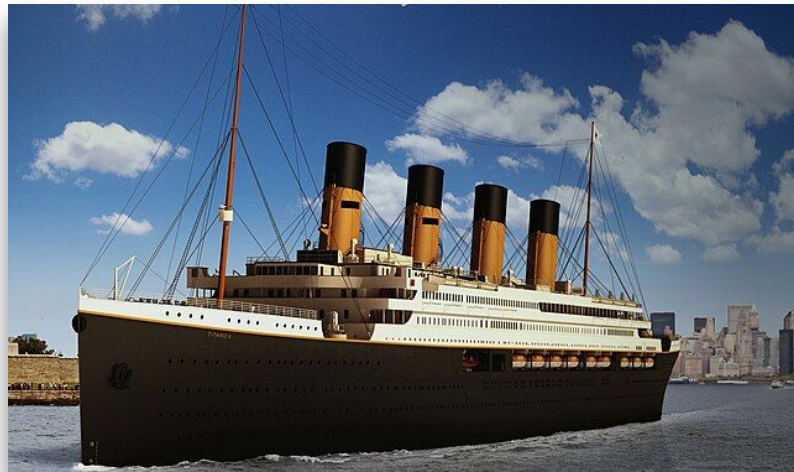
El Titanic se hundió en Abril de 1912.

Tenía la fama de 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? ¡Explore los datos y escribanlas en un papel (o gráfiquenlas)!

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.

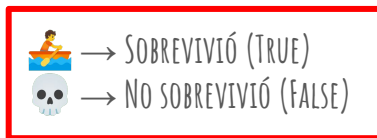
```
def clasificador_titanic(x):  
    vive = False  
    if REGLA:  
        vive = True  
    return vive
```

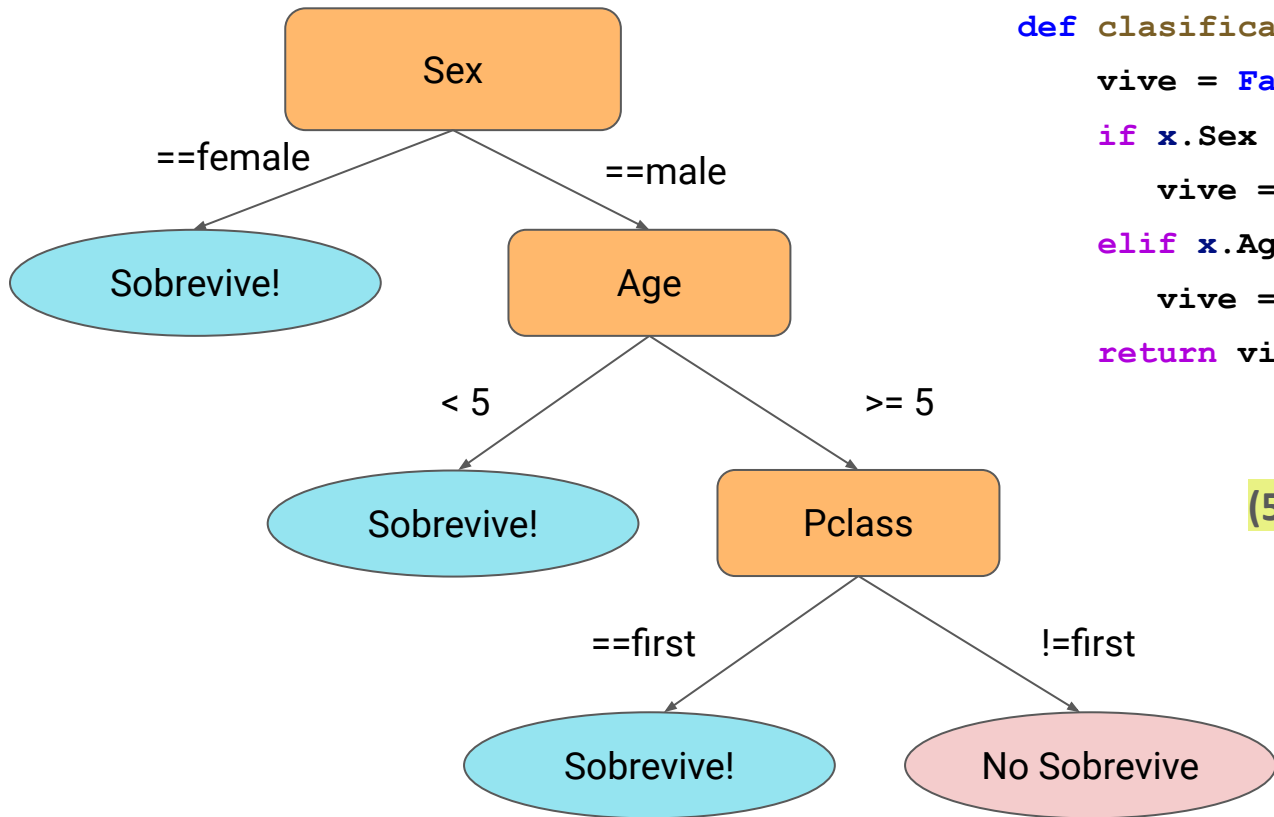
Tienen 10 min!

<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/tUDoVg3YuYg6ALLY6>





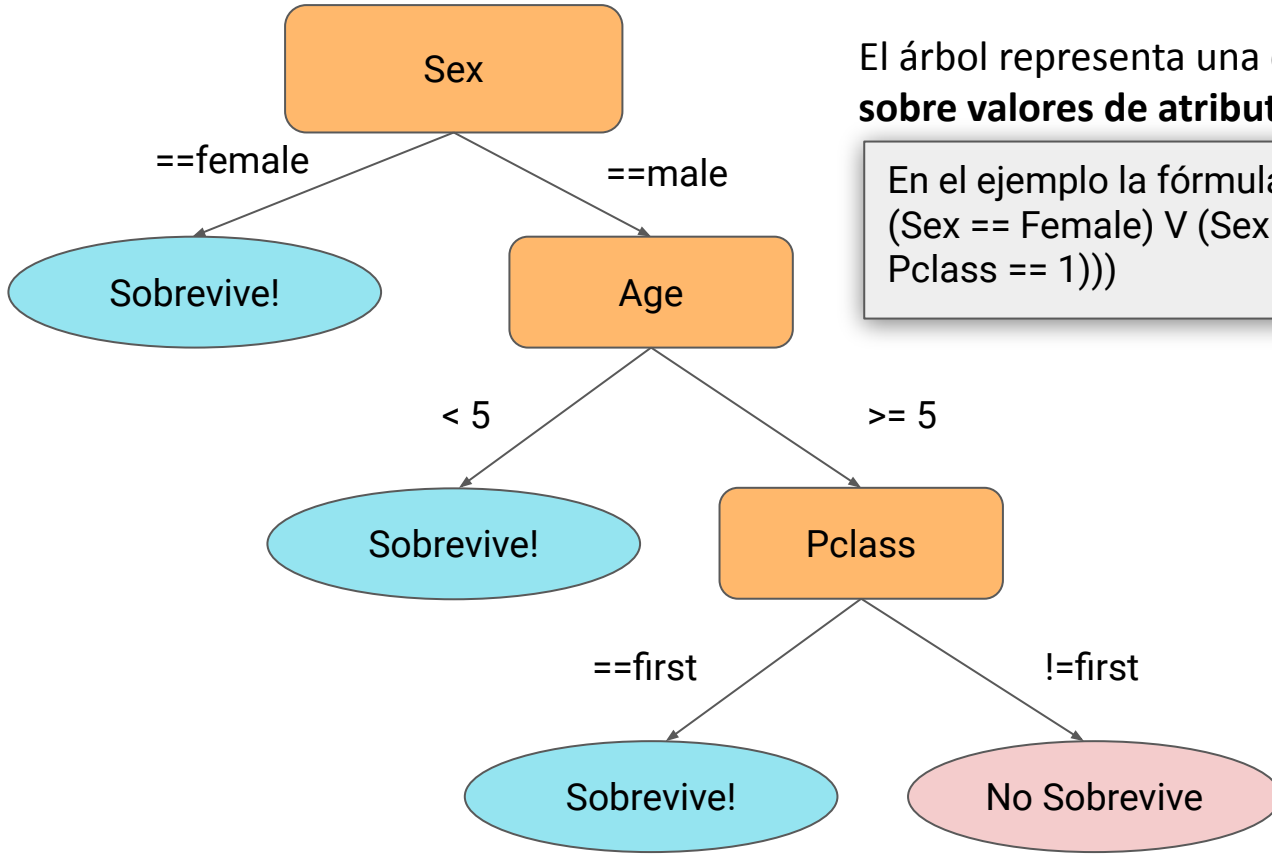
```
def clasificador_titanic(x):  
    vive = False  
    if x.Sex == "female":  
        vive = True  
    elif x.Age < 5 or x.Pclass == 1 :  
        vive = True  
    return vive
```

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:

$$(\text{Sex} == \text{Female}) \vee (\text{Sex} == \text{Male} \wedge ((\text{Age} < 5) \vee (\text{Age} \geq 5 \wedge \text{Pclass} == 1)))$$

Competencia - Evaluamos los clasificadores.

Vemos respuestas del form.

Evaluamos también con un conjunto de test.

Accuracy - cantidad de errores de cada tipo?

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** \in **A**, el atributo que produce el **mejor corte** de **S** para el nodo actual.
2. Para cada valor **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**.

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

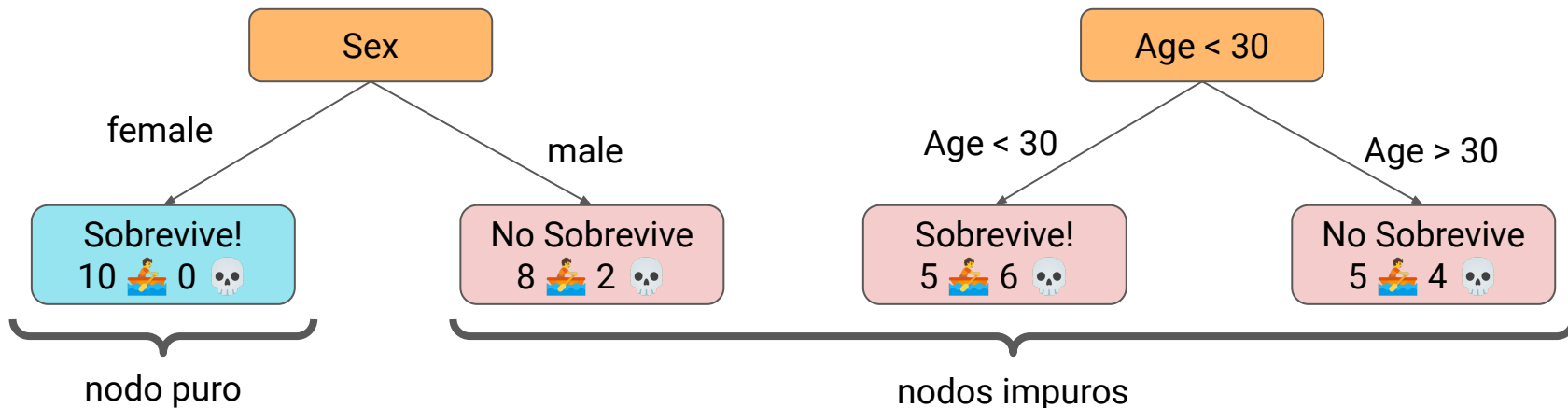
4. Repetir para cada hijo del nodo en el que haya instancias de más de una clase (salvo que se cumpla algún criterio de corte en cuyo caso terminamos).

*¿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



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

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

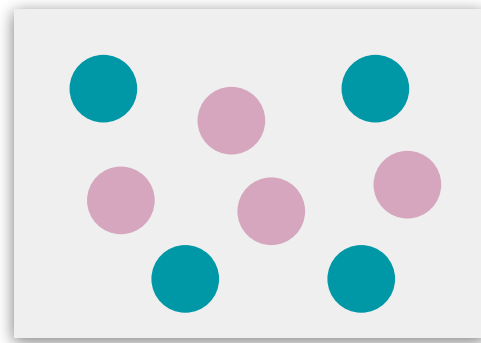
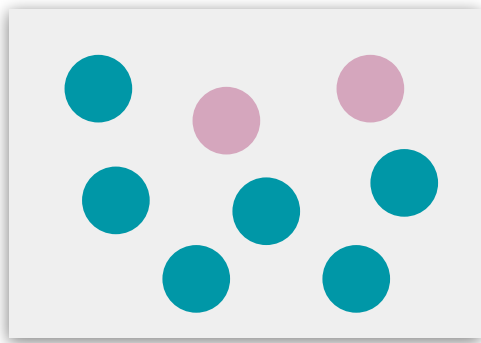
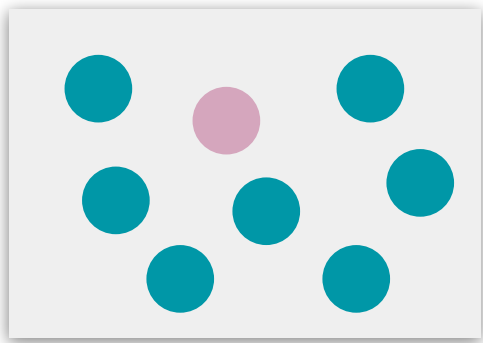
“cuánto gano si divido a **S** en regiones **S**_{c,≤} y **S**_{c,>} según la medida **M**” (caso binario)

Árboles de decisión - *Medidas de impureza*

Entropía

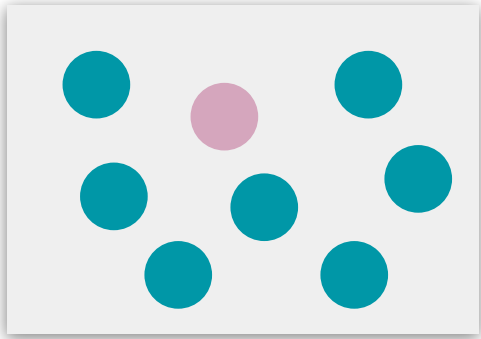
Mide la incertidumbre inherente a los posibles resultados de una variable aleatoria (\sim información de Shannon).

Voy a sacar una bolita. ¿Cuánta incertidumbre hay sobre el resultado?

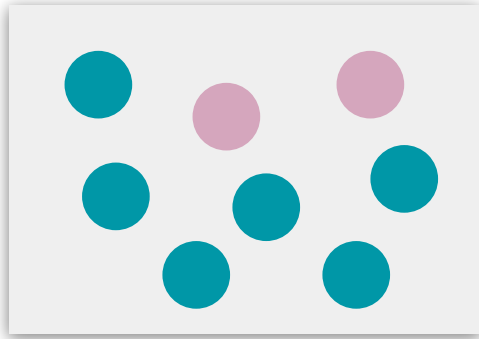


$$H = - \sum p_i \log_2(p_i)$$

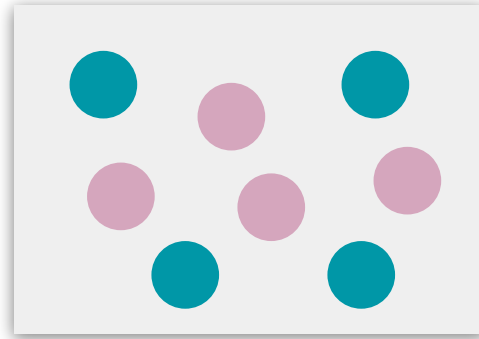
$$p_A = 7/8, p_R = 1/8$$



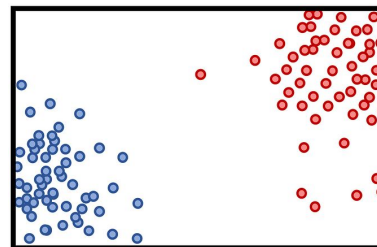
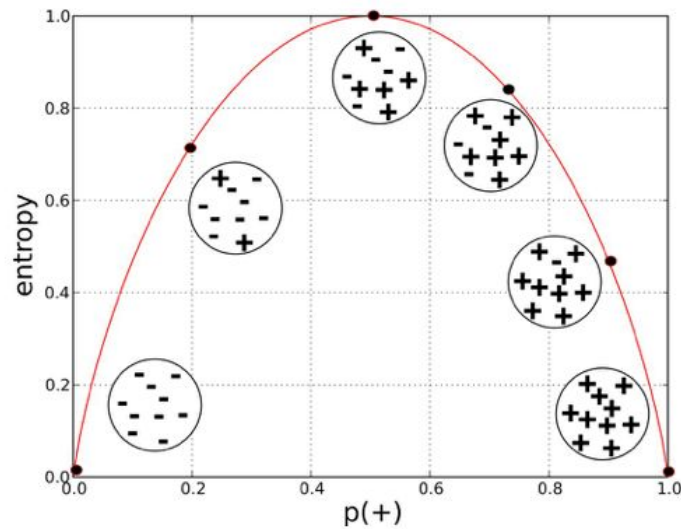
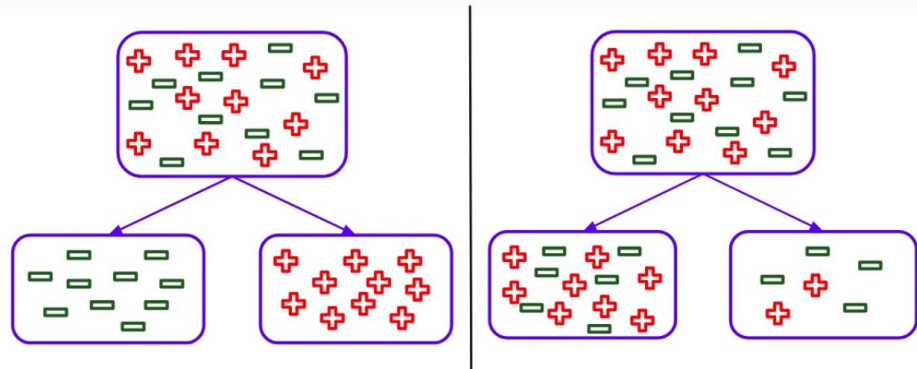
$$p_A = 6/8, p_R = 2/8$$



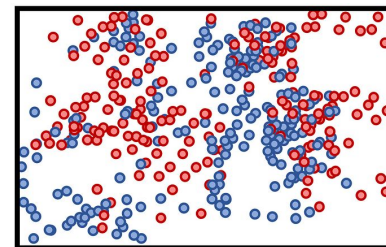
$$p_A = 4/8, p_R = 4/8$$



Árboles de decisión - *Medidas de impureza*



Low Entropy



High Entropy

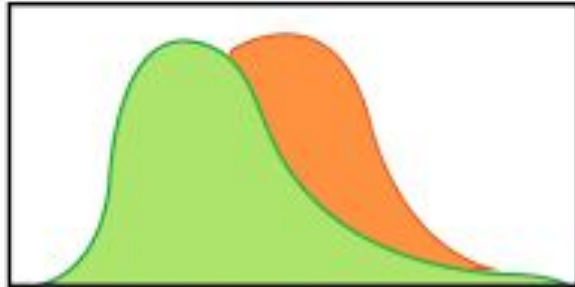
Árboles de decisión - *Medidas de impureza*

Info Gain (Ganancia de Información)

Mide Cuánta entropía removemos al hacer un corte.

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

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



Low information gain High entropy



High information gain Low entropy

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.

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

$$Gini_Gain(S, \langle a, c \rangle) = Gini(S) - (Prop_{\leq} \cdot Gini(S_{\leq}) + Prop_{>} \cdot Gini(S_{>}))$$

Sesgo inductivo

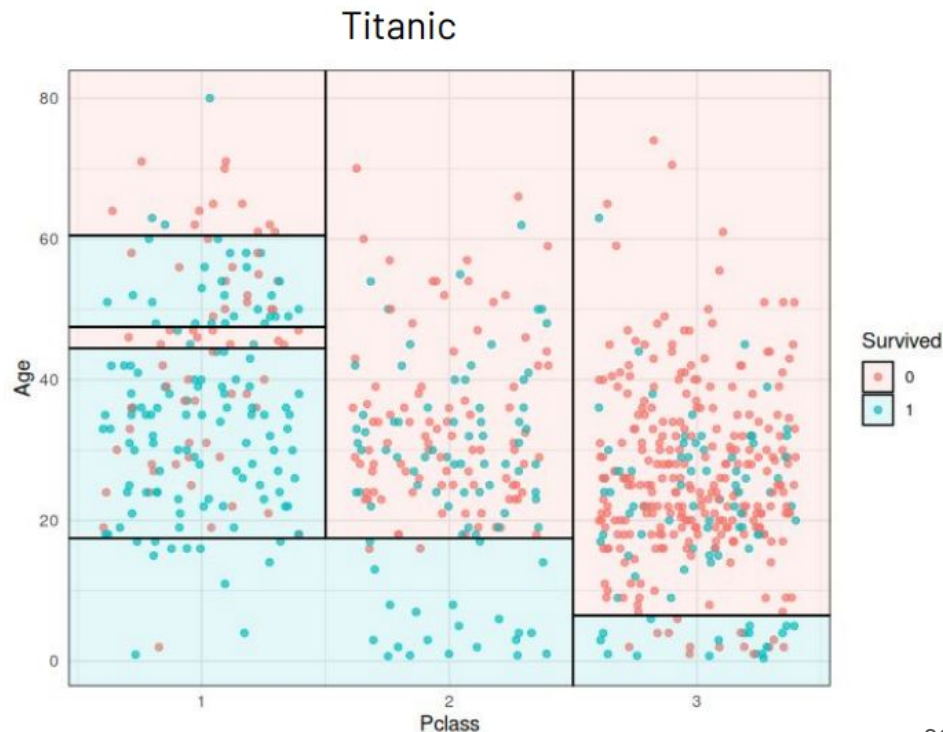
Se denomina sesgo inductivo de un algoritmo de aprendizaje automático al conjunto de afirmaciones que el algoritmo utiliza para construir un modelo.

- 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)
- Si un algoritmo de aprendizaje tiene sesgo fuerte:
 - Mayores restricciones al poder expresivo de las hipótesis.
 - Suele aproximar peor a la función objetivo.

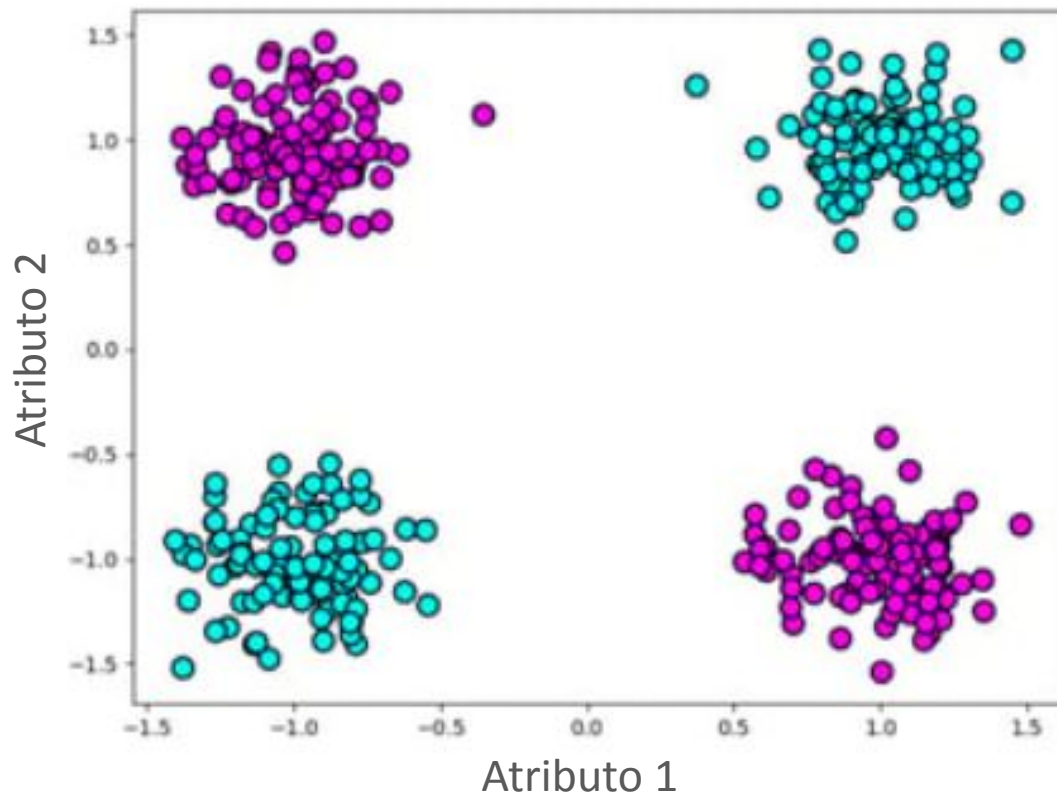
Sesgo inductivo

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 (=ávida, glotona): en cada paso optamos por mejorar la métrica de manera local.



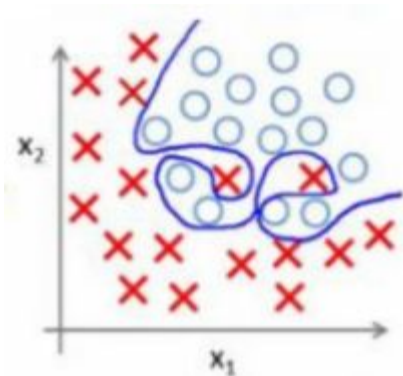
Claramente clasificable por árbol, difícil de automatizar



XOR

¿Cómo hago para entrenar de manera automática un árbol como los que estamos considerando con este conjunto de datos? (el algoritmo de entrenamiento no va a encontrar ningún atributo que por sí solo me permita empezar una clasificación)

Overfitting - Sobreajuste



Fernando Berzal, DECSAI,
Universidad de Granada

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

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

Arbol demasiado profundo (sobreajustado = da cuenta de todos los “casos particulares” correctamente) → bajo sesgo, alta variabilidad ante datos nuevos, no generaliza bien

Arbol demasiado chato → alto sesgo, baja variabilidad, no llega a interpretar los grandes patrones

Overfitting - Sobreajuste

Datos:

- entrenamiento
- test (independiente)

Accuracy (exactitud):

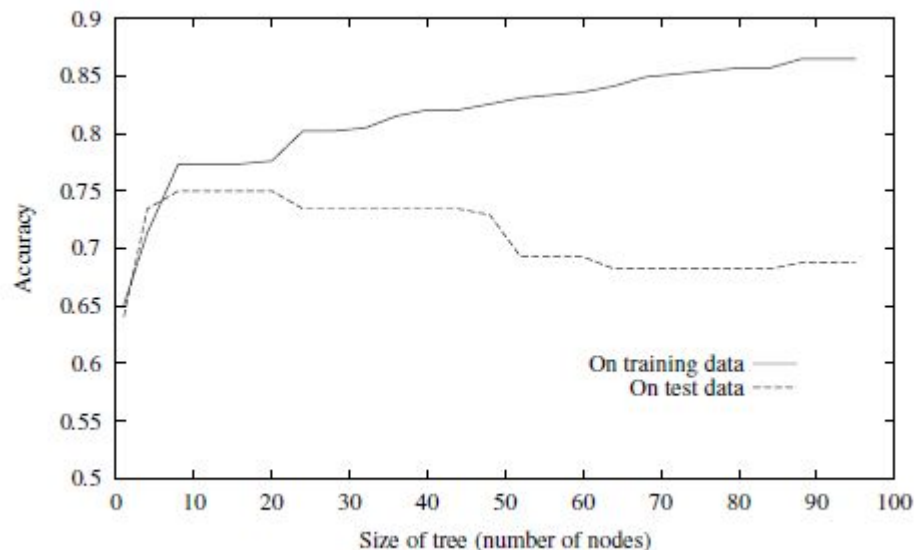
$$(TP+TN)/(TP+TN+FP+FN)$$

TP = True Positives (positivos correctos)

TN = True Negatives (negativos correctos)

FP = False Positives (falsos positivos)

FN = False Negatives (falsos negativos)



Machine Learning, Tom M. Mitchell, McGraw-Hill, 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

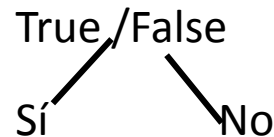
Si tenemos un atributo A numérico, lo **discretizamos**. Definimos nuevos atributos discretos que particionan los valores de A en intervalos discretos.

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 la forma de partir la lista tal que maximice la reducción de impureza (umbrales posibles: 6,11,25,33), ej. $(22+28)/2$ ¿Temperatura < 25?

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



Existen **extensiones** para particionar atributos continuos 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

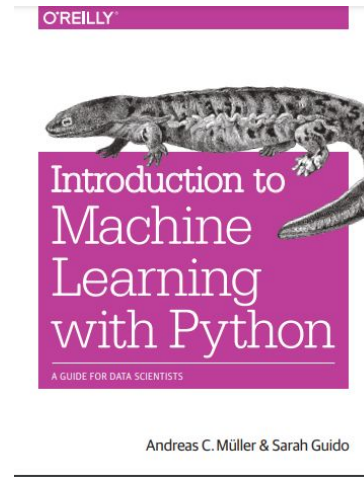
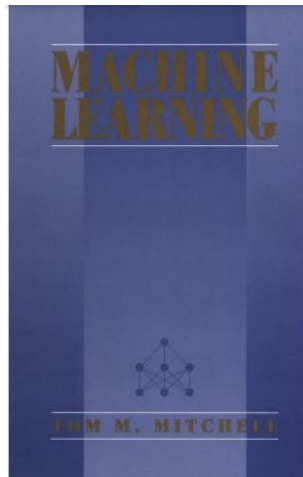
Desventajas:

- pueden tener sobreajuste
- suelen necesitarse ensambles de árboles (->“random forest”) 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)
```

[\[source\]](#)

```
from sklearn.tree import DecisionTreeClassifier
```

```
arbol = DecisionTreeClassifier()
```

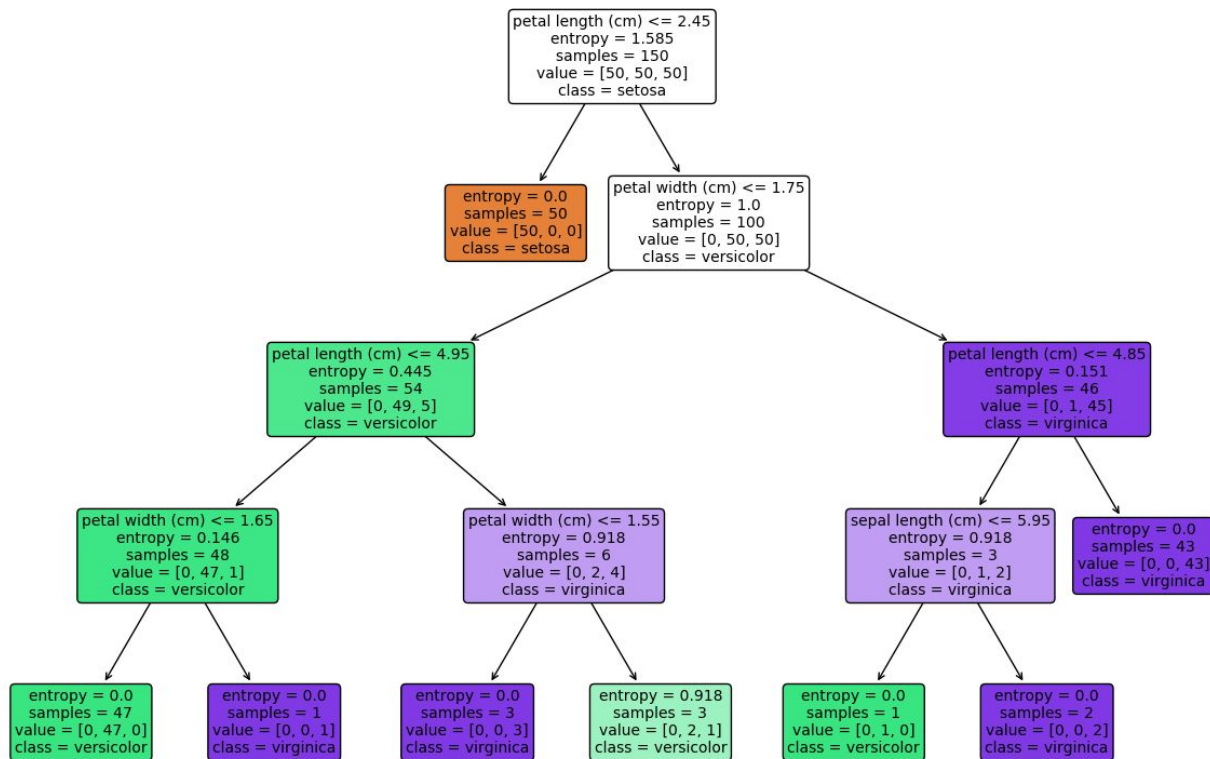
```
arbol.fit(X, y) # Entrenamiento del modelo
```

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

Hiperparámetros

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

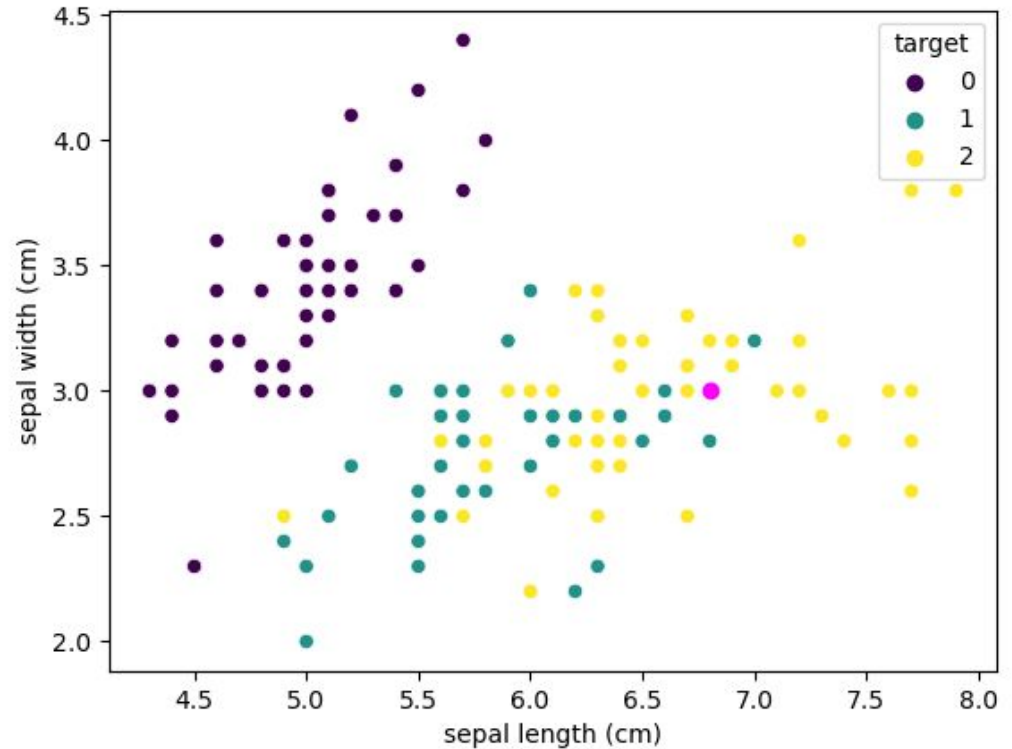
Ejemplo con Iris



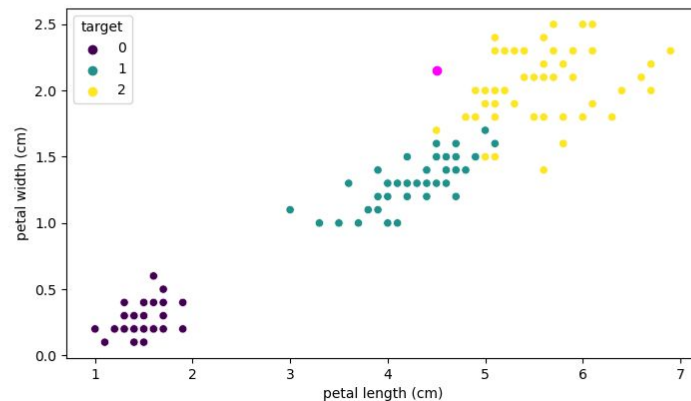
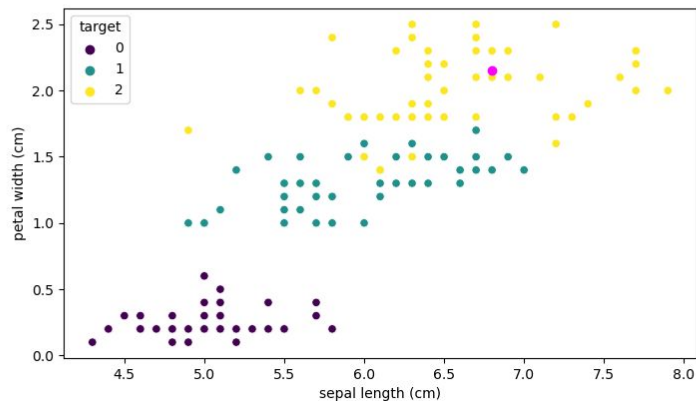
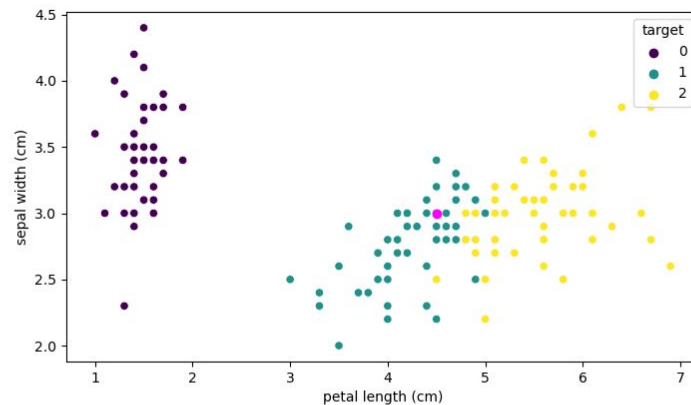
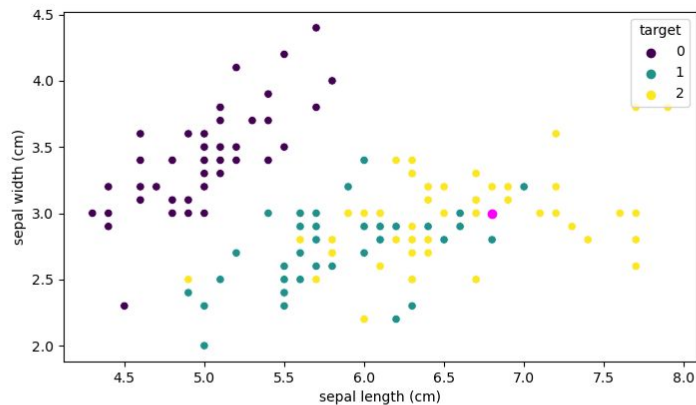
Predicción de dato nuevo

sepal length (cm) = 6.8
sepal width (cm) = 3
petal length (cm) = 4.5
petal width (cm) = 2.15

Clase predicha: 2



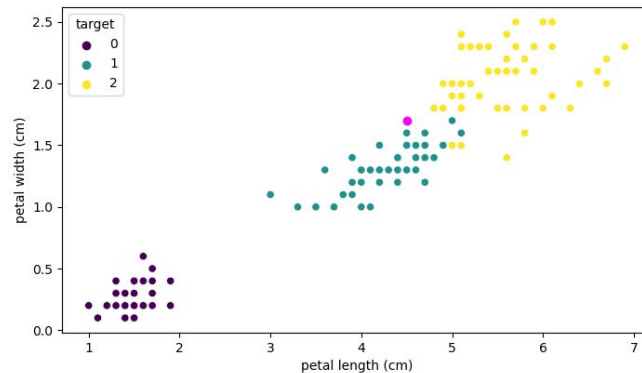
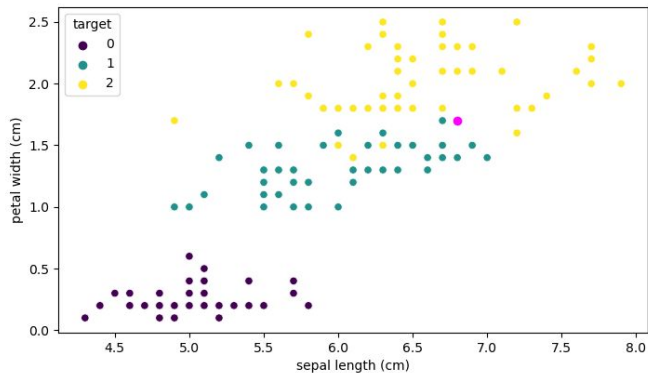
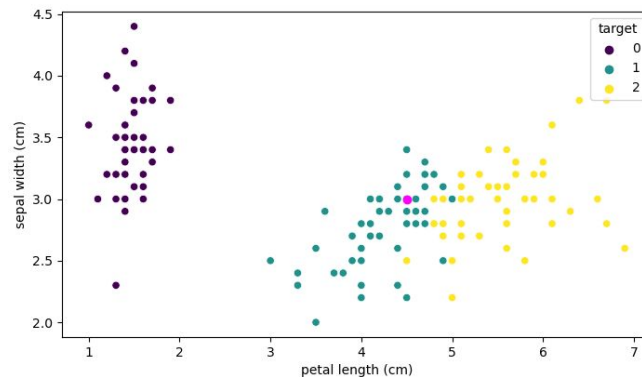
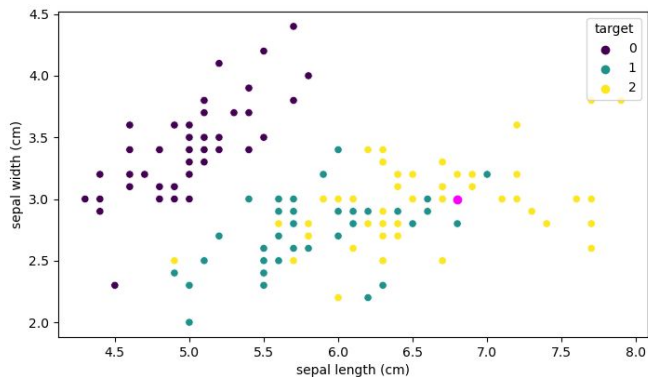
Predicción de dato nuevo



Predicción de dato nuevo (II)

sepal length (cm) = 6.8
sepal width (cm) = 3
petal length (cm) = 4.5
petal width (cm) = 1.7

Clase predicha: 2



Ejemplo con árboles
de la ciudad



Especies que vamos a considerar



JACARANDÁ

Especies que vamos a considerar



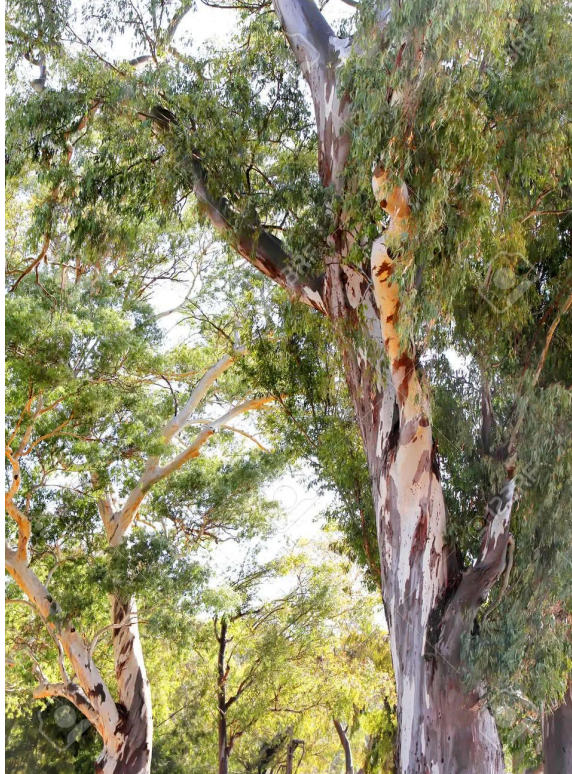
CEIBO

Especies que vamos a considerar



PINDÓ

Especies que vamos a considerar



EUCALIPTUS

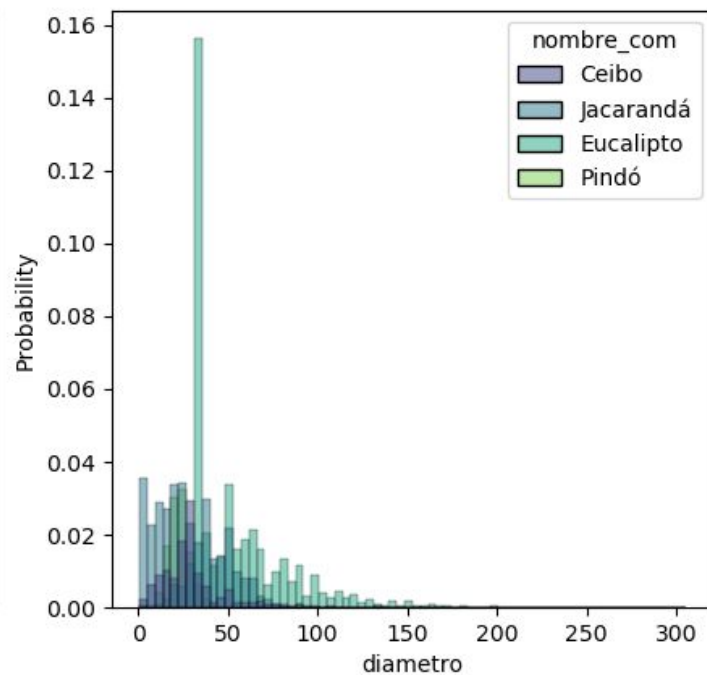
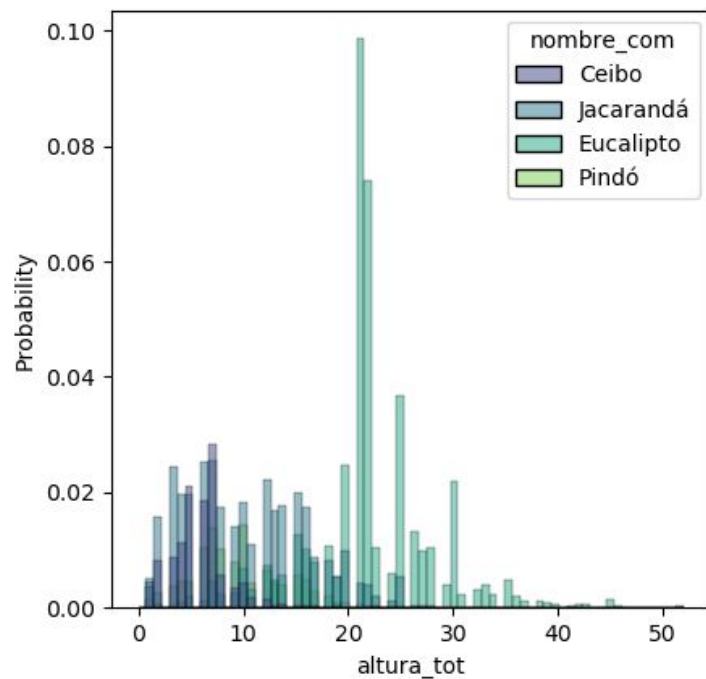
El dataframe

Está en el campus

Index	altura_tot	diametro	inclinacio	nombre_com
2319	17	45	9	Jacarandá
2320	2	9	0	Ceibo
2321	6	15	0	Pindó
2322	12	42	0	Jacarandá
2323	12	33	10	Jacarandá
2324	12	48	4	Jacarandá
2325	12	39	0	Jacarandá
2326	18	30	4	Pindó
2327	17	30	3	Pindó
2328	12	20	0	Pindó
2329	20	56	0	Eucalipto
2330	27	70	0	Eucalipto
2331	16	37	7	Jacarandá
2332	15	28	0	Jacarandá

Histogramas

Histogramas de altura y diámetro



Ejercicio

- + Graficar histogramas de los atributos
- + Graficar scatterplot diámetro-altura con colores por especie
- + Entrenar un árbol de decisión para clasificar árboles
 - probar con distintas profundidades
 - probar con criterio de información y de gini
 - probar no utilizar todos los atributos
- + En cada caso graficar el árbol de decisión y registrar la exactitud
- + Tengo un árbol que mide 22 m, diámetro 56 e inclinación 8°
¿cuál especie es?