



# Algoritmos de clasificación

Árboles, KNN, Naive Bayes

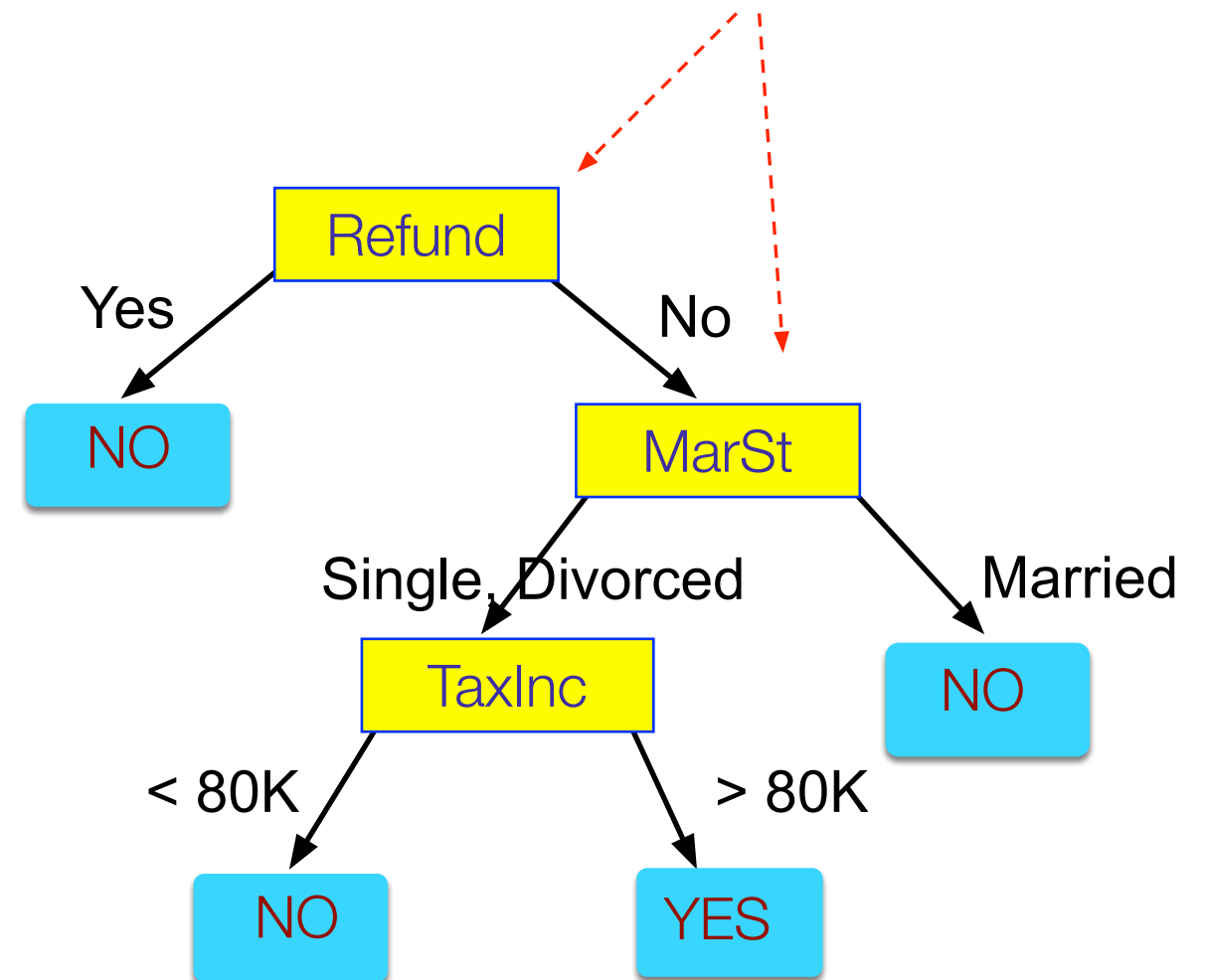
# Árbol de Decisión

categorical  
categorical  
continuous  
class

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Datos de  
Entrenamiento

**Atributos de Separación**



Modelo: Árbol de Decisión

# Árbol de Decisión

El árbol tiene tres tipos de nodos:

1. Un nodo raíz que no tiene arcos entrantes y tiene arcos salientes.
  2. Nodos internos, cada uno de los cuales tiene exactamente un arco entrante y dos o más arcos salientes.
  3. Nodos hoja o terminales, cada uno de los cuales tiene exactamente un arco entrante.
- A cada nodo de hoja se le asigna una **etiqueta** de clase.
  - Los nodos no terminales, que incluyen la raíz y otros nodos internos, contienen **tests** sobre los atributos para separar los ejemplos que tienen valores diferentes para esos atributos.
  - El árbol de decisión **fragmenta** el dataset de manera recursiva hasta asignar los ejemplos a una clase.

# Otro Ejemplo

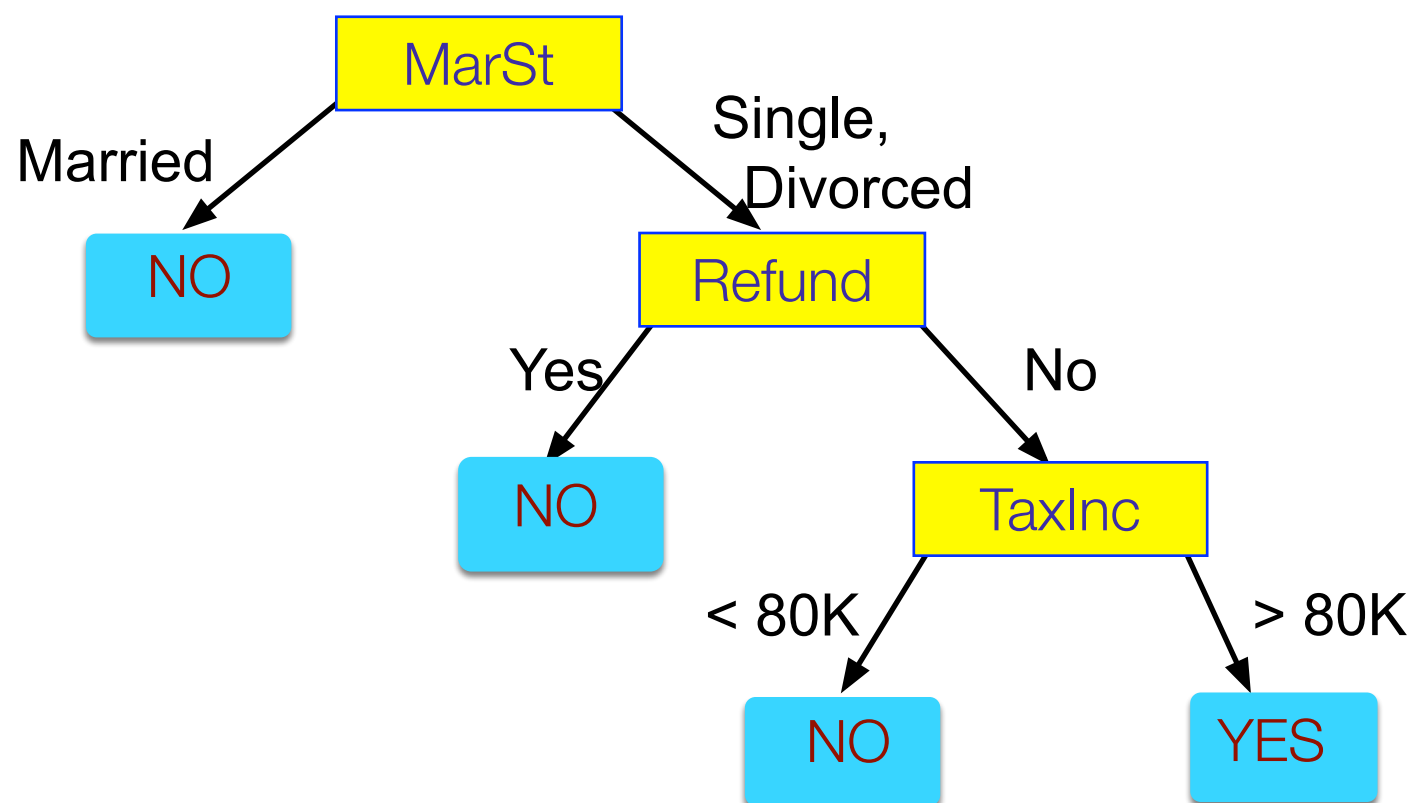
<i>Tid</i>	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

categorical

categorical

continuous

class



**¿Puede existir más de un árbol que se ajuste a los datos!**

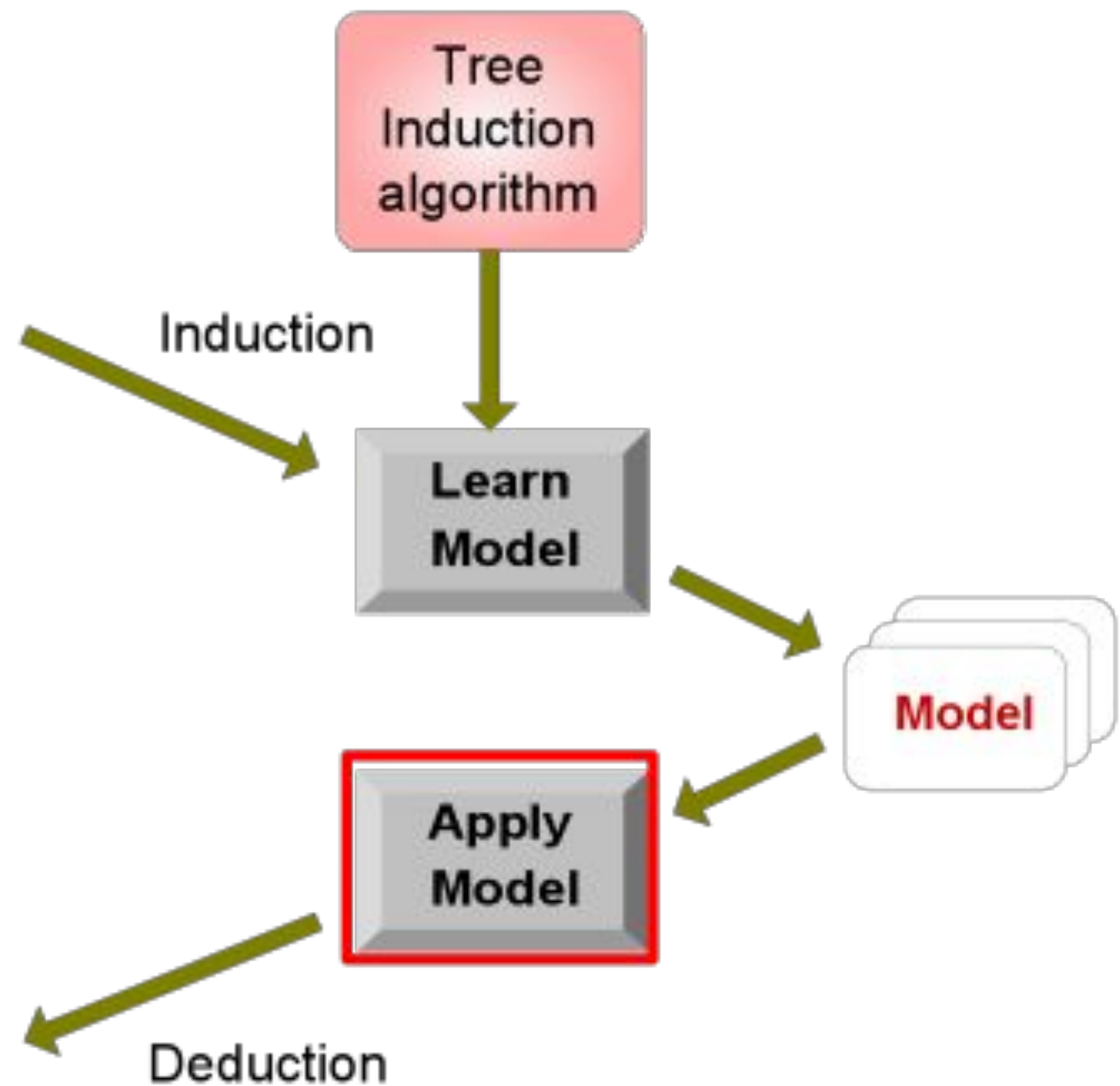
# Clasificando con un árbol de decisión

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set

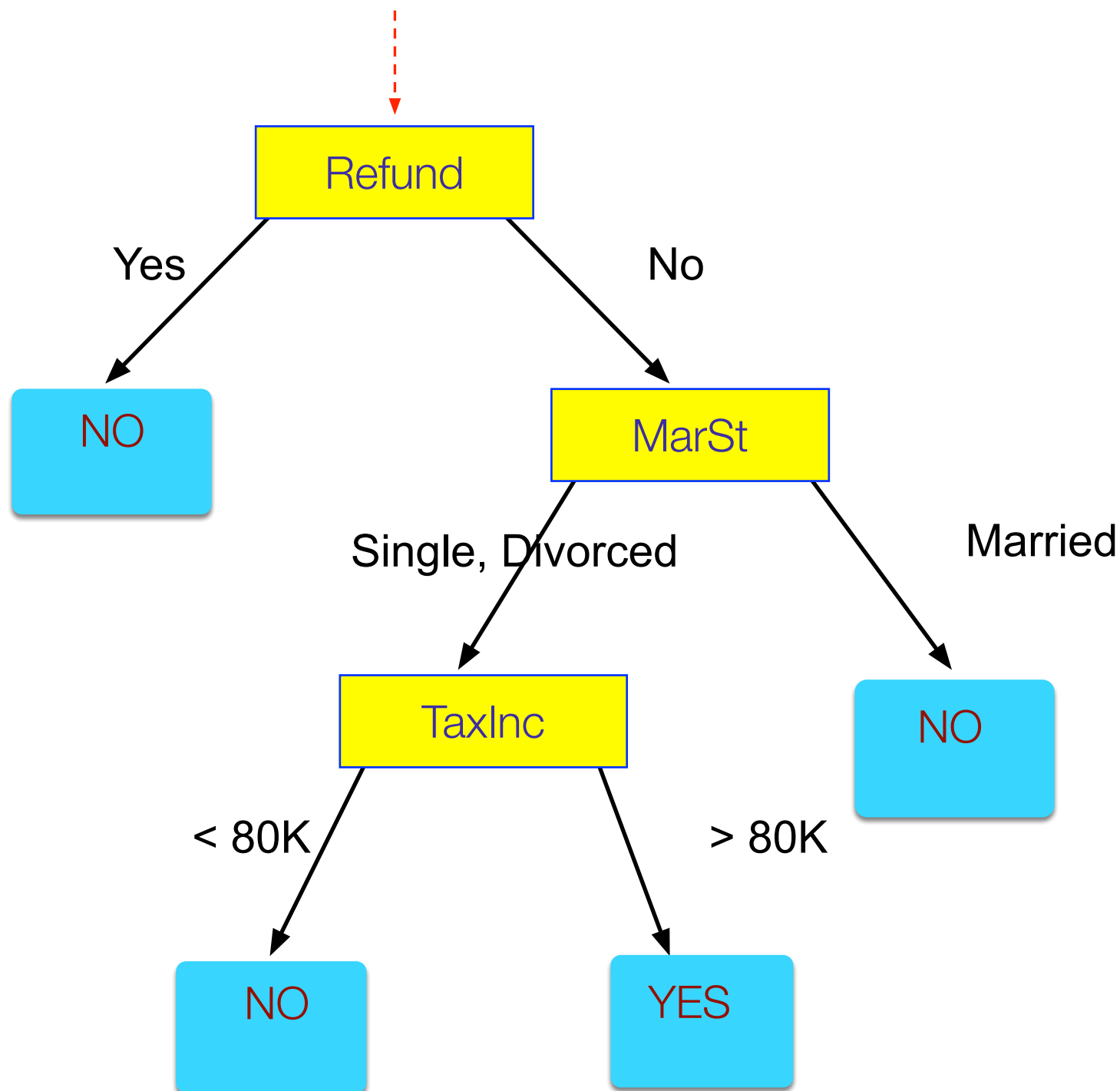


# Aplicamos el modelo

Comenzamos en la raíz

**Dato de Evaluación**

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



# Clasificando con un árbol de decisión

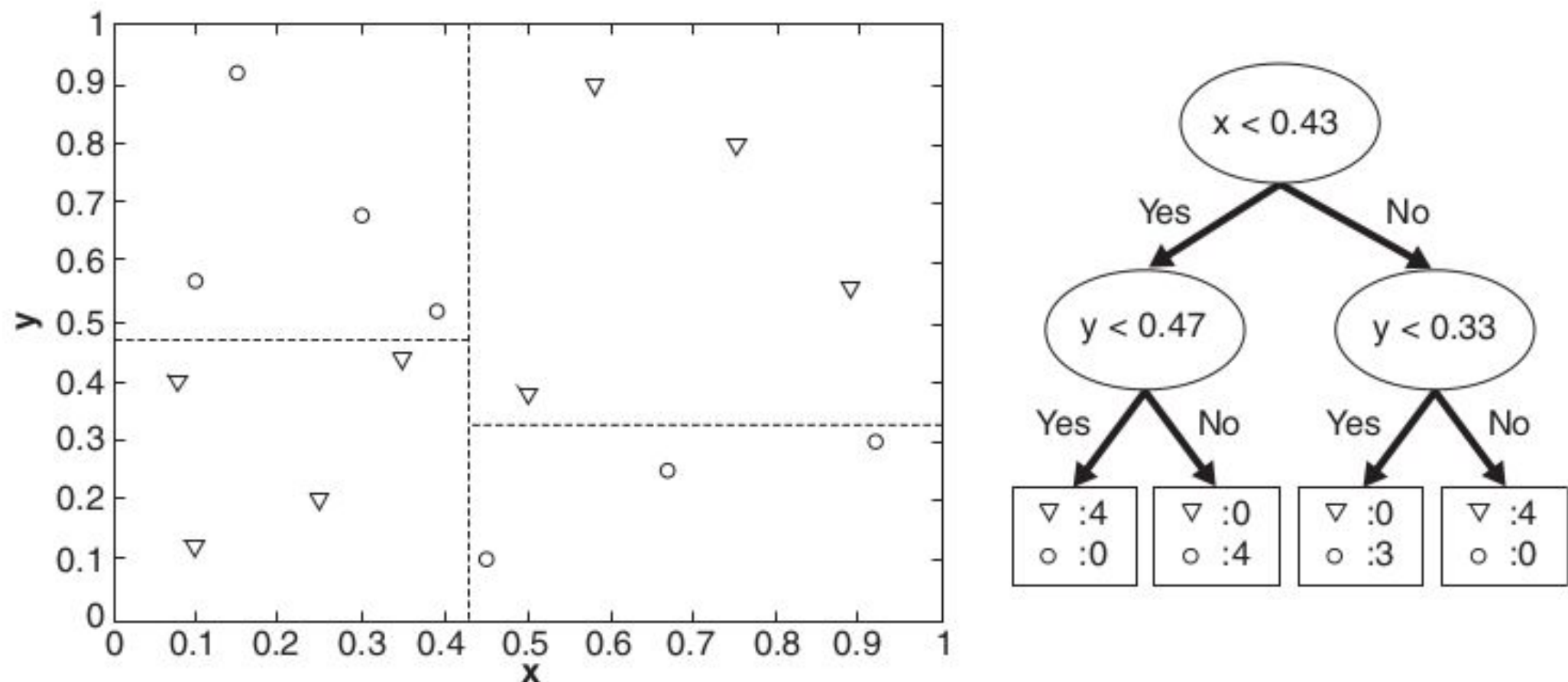
- Muchos algoritmos
  - CART
  - ID3, C4.5 (J48 en Weka)
  - SLIQ, SPRINT

# Construyendo un Árbol de Decisión

- Estrategia: Top down (greedy) - Divide y vencerás recursiva
  - Primero: seleccionar un atributo para el nodo raíz y crear rama para cada valor posible del atributo .
  - Luego: dividir las instancias del dataset en subconjuntos, uno para cada rama que se extiende desde el nodo.
  - Por último: repetir de forma recursiva para cada rama, utilizando sólo las instancias que llegan a ésta.
- Detenerse cuando todas las instancias del nodo sean de la misma clase.



# Un árbol de decisión hace cortes perpendiculares a los ejes



**Figure 3.20.** Example of a decision tree and its decision boundaries for a two-dimensional data set.

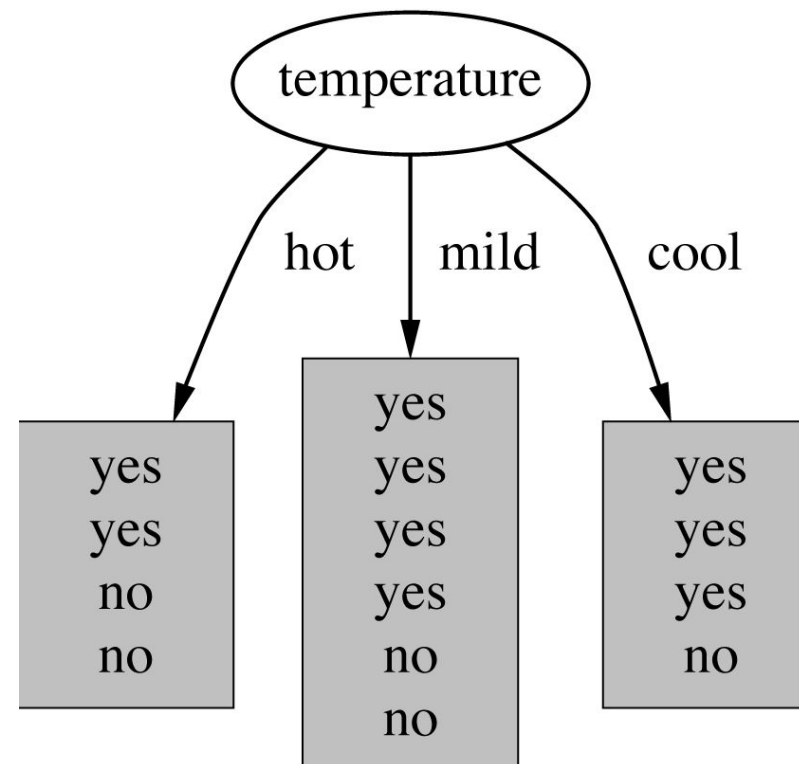
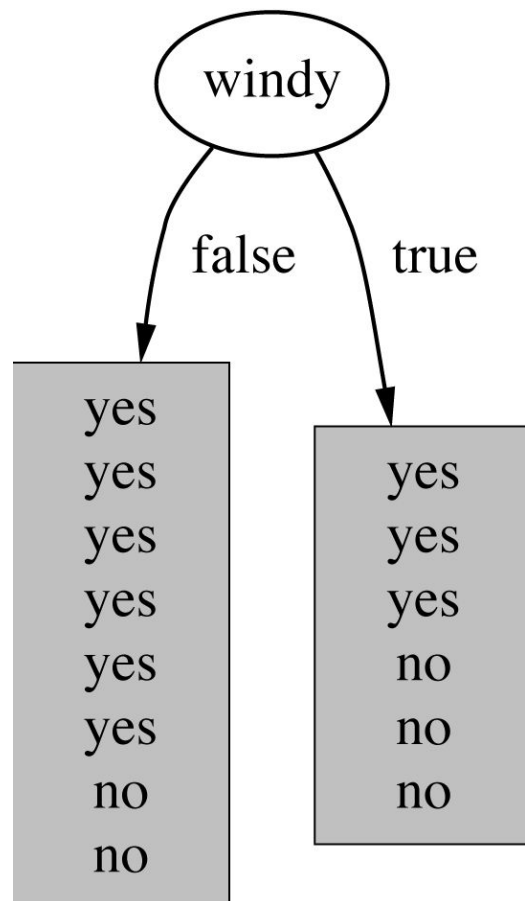
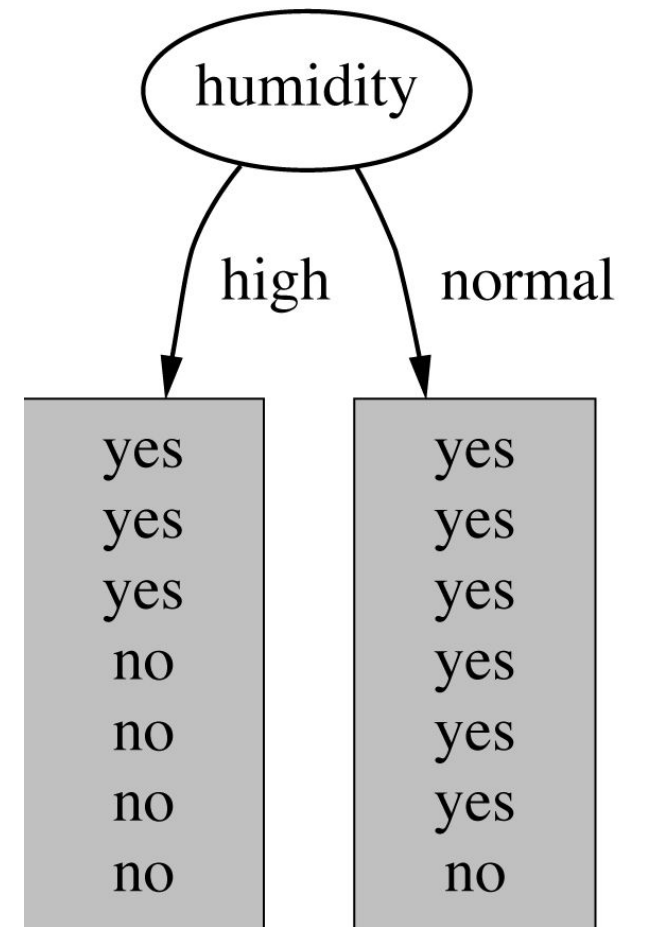
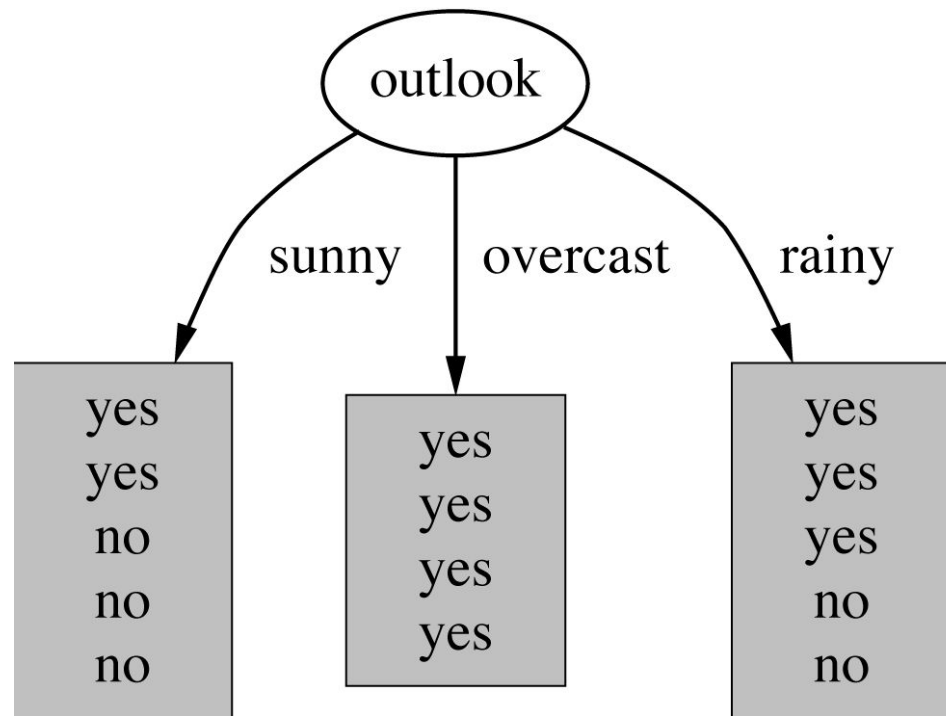
# El dataset Weather

Condiciones para salir a jugar tenis:

**Table 4.6**      **The weather data with identification codes.**

ID code	Outlook	Temperature	Humidity	Windy	Play
a	sunny	hot	high	false	no
b	sunny	hot	high	true	no
c	overcast	hot	high	false	yes
d	rainy	mild	high	false	yes
e	rainy	cool	normal	false	yes
f	rainy	cool	normal	true	no
g	overcast	cool	normal	true	yes
h	sunny	mild	high	false	no
i	sunny	cool	normal	false	yes
j	rainy	mild	normal	false	yes
k	sunny	mild	normal	true	yes
l	overcast	mild	high	true	yes
m	overcast	hot	normal	false	yes
n	rainy	mild	high	true	no

# ¿Cómo escoger atributos?



# Criterio para escoger el mejor atributo

- ¿Qué atributo escojo?
  - La idea es crear el árbol más pequeño posible.
  - Heurística: escoge el atributo que produce nodos lo más “puros” posible.
- El criterio más popular de pureza:  
**information gain**
  - Information gain crece cuando crece la pureza promedio de los subconjuntos.
- Estrategia: escoger el atributo que maximiza el valor de information gain.

# Computando la Información

La información se puede medir en **bits**

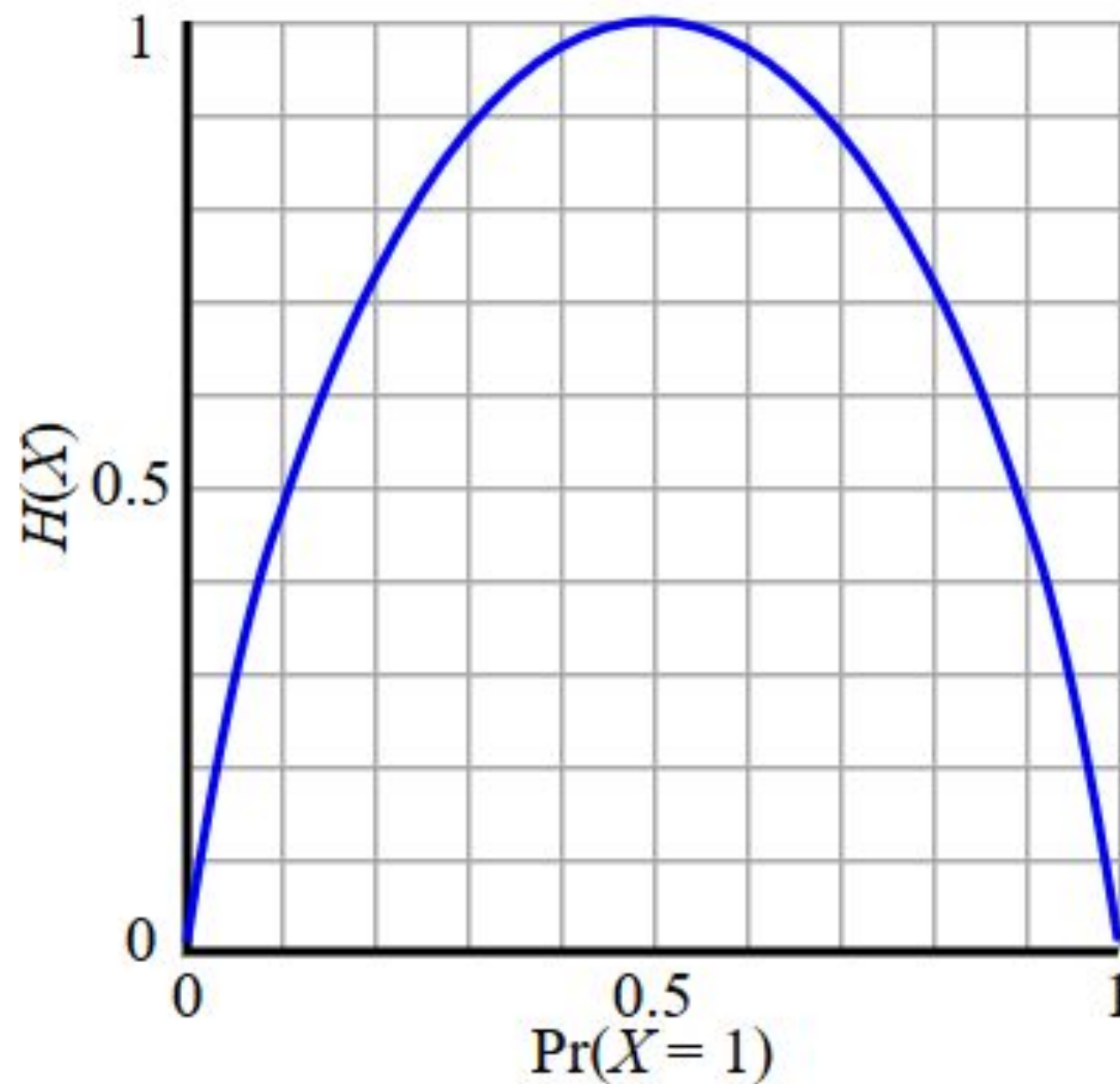
- **Entropía:** información promedio requerida para codificar un evento dado una distribución de probabilidad (viene de la teoría de información de Claude Shannon).
- La entropía nos entrega la información esperada en bits (puede ser una fracción).

Fórmula para calcular la entropía:

$$\text{entropy}(p_1, p_2, \dots, p_n) = -p_1 \log_2 p_1 - p_2 \log_2 p_2 \dots - p_n \log_2 p_n$$

$$H(X) = - \sum_{i=1}^n P(x_i) \log_b P(x_i)$$

# Entropía para dos Clases con distintas Proporciones



La entropía toma su máximo valor cuando  $p=0.5$  (máxima incerteza).

# Ejemplo: atributo *outlook*

- *Outlook = Sunny* :

$$\text{info}([2,3]) = \text{entropy}(2/5,3/5) = -2/5 \log(2/5) - 3/5 \log(3/5) = 0.971 \text{ bits}$$

- *Outlook = Overcast* :

$$\text{info}([4,0]) = \text{entropy}(1,0) = -1 \log(1) - 0 \log(0) = 0 \text{ bits}$$

***Nota: esto normalmente queda indefinido***

- *Outlook = Rainy* :

$$\text{info}([2,3]) = \text{entropy}(3/5,2/5) = -3/5 \log(3/5) - 2/5 \log(2/5) = 0.971 \text{ bits}$$

- Información esperada para el atributo

$$\text{info}([3,2],[4,0],[3,2]) = (5/14) \times 0.971 + (4/14) \times 0 + (5/14) \times 0.971 = 0.693 \text{ bits}$$

# Calculando information gain

- Information gain: Información antes del split – información después del split

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

$$\begin{aligned} \text{gain}(\text{Outlook}) &= \text{info}([9,5]) - \text{info}([2,3],[4,0],[3,2]) \\ &= 0.940 - 0.693 \\ &= 0.247 \text{ bits} \end{aligned}$$

- Information gain para los atributos de los datos de weather:

$$\text{gain}(\text{Outlook}) = 0.247 \text{ bits}$$

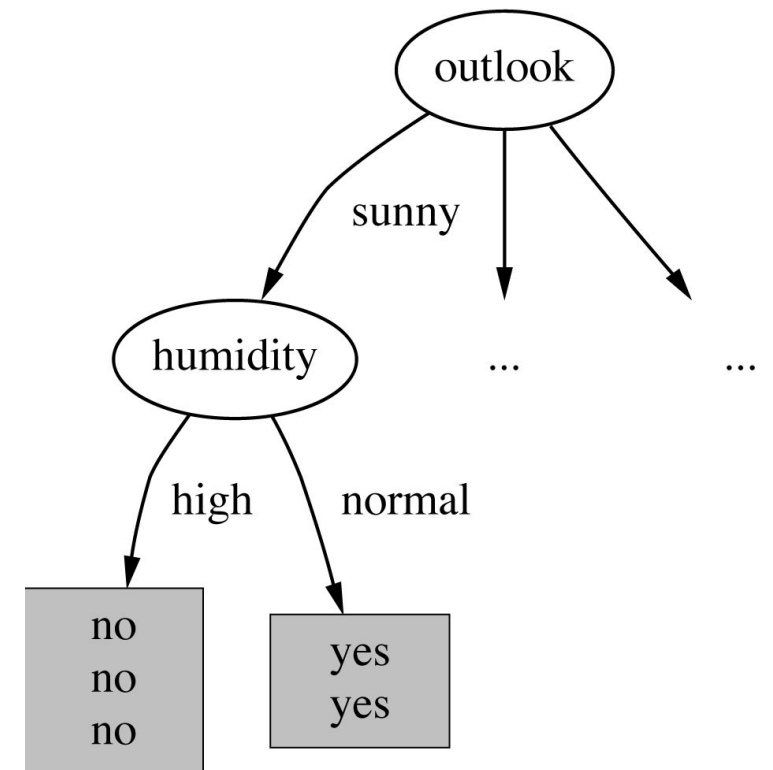
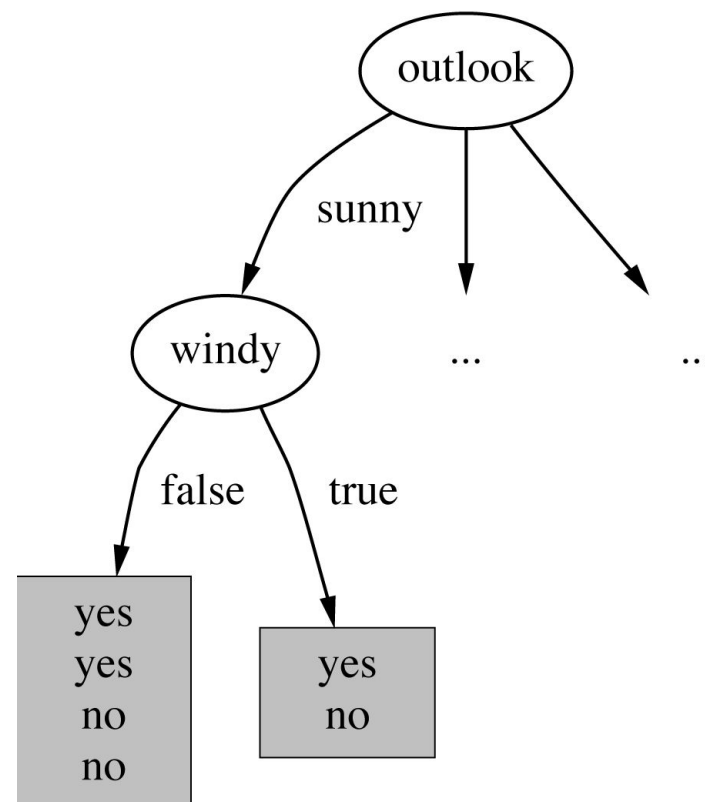
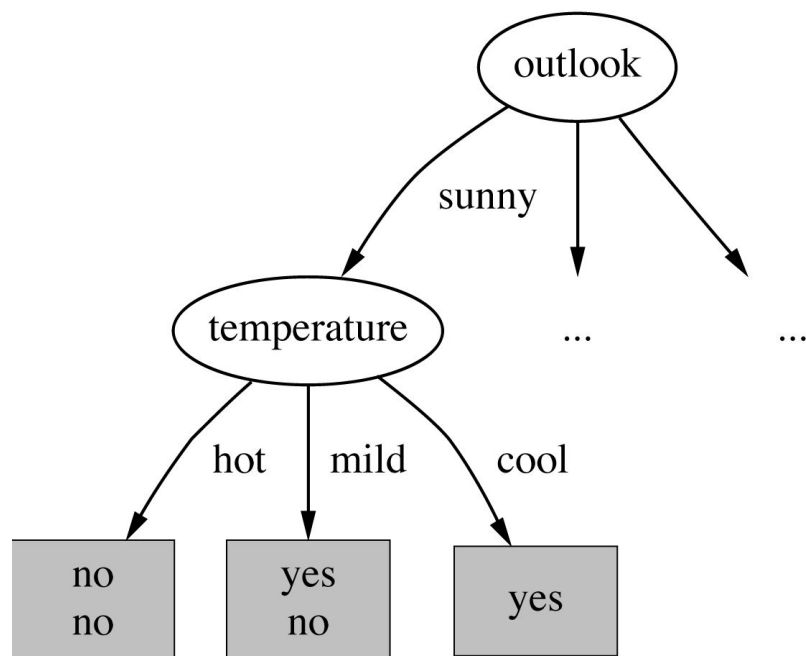
$$\text{gain}(\text{Temperature}) = 0.029 \text{ bits}$$

$$\text{gain}(\text{Humidity}) = 0.152 \text{ bits}$$

$$\text{gain}(\text{Windy}) = 0.048 \text{ bits}$$



# Seguimos particionando

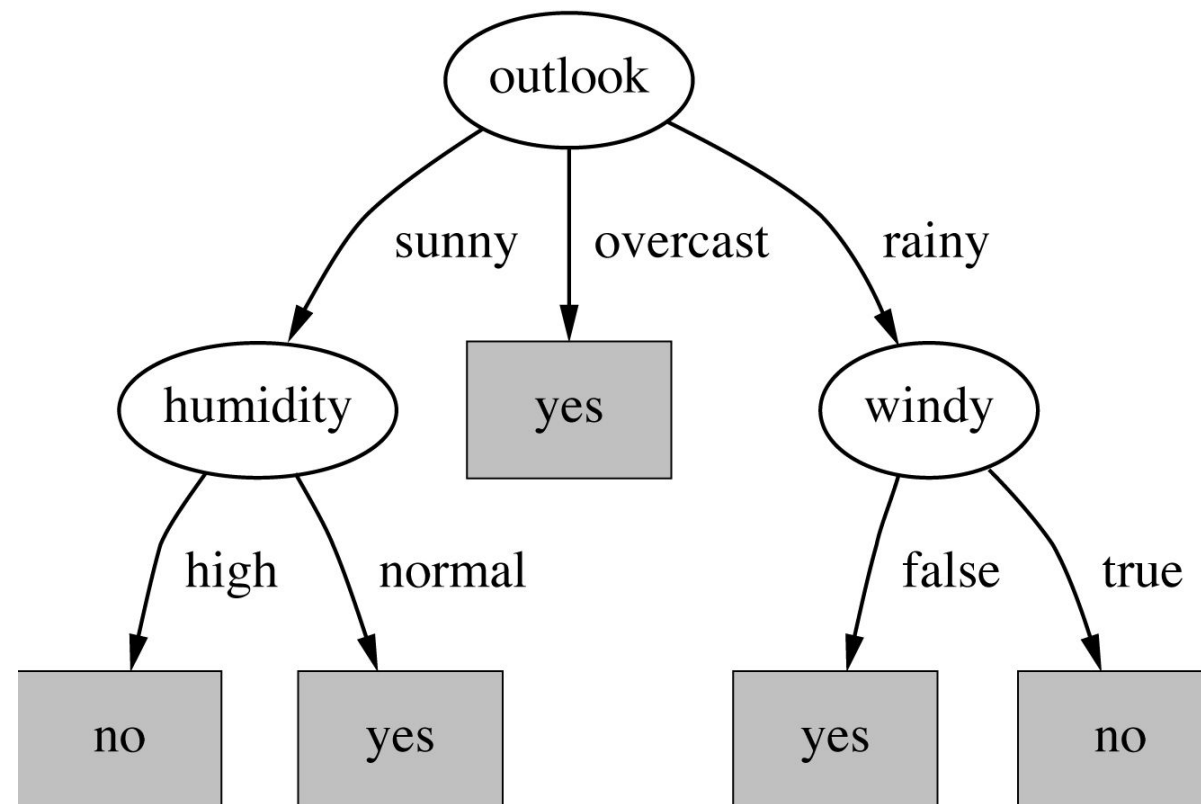


$\text{gain}(\text{Temperature}) = 0.571$  bits

$\text{gain}(\text{Humidity}) = 0.971$  bits

$\text{gain}(\text{Windy}) = 0.020$  bits

# Árbol de Decisión Resultante



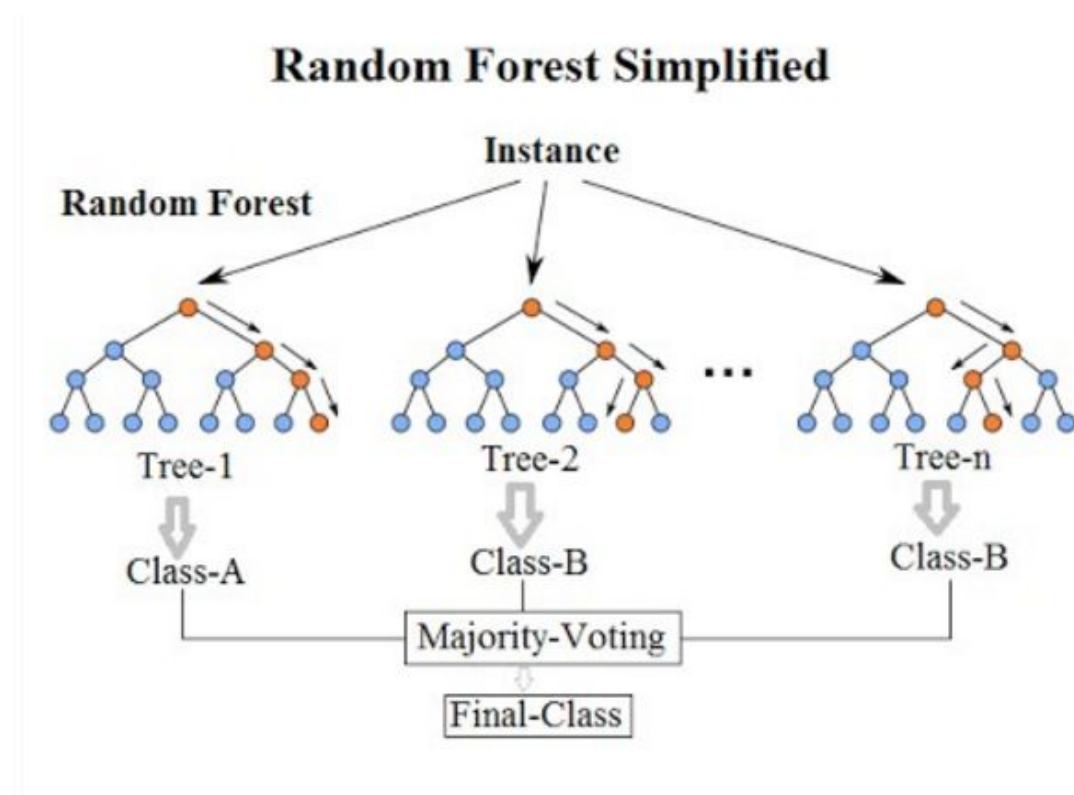
- Nota: no todas las hojas tienen que ser puras; a veces instancias idénticas tienen clases diferentes.
  - → El splitting termina cuando los datos no se pueden seguir particionando.
- Se puede exigir un mínimo número de instancias en la hoja para evitar sobreajuste.
- Puede predecir probabilidades usando las frecuencias relativas de las clases en la hoja.

# Otras cosas sobre árboles

- Information gain tiende a favorecer atributos de muchas categorías por su capacidad de fragmentar el dataset en muchas bifurcaciones.
- Una solución es usar una métrica llamada **Gain ratio**.
- **Gain ratio** toma en cuenta el número y el tamaño de las ramas (respecto a la cantidad de ejemplos que alcanzan) al elegir un atributo.
- Los atributos numéricos son discretizados, escogiendo la partición que maximice information gain (o gain ratio).
- Existen otras métricas para medir pureza distintas a entropía como el índice de **Gini**  $= 1 - \text{Pr}(\text{Sacar dos ejemplos de la misma clase})$ .
- Para evitar sobre-ajuste los árboles pueden ser **podados** (se eliminan ramas que alcanzan muy pocos ejemplos).
- La gran ventaja de los árboles es la **interpretabilidad**.

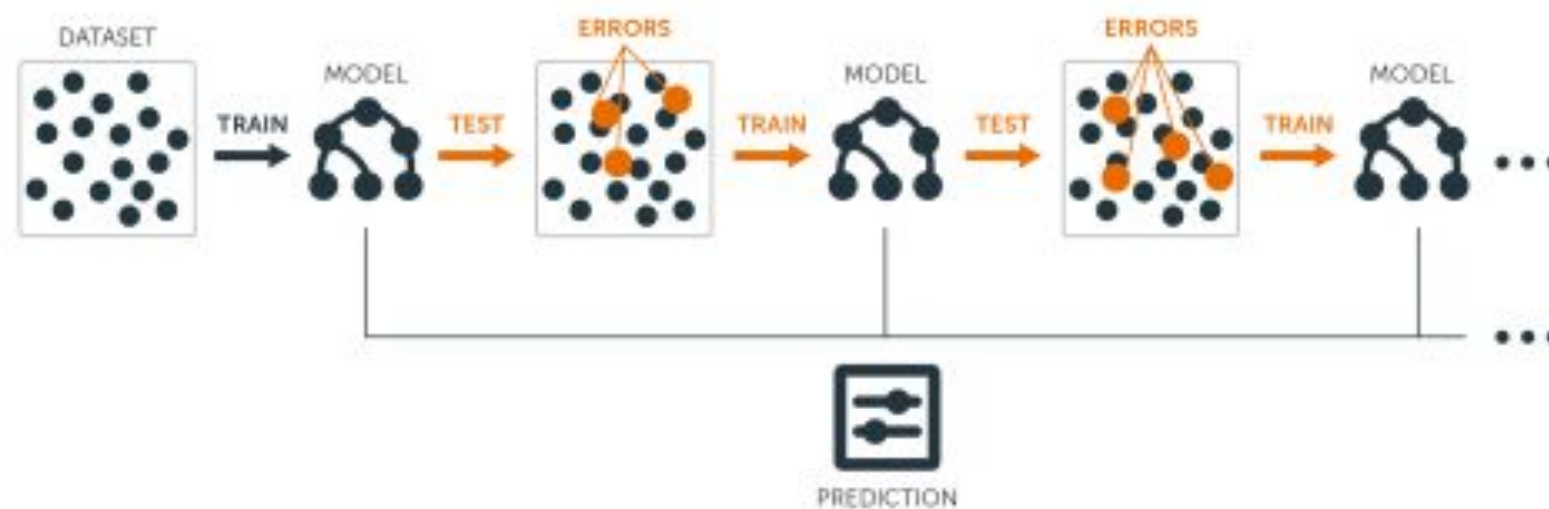
# Random Forest

- Es posible combinar varios árboles de decisión para tener modelos con mayor capacidad predictiva.
- Random Forest:
  - un **ensamble** de árboles de decisión (a veces de un sólo nivel).
  - Cada árbol se entrena con distintas muestras aleatorias (con reemplazo) del dataset de entrenamiento.
  - Luego los distintos árboles votan para hacer predicciones finales



# Gradient Boosting

- Boosting es otro enfoque para combinar varios clasificadores
- Idea: se entrenan modelos de manera secuencial donde cada modelo siguiente trata de corregir los errores del modelo anterior
- En Gradient Boosting los modelos son **árboles de decisión**.
- Cada árbol se entrena para predecir los errores del anterior.
- La predicción final es una suma ponderada de todos los árboles.
- Existen implementaciones eficientes de la idea como XGBoost y CatBoost para atributos categóricos.

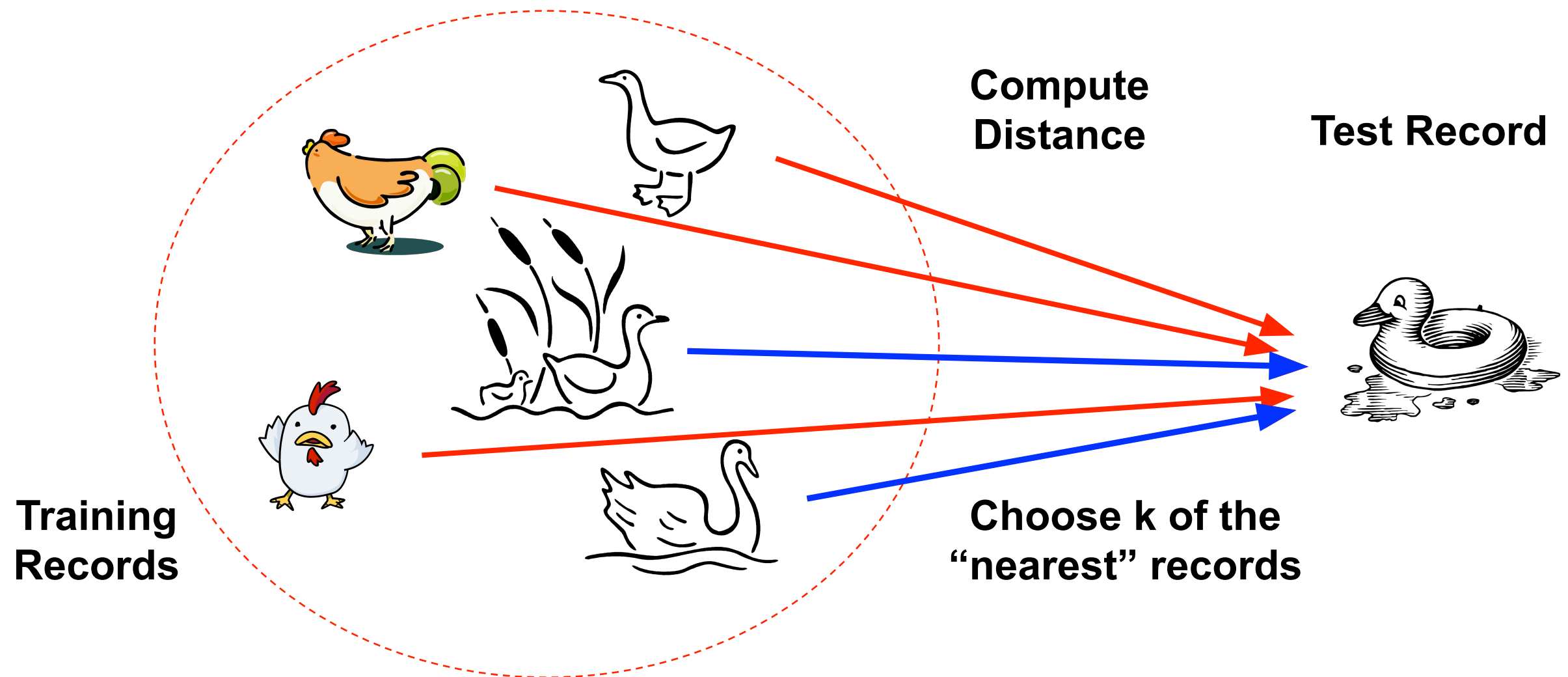


# Clasificador KNN

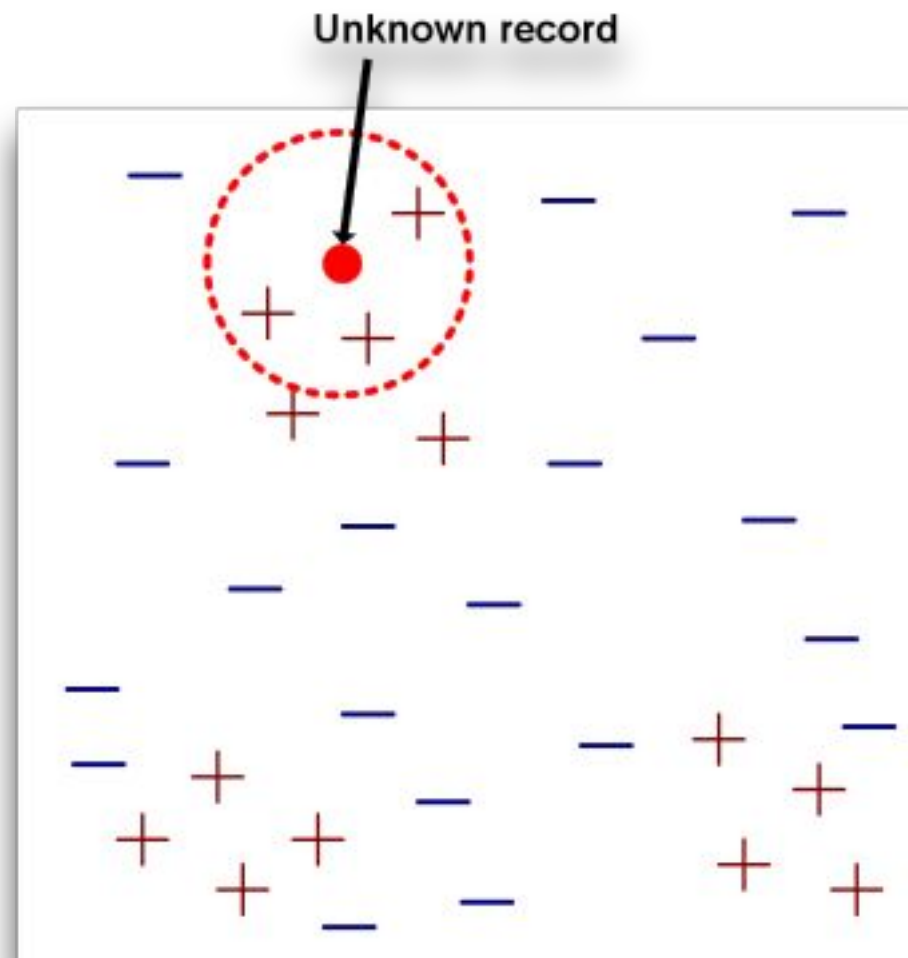
- Nearest Neighbor Classifier (o k-nn)
- Es un clasificador basado en **instancias**
- Conocido como **lazy**
  - Usa los **k** puntos más cercanos (nearest neighbors) para realizar la clasificación

# Clasificadores KNN

- Idea:
  - **If it walks like a duck, quacks like a duck, then it's probably a duck**



# Clasificadores KNN



- Necesita 3 cosas
  - Set de records almacenados.
  - Métrica de distancia para calcular la distancia entre records.
  - Valor de  $k$ , el número de vecinos cercanos a obtener.
- Para clasificar un récord nuevo
  - Calcular la distancia los los récords almacenados.
  - Identificar  $k$  nearest neighbors .
  - Utilizar la clase de los knn para asignar la clase al record nuevo (e.j. voto de la mayoría).



# Métricas de distancia

- Para atributos numéricos usamos la distancia euclidiana:

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{k=1}^n (x_k - y_k)^2},$$

- Una versión más general es la distancia de **Minkowsky**  
( $r=1 \Rightarrow$  distancia Manhattan,  $r=2 \Rightarrow$  distancia euclídeana)

$$d(\mathbf{x}, \mathbf{y}) = \left( \sum_{k=1}^n |x_k - y_k|^r \right)^{1/r}$$

- Es muy importante que los atributos estén normalizados.

# Escalando atributos

- Problemas de escalas

- Atributos deben ser escalados para prevenir que algún atributo domine la métrica de distancia

- Ejemplos:

- La altura de una persona puede variar entre 1.5m a 1.8m

- El peso puede variar entre 40 kg a 150 kg

- El ingreso de una persona puede variar entre \$150K a \$10M

# Técnicas para escalar atributos

- Normalización a media cero y varianza unitaria:

$$\frac{x - \mu_x}{\sigma_x}$$

- Normalización a rango entre 0 y 1:

$$\frac{x - \min_x}{\max_x - \min_x}$$

OJO: Apliquen la misma transformación a los datos de training y testing -> los valores de normalización se calculan sobre los datos de training.

# Distancia y similitud

- Es importante distinguir entre métricas de similitud y métricas de distancia.
- Similitud: entre más cerca dos objetos mayor el valor de la métrica.
- Distancia: entre más lejos dos objetos mayor el valor de la métrica.

Attribute Type	Dissimilarity	Similarity
Nominal	$d = \begin{cases} 0 & \text{if } x = y \\ 1 & \text{if } x \neq y \end{cases}$	$s = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{if } x \neq y \end{cases}$
Ordinal	$d =  x - y  / (n - 1)$ (values mapped to integers 0 to $n-1$ , where $n$ is the number of values)	$s = 1 - d$
Interval or Ratio	$d =  x - y $	$s = -d, s = \frac{1}{1+d}, s = e^{-d},$ $s = 1 - \frac{d - \min\_d}{\max\_d - \min\_d}$

# Similitud Coseno

- Cuando nuestros objetos son vectores sparse (muchas columnas con cero) es conveniente usar la similitud coseno.
- Corresponde al coseno del ángulo entre los dos vectores.

$$\cos(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\| \|\mathbf{y}\|} \quad \|\mathbf{x}\| = \sqrt{\sum_{k=1}^n x_k^2} = \sqrt{\mathbf{x} \cdot \mathbf{x}}.$$

- Un ejemplo común es cuando tratamos documentos como bolsas de palabras (cada columna es una palabra del vocabulario).

Ejemplo:

$$\mathbf{x} = (3, 2, 0, 5, 0, 0, 0, 2, 0, 0)$$

$$\mathbf{y} = (1, 0, 0, 0, 0, 0, 0, 1, 0, 2)$$

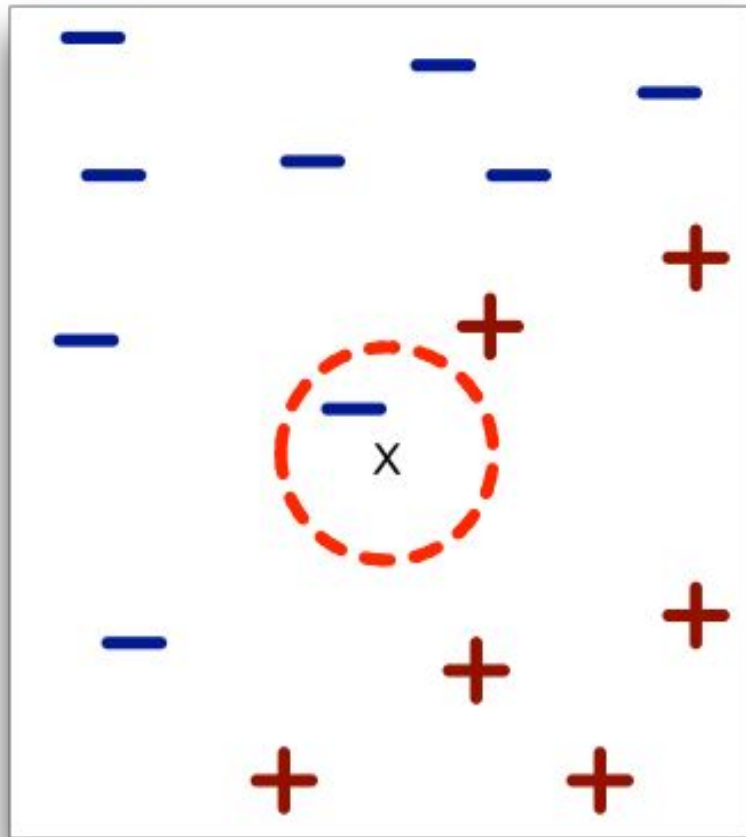
$$\mathbf{x} \cdot \mathbf{y} = 3 * 1 + 2 * 0 + 0 * 0 + 5 * 0 + 0 * 0 + 0 * 0 + 0 * 0 + 2 * 1 + 0 * 0 + 0 * 2 = 5$$

$$\|\mathbf{x}\| = \sqrt{3 * 3 + 2 * 2 + 0 * 0 + 5 * 5 + 0 * 0 + 0 * 0 + 0 * 0 + 2 * 2 + 0 * 0 + 0 * 0} = 6.48$$

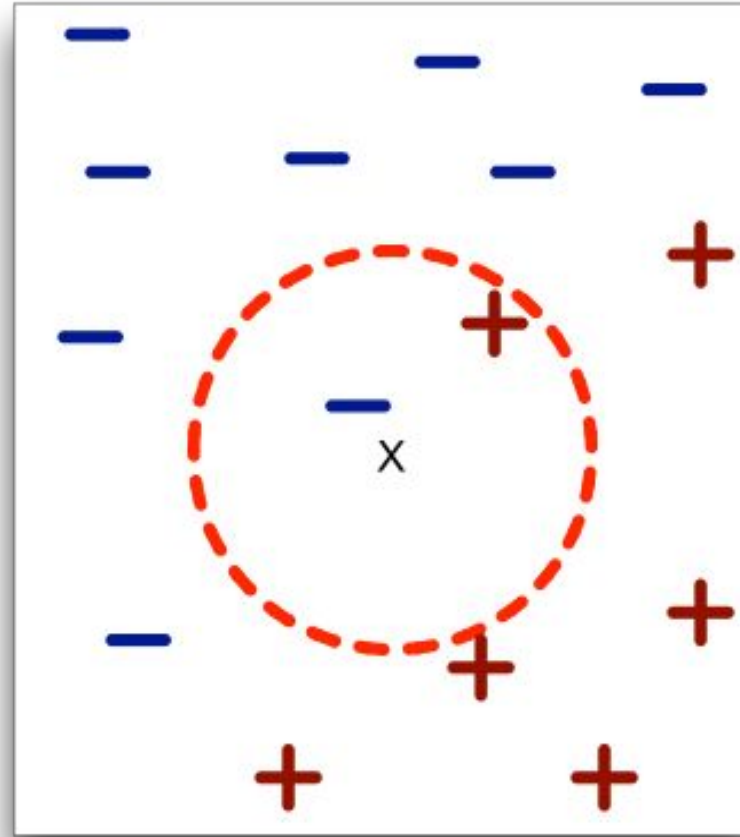
$$\|\mathbf{y}\| = \sqrt{1 * 1 + 0 * 0 + 0 * 0 + 0 * 0 + 0 * 0 + 0 * 0 + 0 * 0 + 1 * 1 + 0 * 0 + 2 * 2} = 2.24$$

$$\cos(\mathbf{x}, \mathbf{y}) = \mathbf{0.31}$$

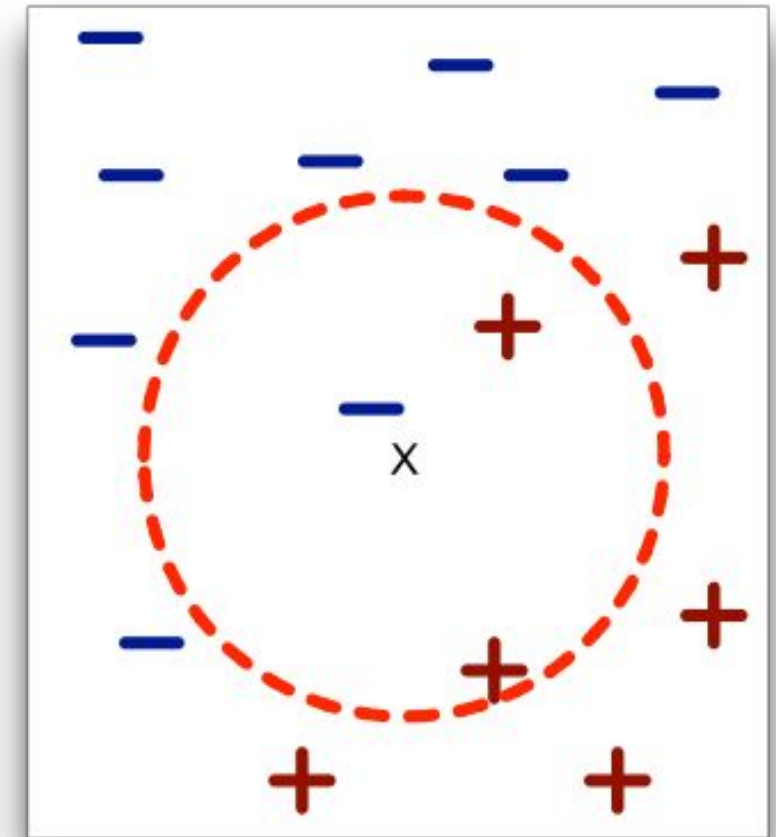
# Definición de NN



(a) 1-nearest neighbor



(b) 2-nearest neighbor

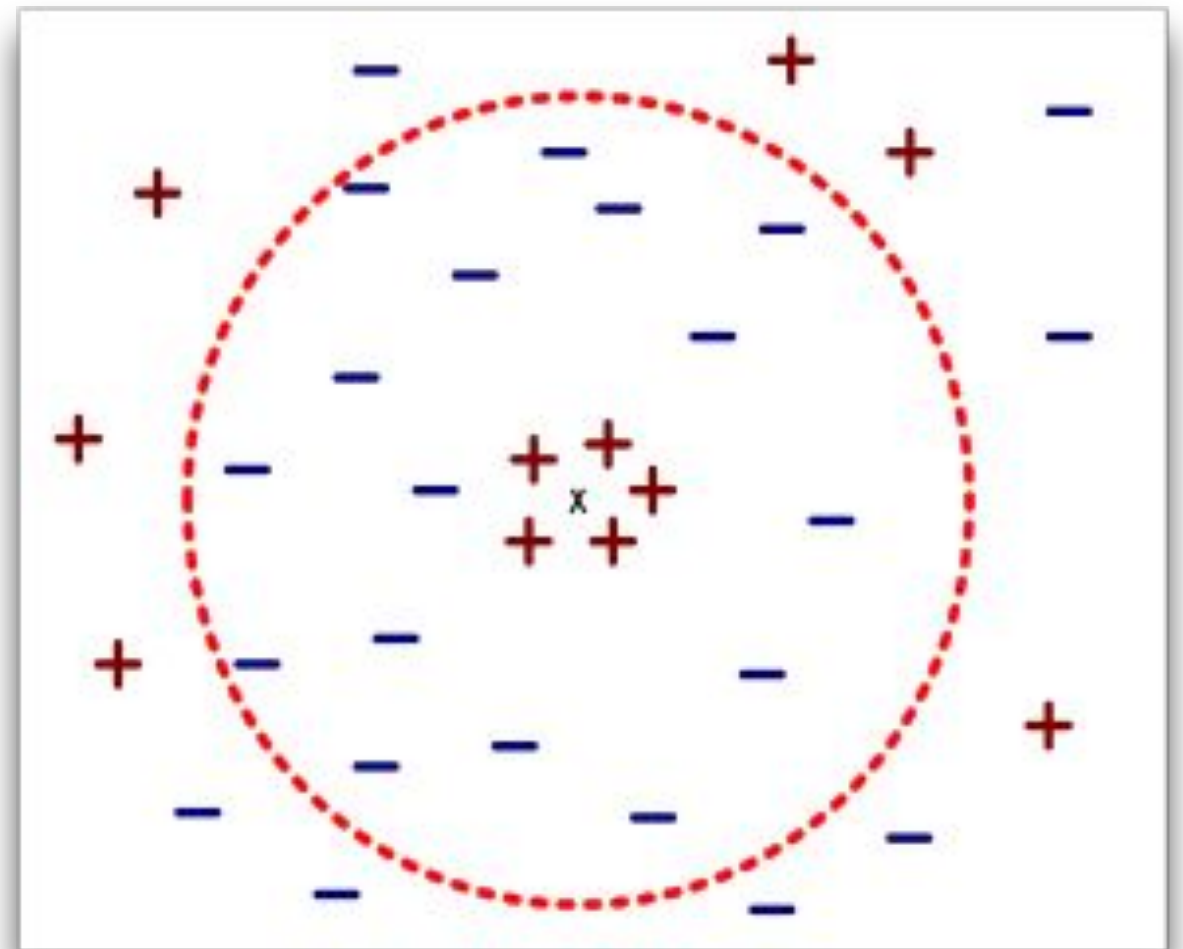


(c) 3-nearest neighbor

K-NN de un record  $x$  son los puntos que tienen las  $k$  menores distancias a  $x$

# Eligiendo el valor de K

- k muy pequeño es susceptible a ruido
- k muy grande puede incluir puntos de otra clase



# Clasificación kNN

- Los clasificadores k-NN son **lazy learners**.
- No construyen modelos explícitos, es más flexible ya que no necesita comprometerse con un modelo global a priori.
- Al contrario de otros **eager learners** como los árboles de decisión o clasificadores basados en reglas.
- Es independiente del nro. de clases.
- La clasificación es más costosa (memoria y tiempo).



# La Maldición de la Dimensionalidad

- Cuando los datos tienen una alta dimensionalidad KNN está sujeta a la **Maldición de la Dimensionalidad**.
- Fenómeno en que muchos tipos de análisis de datos se vuelven significativamente más difíciles a medida que aumenta la dimensionalidad de los datos.
- Para la clasificación, esto puede significar que no haya suficientes ejemplos para crear un modelo que asigne de forma confiable una clase a todos los ejemplos posibles.
- Para técnicas basadas en distancias (KNN, K-means) las distancias entre objetos se vuelven menos claras cuando hay muchas dimensiones.

# Clasificación basada en Naïve Bayes

- Familia de modelos basados en Bayes.
- Veremos Clasificador de Naive Bayes.
- También existen las Redes Bayesianas.

# Clasificación Basada en Naïve Bayes

- Modelo que busca **modelar la relación probabilística** entre atributos y clase.
- Modelo generativo, asume una distribución conjunta entre  $X$  e  $Y$ .
- Supuesto: atributos independientes dado la clase (**naïve assumption**).

# Clasificador Bayesiano

- Esquema probabilístico para resolver problemas de clasificación.

- Probabilidad condicional: 
$$P(C | A) = \frac{P(A, C)}{P(A)}$$

$$P(A | C) = \frac{P(A, C)}{P(C)}$$

- Teorema de Bayes: 
$$P(C | A) = \frac{P(A | C)P(C)}{P(A)}$$

# Ejemplo Teorema de Bayes

- Dado:
  - Un doctor sabe que la meningitis produce rigidez de cuello el 50% de las veces.
  - La probabilidad previa de que cualquier paciente tenga meningitis es  $1/50,000$ .
  - La probabilidad previa de que cualquier paciente tenga rigidez en el cuello es de  $1/20$ .
- ¿Si un paciente tiene el cuello rígido, cuál es la probabilidad de que tenga meningitis?

$$P(M | S) = \frac{P(S | M)P(M)}{P(S)} = \frac{0.5 \times 1/50000}{1/20} = 0.0002$$

# Clasificador Naïve Bayes

- Considerar cada atributo como variable condicionalmente independiente de la clase (eso es “naive”).
- Dado un record con atributos  $(A_1, A_2, \dots, A_n)$ .
  - La meta es predecir la clase  $C$ .
  - Específicamente queremos encontrar el  $C$  que maximice  $P(C | A_1, A_2, \dots, A_n)$ .
- ¿Podemos estimar  $P(C | A_1, A_2, \dots, A_n)$  directamente de los datos?

# Clasificador Naïve Bayes

- Aproximación
- Computar la probabilidad posterior  $P(C \mid A_1, A_2, \dots, A_n)$  para todos los valores de  $C$  usando el Teorema de Bayes.

$$P(C \mid A_1 A_2 \dots A_n) = \frac{P(A_1 A_2 \dots A_n \mid C) P(C)}{P(A_1 A_2 \dots A_n)}$$

- Elegir un valor de  $C$  que maximice  $P(C \mid A_1, A_2, \dots, A_n)$ .
- Equivalente a elegir un valor de  $C$  que maximice  $P(A_1, A_2, \dots, A_n \mid C) P(C)$ .
- Esto es porque el numerador  $P(A_1 A_2 \dots A_n)$  es constante para todas las clases.

# Clasificador Naïve Bayes

- Asume independencia entre los atributos  $A_i$  cuando la clase está dada (independencia condicional):
  - $P(A_1, A_2, \dots, A_n | C) = P(A_1 | C) P(A_2 | C) \dots P(A_n | C)$ .
  - Se puede estimar  $P(A_i | C)$  para todos los  $A_i$  y  $C$ .
  - Un punto nuevo  $A$ , se clasifica como  $C_j$  si  $P(C_j) \prod P(A_i | C_j)$  es máxima (en comparación con otros valores de  $C$ ).



# ¿Cómo estimar probabilidades a partir de los datos?

<i>Tid</i>	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

- Clase:  $P(C_k) = \frac{\text{count}(C_k)}{N}$

- e.g.,  $P(\text{No}) = 7/10$ ,  
 $P(\text{Yes}) = 3/10$

- Para atributos discretos:

$$P(A_i = b|C_k) = \frac{\text{count}(A_{ik} = b)}{\text{count}(C_k)}$$

- donde  $\text{count}(A_{ik} = b)$  es el número de instancias que tiene el valor  $b$  para el atributo  $A_i$  y que pertenecen a la clase  $C_k$

- Ejemplos:

- $P(\text{Status} = \text{Married} | \text{No}) = 4/7$   
 $P(\text{Refund} = \text{Yes} | \text{Yes}) = 0$

# Laplace Smoothing

- $P(C \mid A_1, A_2, \dots, A_n)$  se puede ir a cero cuando  $|A_{ik}=b| = 0$ , osea cuando para alguna clase  $C_k$  no hay ningún ejemplo con  $A_i=b$ .
- En ese caso Naive Bayes le asignaría probabilidad cero a la clase  $C_k$  a cualquier ejemplo con  $|A_{ik}=b| = 0$ , ignorando el valor de los otros atributos (acuérdense que las probabilidades se multiplican).
- Eso no es bueno para la generalización del modelo.
- Laplace Smoothing: soluciona el problema sumándole 1 a todos los conteos para que ninguna probabilidad quede en cero:

$$P(A_i = b|C_k) = \frac{\text{count}(A_{ik} = b) + 1}{\text{count}(C_K) + \text{values}(A_i)}$$

- Donde  $\text{values}(A_i)$  es la cantidad de categorías del atributo  $A_i$ .
- Con Laplace smoothing  $P(\text{Status} = \text{Married}|\text{No}) = (4+1)/(7+3)$

# Atributos Numéricos

- ¿Cómo calculamos  $P(A_{ik}=b|C_k)$  cuando el atributo  $A_i$  es numérico (ej: Taxable income) ?
- Una opción es discretizar el atributo y proceder de la forma anterior.
- Otra solución es asumir que el atributo sigue una distribución Gaussiana y estimar los parámetros de la función de densidad:

$$P(A_i = b|C_k) = \frac{1}{\sqrt{2\pi}\sigma_{ik}} \exp -\frac{(b-\mu_{ik})^2}{2\sigma_{ik}^2}$$

- Aquí  $\mu_{ik}$  y  $\sigma_{ik}$  se estiman como la media muestral y la desviación estándar de los ejemplos del atributo  $A_i$  cuando la clase es  $C_k$
- Sea  $A_i$  = Taxable income y  $C_k$ =No,  $\mu_{ik} = \text{mean}(125,100,70,120,60,220,75) = 110$   
y  $\sigma_{ik} = \text{sd}(125,100,70,120,60,220,75) = 54.5$

$P(\text{Taxable Income} = 130 | \text{No}) = \text{dnorm}(x=130, \text{mean}=110, \text{sd}=54.5) = 0.006843379$

# Naïve Bayes (Resumen)

- Es robusto ante puntos de ruido aislados.
- Maneja valores faltantes ignorando la instancia durante los cálculos de estimación de probabilidades.
- Robusto a atributos irrelevantes (afectan de igual manera a todas las clases).
- El supuesto de independencia entre atributos puede no ser cierto en todos los casos.
- Las redes Bayesianas o los modelos gráficos dirigidos permiten hacer modelos probabilísticos con supuestos de independencia menos restrictivos.



**dcc**

CIENCIAS DE LA COMPUTACIÓN  
UNIVERSIDAD DE CHILE

[www.dcc.uchile.cl](http://www.dcc.uchile.cl)

f @ in  / DCCUCHILE