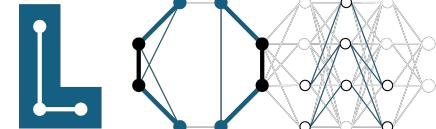


# Ciencia y Tecnología

Secretaría de Ciencia, Humanidades,  
Tecnología e Innovación



Laboratorio de Ciencia de Datos y Aprendizaje Automático

# Curso IA desde Cero

Dr. Irvin Hussein López Nava  
M.C. Joan M. Raygoza Romero

Departamento de Ciencias de la Computación  
Centro de Investigación Científica y de Educación Superior de Ensenada  
Edición 2025

CURSO VIRTUAL

**IA desde Cero:**  
Un Curso Práctico

EDUCACIÓN CONTINUA | FÍSICA APLICADA | CICESE



Del 3 de Nov  
al 1 de Dic

Dirigido al Público en general,  
que desee aprender técnicas de IA.

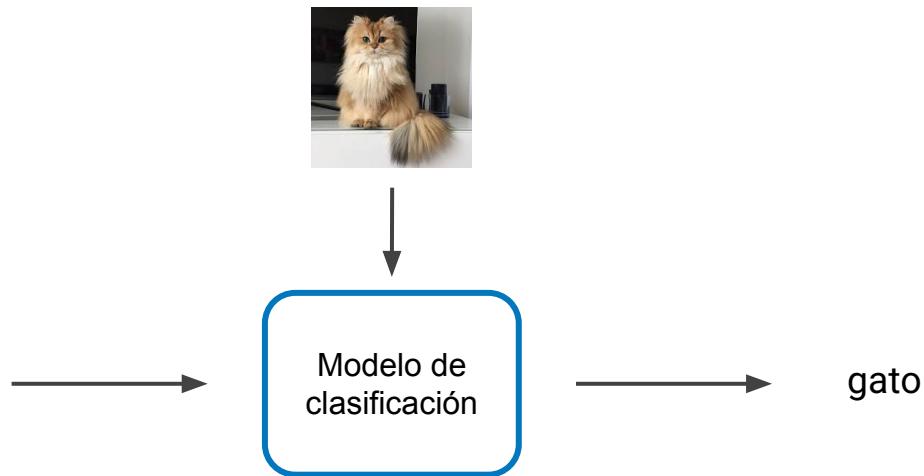


**Duración: 18 horas**

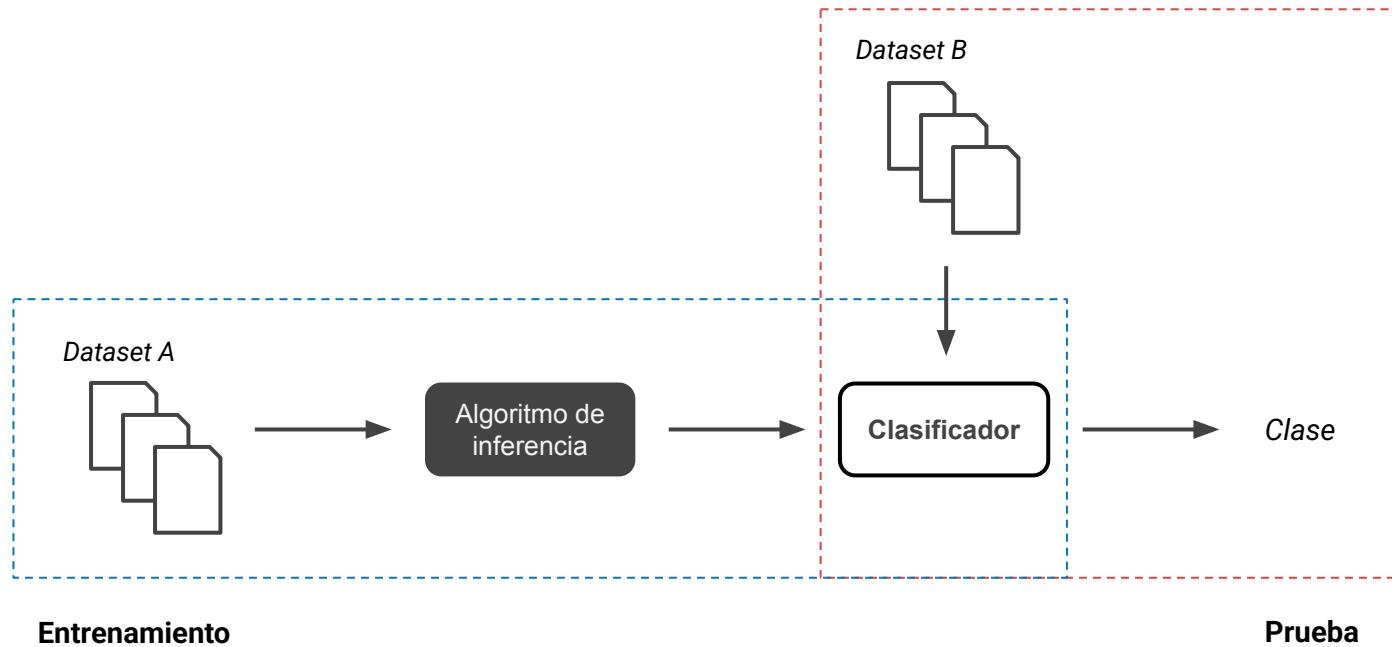
Horario: Lunes y miércoles,  
5:00 p.m. - 7:00 p.m.



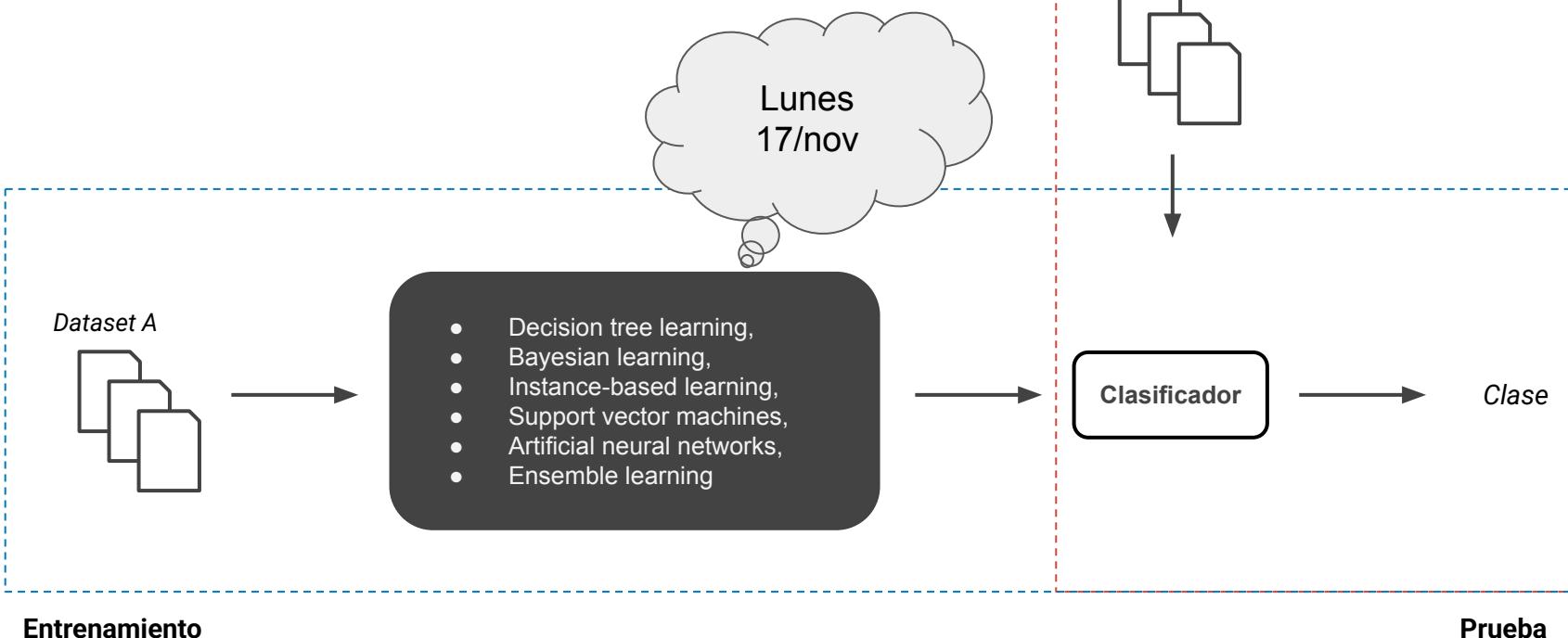
# Un primer caso



# Clasificador general



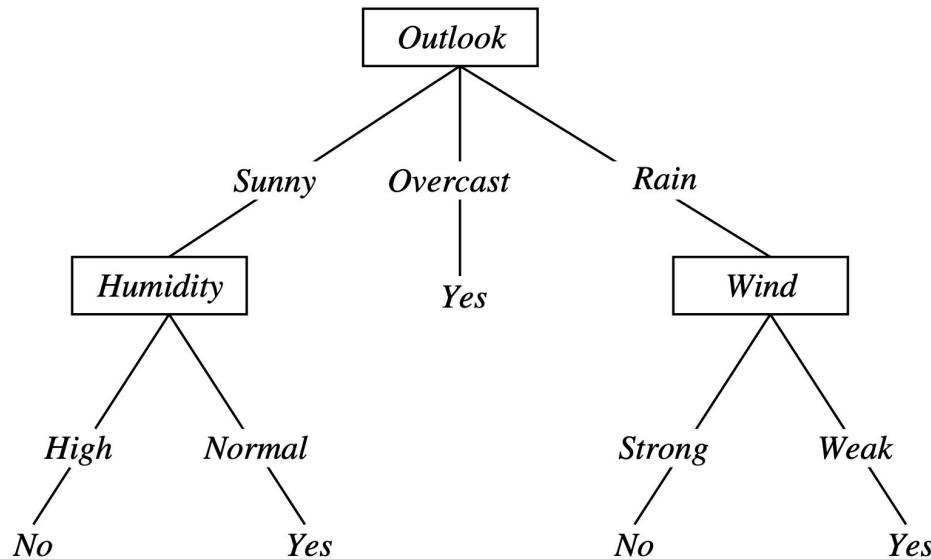
# Algoritmos de inferencia



# Algoritmos de inferencia



# Ejemplo: salir a jugar tenis



# *Decision tree*

- Un **árbol de decisión** representa una función que toma como entrada un vector de valores de atributos y devuelve una "decisión", i.e., un único valor de salida.
- Los valores de entrada y salida pueden ser **discretos** o **continuos**.
- Representación de un árbol de decisión:
  - Cada **nodo interno** prueba un atributo
  - Cada **rama** corresponde a un valor de atributo
  - Cada **nodo hoja** asigna una clasificación

# ¿Cuándo considerar los árboles de decisión?

- Instancias descriptibles por pares atributo-valor.
- La función objetivo es discreta.
- Puede ser necesaria una hipótesis disyuntiva.
- Datos de entrenamiento posiblemente ruidosos.

Ejemplos:

- Diagnóstico médico.
- Análisis de riesgo de crédito y detección de fraudes.
- Predicción de retención de empleados.
- Recomendación de productos.

# Árboles de decisión

- El algoritmo básico de aprendizaje de árboles de decisión, como el ID3 (Quinlan 1986) y su sucesor C4.5 (Quinlan 1993), utiliza una búsqueda codiciosa (*greedy*) de **arriba hacia abajo** a través del espacio de posibles árboles de decisión.
- Este enfoque comienza con la pregunta de qué atributo debe ser probado en la raíz del árbol.
  - Para decidir esto, cada atributo se evalúa mediante una prueba estadística para determinar qué tan bien clasifica los ejemplos de entrenamiento.

# Árboles de decisión

- El **mejor atributo** se selecciona para ser probado en la raíz del árbol.
- Después, se crea un **nodo descendiente** para cada valor posible de este atributo, y los ejemplos de entrenamiento se distribuyen al nodo correspondiente según su valor.
- Este proceso se repite **recursivamente** en cada **nodo descendiente**, seleccionando el **mejor atributo** en cada punto del árbol, lo que constituye una búsqueda *greedy* sin retroceso para construir un árbol de decisión aceptable.

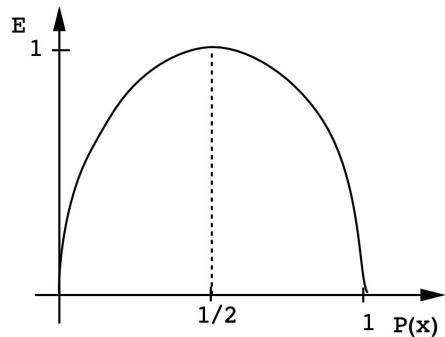
# ¿Cuál atributo es el mejor clasificador?

- Se desea elegir el atributo más útil para **clasificar** los ejemplos, ¿cuál es una buena medida cuantitativa del valor de un atributo?
- La propiedad estadística llamada **ganancia de información**, mide qué tan bien un atributo separa los ejemplos de entrenamiento según su clasificación objetivo.
  - ID3 utiliza esta medida para seleccionar entre los atributos candidatos en cada paso mientras construye el árbol.

# Entropía

- $S$  es una muestra de ejemplos de entrenamiento.
- $S_+$  es la proporción de ejemplos positivos en  $S$ .
- $S_-$  es la proporción de ejemplos negativos en  $S$ .
- La **entropía** mide la impureza de  $S$ .

$$Entropy(S) \equiv -S_+ \log_2 S_+ - S_- \log_2 S_-$$



# Aplicando la Entropía

- Para ilustrar esto, suponga que  $S$  es una colección de 14 ejemplos de algún concepto booleano, incluidos 9 ejemplos positivos y 5 negativos (adoptando la notación [9+, 5-] para resumir dicha muestra de datos).
- Entonces, la entropía de  $S$  relativa a esta clasificación booleana es

$$\text{Entropy}(S) \equiv \sum_{i=1}^c -p_i \log_2 p_i$$

$$\text{Entropy}([9+, 5-]) = -(9/14) \log_2(9/14) - (5/14) \log_2(5/14) = 0.940$$

# Ganancia de información

- $Gain(S, A)$  = reducción esperada de entropía debido a la clasificación en A causada por la partición de los ejemplos según este atributo.

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

donde  $Values(A)$  es el conjunto de todos los valores posibles para el atributo A, y  $S_v$  es el subconjunto de  $S$  para el que el atributo A tiene valor  $v$ .

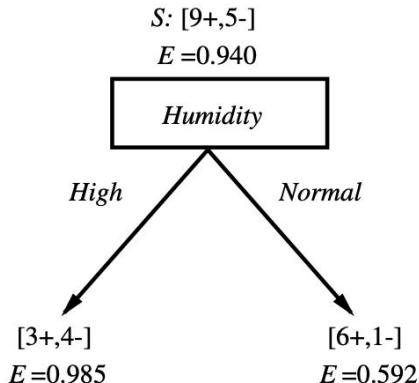
- Note que el primer término de la ecuación es la entropía de la colección original  $S$ , y el segundo término es el valor esperado de la entropía después de que  $S$  se particione utilizando el atributo  $A$ .

Attributes						✓ play
▲ day	▲ outlook	▲ temp	▲ humidity	▲ wind		✓ play
D1	Sunny	Hot	High	Weak	No	
D2	Sunny	Hot	High	Strong	No	
D3	Overcast	Hot	High	Weak	Yes	
D4	Rain	Mild	High	Weak	Yes	
D5	Rain	Cool	Normal	Weak	Yes	
D6	Rain	Cool	Normal	Strong	No	
D7	Overcast	Cool	Normal	Strong	Yes	
D8	Sunny	Mild	High	Weak	No	
D9	Sunny	Cool	Normal	Weak	Yes	
D10	Rain	Mild	Normal	Weak	Yes	
D11	Sunny	Mild	Normal	Strong	Yes	
D12	Overcast	Mild	High	Strong	Yes	
D13	Overcast	Hot	Normal	Weak	Yes	
D14	Rain	Mild	High	Strong	No	

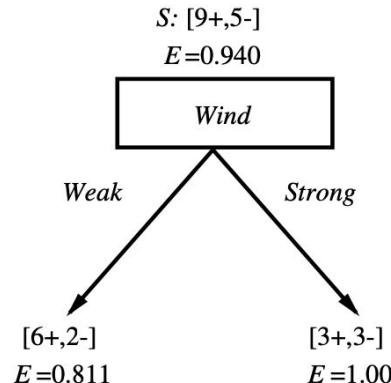
# Seleccionar el atributo siguiente

- ¿Cuál es el mejor atributo para clasificar?

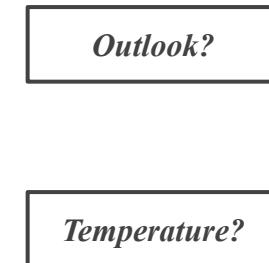
Calcular



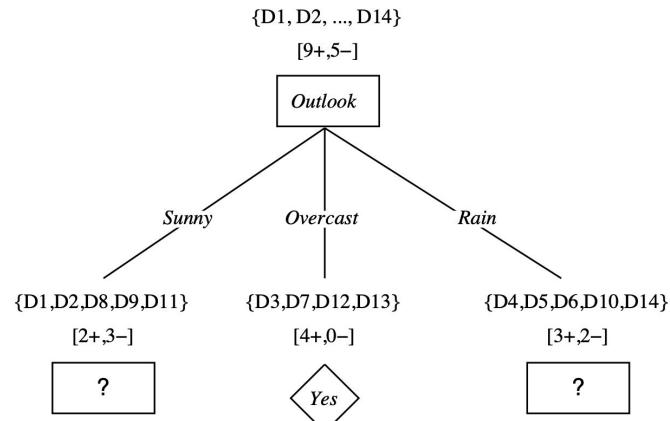
$$\begin{aligned} \text{Gain } (S, \text{ Humidity }) \\ &= .940 - (7/14).985 - (7/14).592 \\ &= .151 \end{aligned}$$



$$\begin{aligned} \text{Gain } (S, \text{ Wind }) \\ &= .940 - (8/14).811 - (6/14)1.0 \\ &= .048 \end{aligned}$$



# ¿y el que sigue?

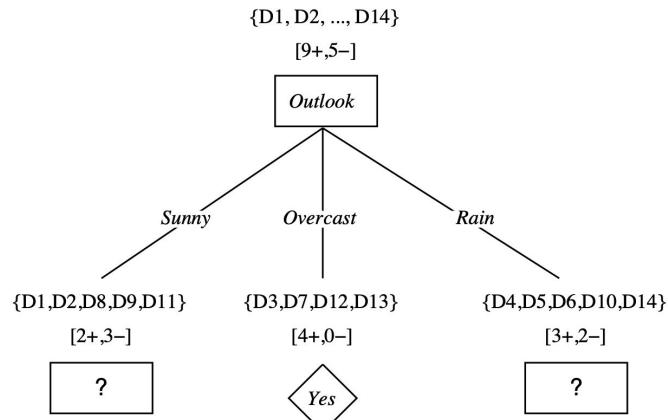


$$S_{Sunny} = \{D1, D2, D8, D9, D11\}$$

- $Gain(S_{Sunny}, Temperature) =$
- $Gain(S_{Sunny}, Humidity) =$
- $Gain(S_{Sunny}, Wind) =$

<u>A</u> day	<u>A</u> outlook	<u>A</u> temp	<u>A</u> humidity	<u>A</u> wind	<u>✓</u> play
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

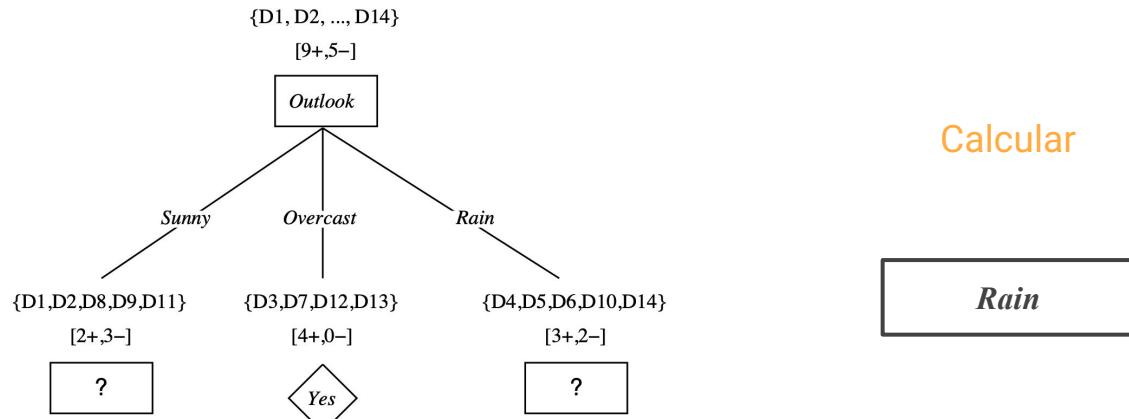
# ¿y el que sigue?



$$S_{Sunny} = \{D1, D2, D8, D9, D11\}$$

- $Gain(S_{Sunny}, Temperature) = 0.97 - (2/5) 0.0 - (2/5) 1.0 - (1/5) 0.0 = 0.57$
- $Gain(S_{Sunny}, Humidity) =$
- $Gain(S_{Sunny}, Wind) =$

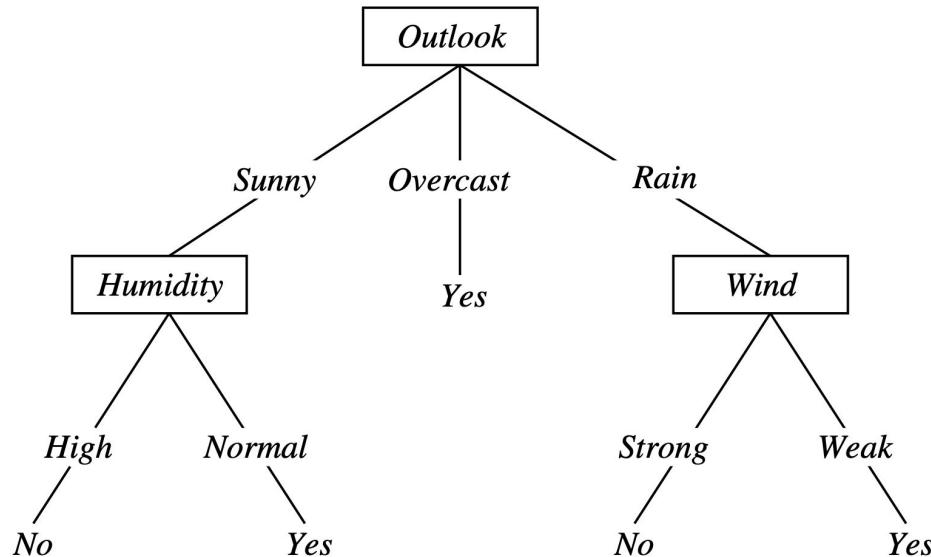
# ¿y el que sigue?



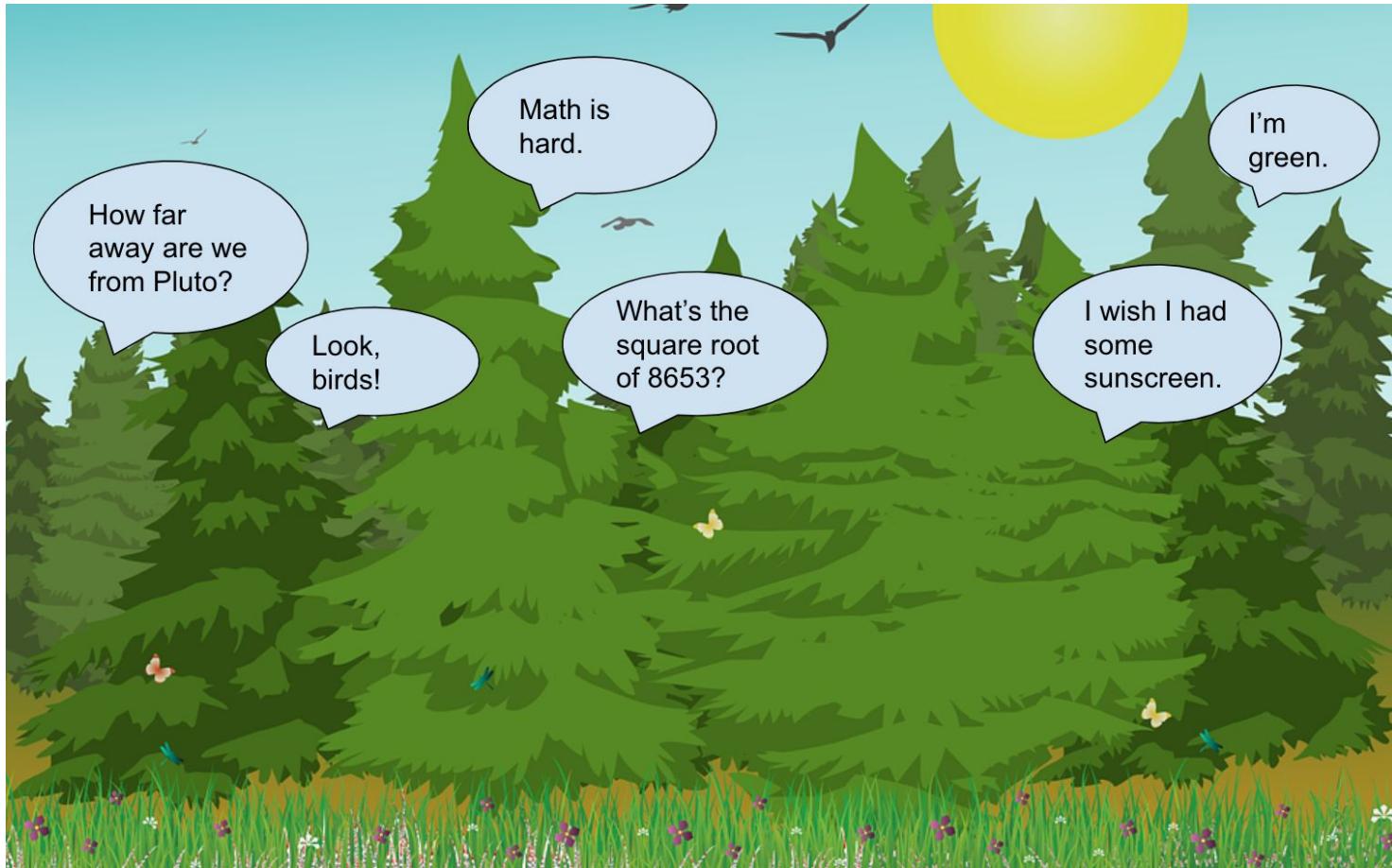
$$S_{Sunny} = \{D1, D2, D8, D9, D11\}$$

- $Gain(S_{Sunny}, \text{Temperature}) = 0.97 - (2/5) 0.0 - (2/5) 1.0 - (1/5) 0.0 = 0.57$
- $Gain(S_{Sunny}, \text{Humidity}) = 0.97 - (3/5) 0.0 - (2/5) 0.0 = 0.97$
- $Gain(S_{Sunny}, \text{Wind}) = 0.97 - (2/5) 1.0 - (3/5) 0.91 = 0.019$

# Ejemplo: salir a jugar tenis



# Just a random forest...



# Bosques aleatorios

- Los **árboles de decisión** son modelos base muy populares para los métodos de ensamble.
- Los 'aprendices fuertes' compuestos por varios árboles se denominan bosques (**forests**).
- Los árboles que componen un bosque pueden ser superficiales o *shallow* (poca profundidad), o profundos (muchas profundidad).
  - Los árboles poco profundos tienen menos varianza pero mayor sesgo, por lo que son la mejor opción para los **métodos secuenciales**.
  - Los árboles profundos tienen un sesgo bajo, pero una varianza alta, por lo que son una opción adecuada para *bagging*.

# Random Forest

- El enfoque de ‘bosques aleatorios’ es un método basado en *bagging* en el que se combinan árboles profundos, ajustados en muestras *bootstrap*, para producir un resultado con menor varianza.
- *Random forest* también utiliza otro truco para hacer que los múltiples árboles ajustados estén un poco menos correlacionados entre sí:
  - al hacer crecer cada árbol, en lugar de muestrear sólo sobre las observaciones para generar un *bootstrap*, **muestra sobre las características** y utiliza sólo un subconjunto aleatorio de ellas para construir el árbol.

# Random Forest

- El **muestreo sobre características** hace que todos los árboles no tengan exactamente la misma información y reduce la correlación entre los distintos resultados devueltos.
- Otra ventaja es que hace que el proceso de toma de decisiones sea más robusto frente a los datos que faltan.
- *Random forest* combina los conceptos de *bagging* y selección aleatoria de subespacios de características para crear modelos más robustos.



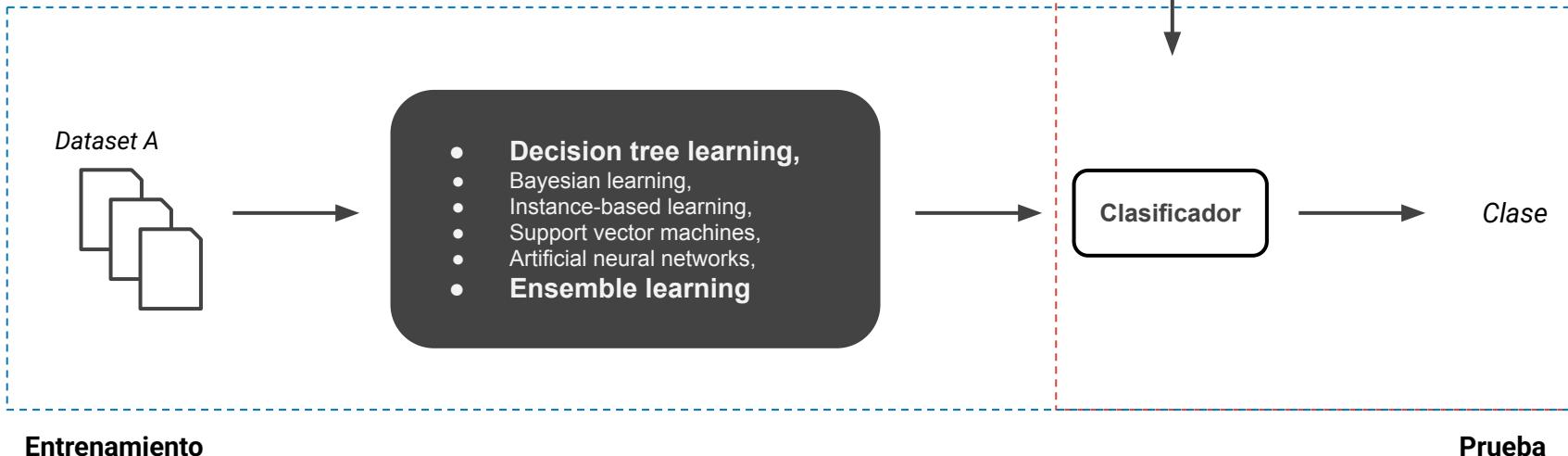
Dataset inicial

Muestras  
bootstrap + Características  
seleccionadas

Árboles profundos ajustados a  
cada muestra *bootstrap* y  
considerando sólo las  
características seleccionadas

*Random forest*  
(promedio de los árboles)

# Algoritmos de inferencia





## BAYES THEOREM

$$p(A|B) = \frac{\text{Posterior probability}}{\text{Likelihood} \times \text{Prior probability}}$$
$$p(A|B) = \frac{p(B|A) p(A)}{p(B)}$$



## BAE'S THEOREM

$$P(chill|Netflix) = \frac{P(Netflix|chill)P(chill)}{P(Netflix)}$$

# Teorema de Bayes

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

- $P(h)$  = probabilidad previa de la hipótesis  $h$ , i.e. *a priori*.
- $P(D)$  = probabilidad previa de los datos de entrenamiento  $D$ , i.e. evidencia.
- $P(h|D)$  = probabilidad de  $h$  dado  $D$ , i.e. *a posteriori*.
- $P(D|h)$  = probabilidad de  $D$  dado  $h$ , i.e. verosimilitud.

# Clasificador Naive Bayes

- Junto con los árboles de decisión, las redes neuronales y los k-vecinos más cercanos, es uno de los métodos de aprendizaje más prácticos.
- Cuándo usar:
  - Conjunto de entrenamiento moderado o grande.
  - Los atributos que describen las instancias son **condicionalmente independientes** dada una clasificación.
- Algunas aplicaciones exitosas:
  - Diagnóstico médico.
  - Clasificación de documentos de texto.

# Formulación

- Suponga funciones objetivo  $f: X \rightarrow V$ , donde cada instancia  $x$  se describe mediante los atributos  $\{a_1, a_2, \dots, a_n\}$ . El valor más probable de  $f(x)$  es:

$$\begin{aligned} v_{MAP} &= \arg \max_{v \in V} P(v_j | a_1, a_2, \dots, a_n) \\ v_{MAP} &= \arg \max_{v \in V} \frac{P(a_1, a_2, \dots, a_n | v_j) P(v_j)}{P(a_1, a_2, \dots, a_n)} \\ &= \arg \max_{v \in V} P(a_1, a_2, \dots, a_n | v_j) P(v_j) \end{aligned}$$

- Suposición ingenua de Bayes:  $P(a_1, a_2, \dots, a_n | v_j) = \prod_i P(a_i | v_j)$
- lo que da un clasificador Naive Bayes:  $v_{NB} = \arg \max_{v \in V} P(v_j) \prod_i P(a_i | v_j)$

# Algoritmo Naive Bayes

*Naive\_Bayes\_Learn(examples)*

For each target value  $v_j$

$$\hat{P}(v_j) \leftarrow \text{estimate } P(v_j)$$

For each attribute value  $a_i$  of each attribute  $a$

$$\hat{P}(a_i|v_j) \leftarrow \text{estimate } P(a_i|v_j)$$

*Classify\_New\_Instance(x)*

$$v_{NB} = \arg \max_{v_j \in V} \hat{P}(v_j) \prod_{a_i \in x} \hat{P}(a_i|v_j)$$

# Nuestro ejemplo

▲ day	▀	▲ outlook	▀	▲ temp	▀	▲ humidity	▀	▲ wind	▀	✓ play	▀
D1		Sunny		Hot		High		Weak		No	
D2		Sunny		Hot		High		Strong		No	
D3		Overcast		Hot		High		Weak		Yes	
D4		Rain		Mild		High		Weak		Yes	
D5		Rain		Cool		Normal		Weak		Yes	
D6		Rain		Cool		Normal		Strong		No	
D7		Overcast		Cool		Normal		Strong		Yes	
D8		Sunny		Mild		High		Weak		No	
D9		Sunny		Cool		Normal		Weak		Yes	
D10		Rain		Mild		Normal		Weak		Yes	
D11		Sunny		Mild		Normal		Strong		Yes	
D12		Overcast		Mild		High		Strong		Yes	
D13		Overcast		Hot		Normal		Weak		Yes	
D14		Rain		Mild		High		Strong		No	

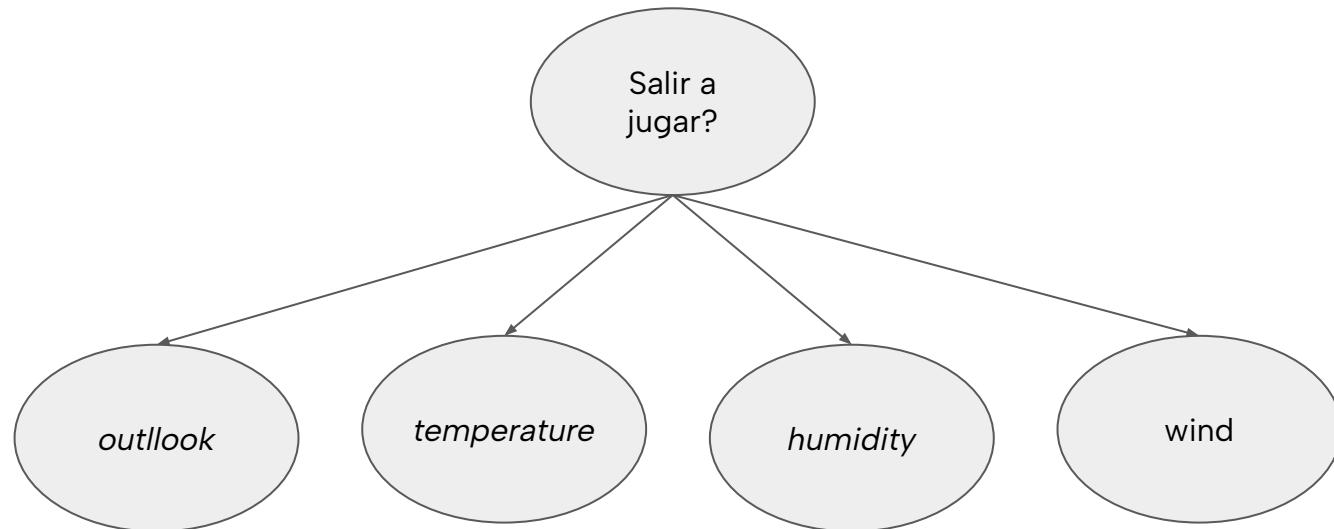
# Nuestro ejemplo con Naive Bayes

- Considere una nueva instancia:  
 $\{\text{outlook} = \text{sunny}, \text{temperature} = \text{cool}, \text{humidity} = \text{high}, \text{wind} = \text{strong}\}$
- Se quiere calcular:  $v_{NB} = \arg \max_{v_j \in V} P(v_j) \prod_i P(a_i | v_j)$
- $P(y) P(\text{sol} | y) P(\text{frío} | y) P(\text{alto} | y) P(\text{fuerte} | y) =$
- $P(n) P(\text{sol} | n) P(\text{frío} | n) P(\text{alto} | n) P(\text{fuerte} | n) =$
- ${}^vNB = ?$

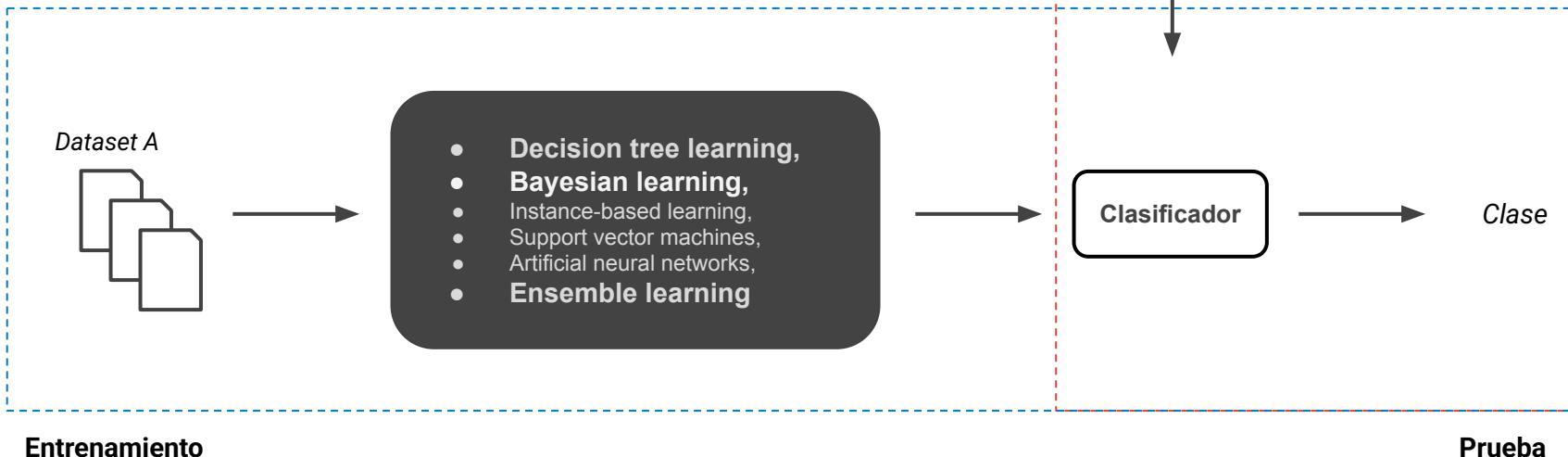
# Nuestro ejemplo con Naive Bayes

- Considere una nueva instancia:  
 $\{\text{outlook} = \text{sunny}, \text{temperature} = \text{cool}, \text{humidity} = \text{high}, \text{wind} = \text{strong}\}$
- Se quiere calcular:  $v_{NB} = \arg \max_{v_j \in V} P(v_j) \prod_i P(a_i | v_j)$
- $P(y) P(\text{sol} | y) P(\text{frío} | y) P(\text{alto} | y) P(\text{fuerte} | y) = 0,005$
- $P(n) P(\text{sol} | n) P(\text{frío} | n) P(\text{alto} | n) P(\text{fuerte} | n) = 0,021$
- ${}^vNB = n$

# Ejemplo: salir a jugar tenis



# Algoritmos de inferencia



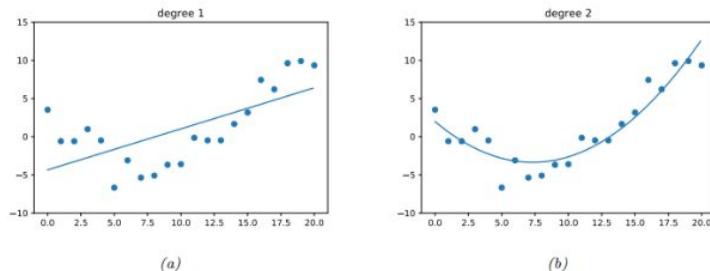
Dime con quien andas



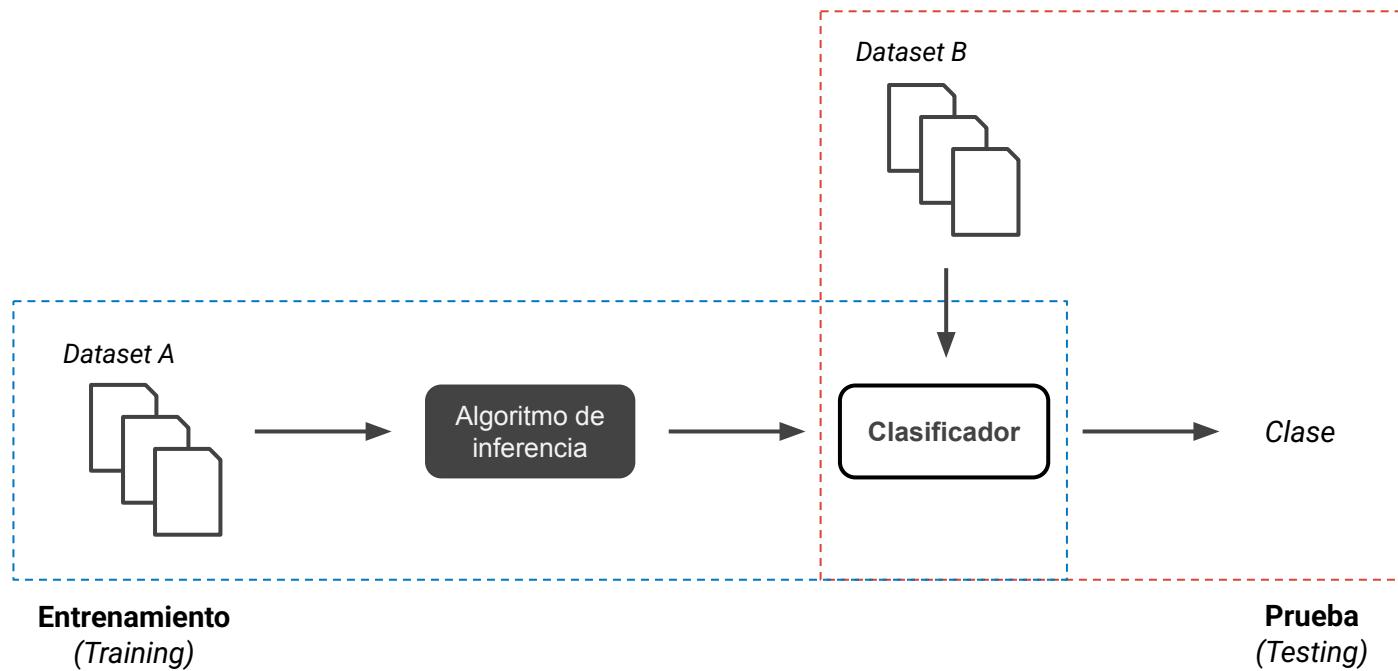
y te diré quién eres...

# Métodos paramétricos

- Construyen una descripción general explícita de la función objetivo  $f$ .
  - El modelo resume los datos a través de una colección de parámetros.
  - Número fijo de parámetros.
  - Independiente del número de instancias en los datos de entrenamiento.



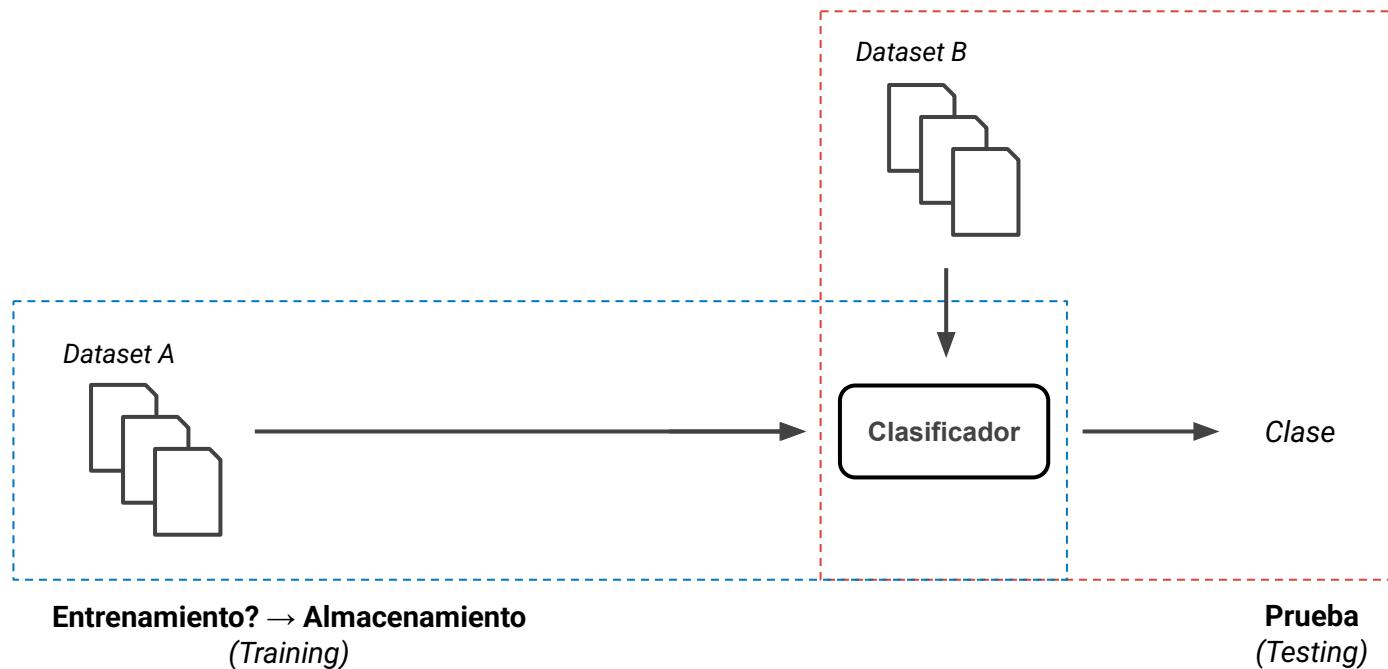
# Métodos paramétricos



# Métodos NO-paramétricos

- No hace suposiciones sobre la función objetivo.
  - Libertad para elegir cualquier función a partir de los datos de entrenamiento.
  - Requieren muchos más datos para estimar la función objetivo.
- Entradas similares tienen salidas similares.
  - Es una suposición razonable: El mundo es suave y las funciones, ya sean densidades, discriminantes o funciones de regresión, cambian lentamente.

# Métodos NO-paramétricos



# Métodos paramétricos vs no-paramétricos

- Los métodos **no paramétricos** también se denominan algoritmos de aprendizaje perezoso (*lazy*), porque a diferencia de los modelos paramétricos ansiosos (*eager*), no calculan un modelo cuando se les da el conjunto de entrenamiento, sino que posponen el cálculo hasta que se les da una instancia de consulta.

# Idea principal

Vecino más próximo (*nearest neighbor*):

- Dada la instancia de consulta  $x_q$ , primero se localiza el ejemplo de entrenamiento más cercano  $x_n$ , después se estima

$$\hat{f}(x_q) \leftarrow f(x_n)$$

k-vecinos más próximos (*k-nearest neighbor*):

- Dado  $x_q$ , se vota entre sus  $k$  vecinos más cercanos (si la función objetivo es de valor discreto), o
- Se toma la media de los valores  $f$  de los  $k$  vecinos más cercanos (si son de valor real).

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^k f(x_i)}{k}$$

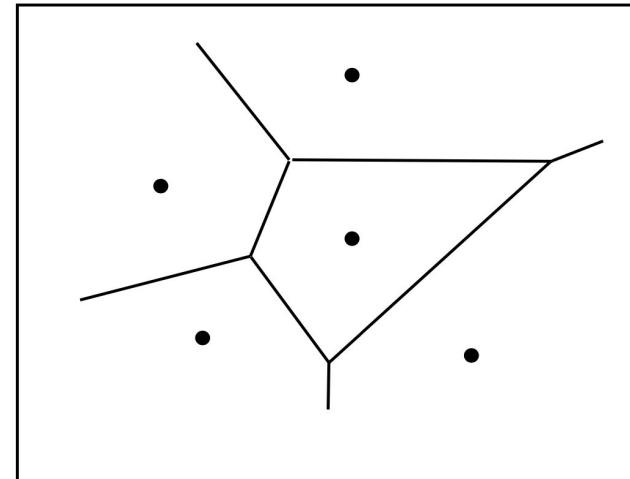
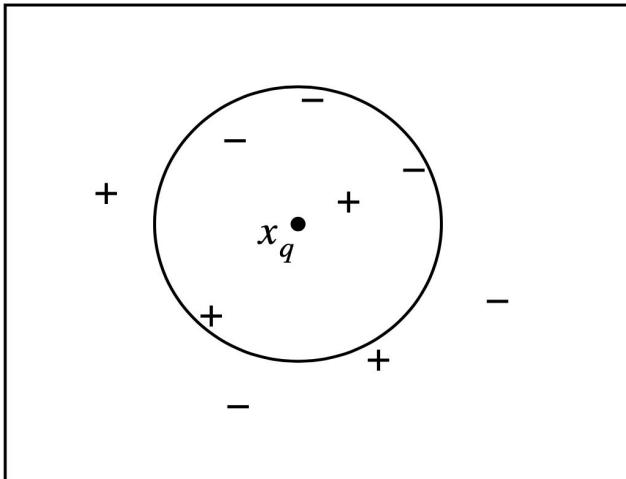
# Suposiciones iniciales

Todas las instancias corresponden a puntos en un espacio n-dimensional  $R^n$ .

- Los vecinos más cercanos de una instancia se definen en términos de la distancia euclíadiana estándar.
- Sea una instancia arbitraria  $x$  descrita por el vector de características  $(a_1(x), a_2(x), \dots, a_n(x))$  donde  $a_r(x)$  denota el  $r$ -ésimo atributo de la instancia  $x$ .
- La distancia entre dos instancias  $x_i$  y  $x_j$  se define como

$$d(x_i, x_j) = \sqrt{\sum_{r=1}^n (a_r(x_i) - a_r(x_j))^2}$$

# Límites de decisión de kNN



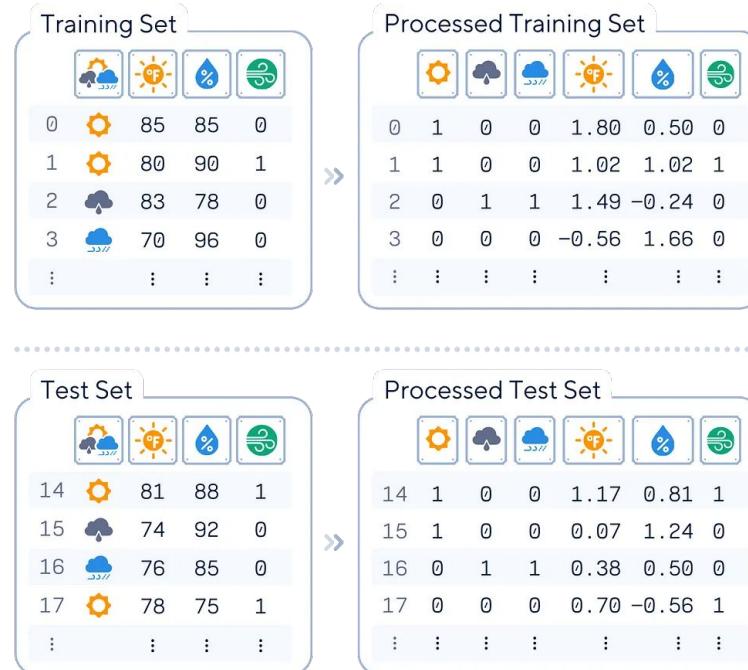
# Ejemplo: weather.numeric

Training Set						
0			85	85	0	No
1			80	90	1	No
2			83	78	0	Yes
3			70	96	0	Yes
4			68	80	0	Yes
5			65	70	1	No
6			64	65	1	Yes
7			72	95	0	No
8			69	70	0	Yes
9			75	10	0	Yes
10			75	70	1	Yes
11			72	90	1	Yes
12			81	75	0	Yes
13			71	80	1	No

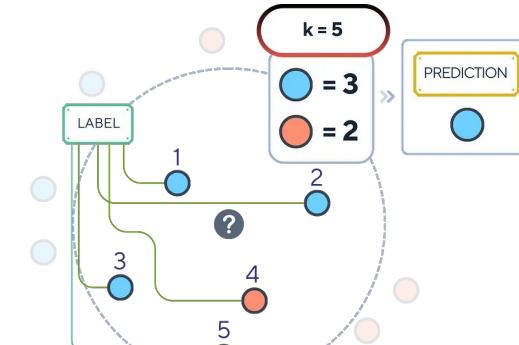
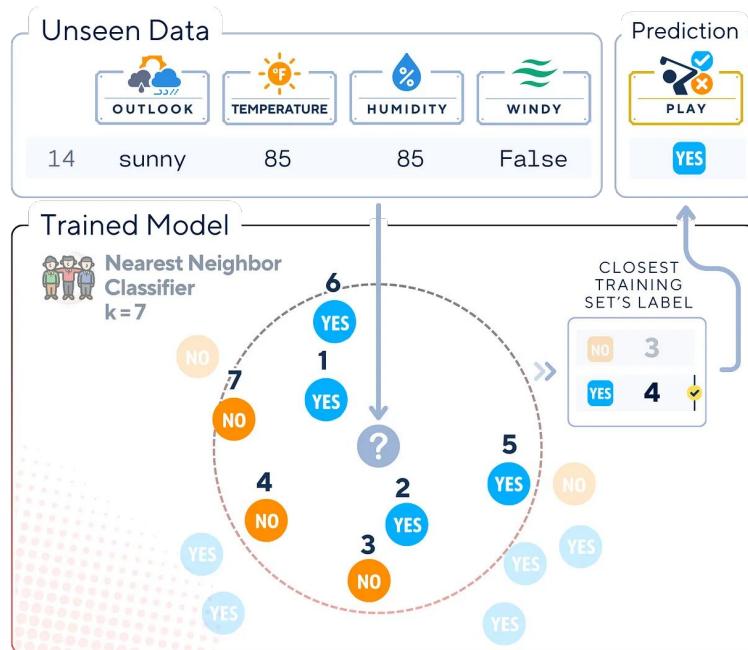
  

Test Set						
14			81	88	1	No
15			74	92	0	Yes
16			76	85	0	Yes
17			78	75	1	No
18			82	92	0	No
19			67	90	1	No
20			85	85	1	Yes
21			73	88	0	Yes
22			88	65	1	Yes
23			77	70	0	Yes
24			79	60	0	Yes
25			80	95	1	Yes
26			66	70	0	No
27			84	78	0	Yes

# Procesamiento, cuantificación



# Espacio de clasificación 2D



# Métrica de distancia

## FORMULA

### Euclidean Distance

COLUMN 1    COLUMN 2    COLUMN 3

X     $x_1$      $x_2$      $x_3$

Y     $y_1$      $y_2$      $y_3$

$$\frac{(x_1 - y_1)^2 + (x_2 - y_2)^2 + (x_3 - y_3)^2}{= \text{Sum Squared}}$$

$$\text{Euclidean Distance} = \sqrt{\text{Sum Squared}}$$

## EXAMPLE

### Distance between ID 0 and ID 1



0	1	0	0	1.80	0.50	0
1	1	0	0	1.02	1.02	1

$$\frac{0^2 + 0^2 + 0^2 + 0.78^2 + (-0.52)^2 + 1^2}{= 1.898}$$

$$\text{Euclidean Distance} = \sqrt{1.898} = 1.379$$

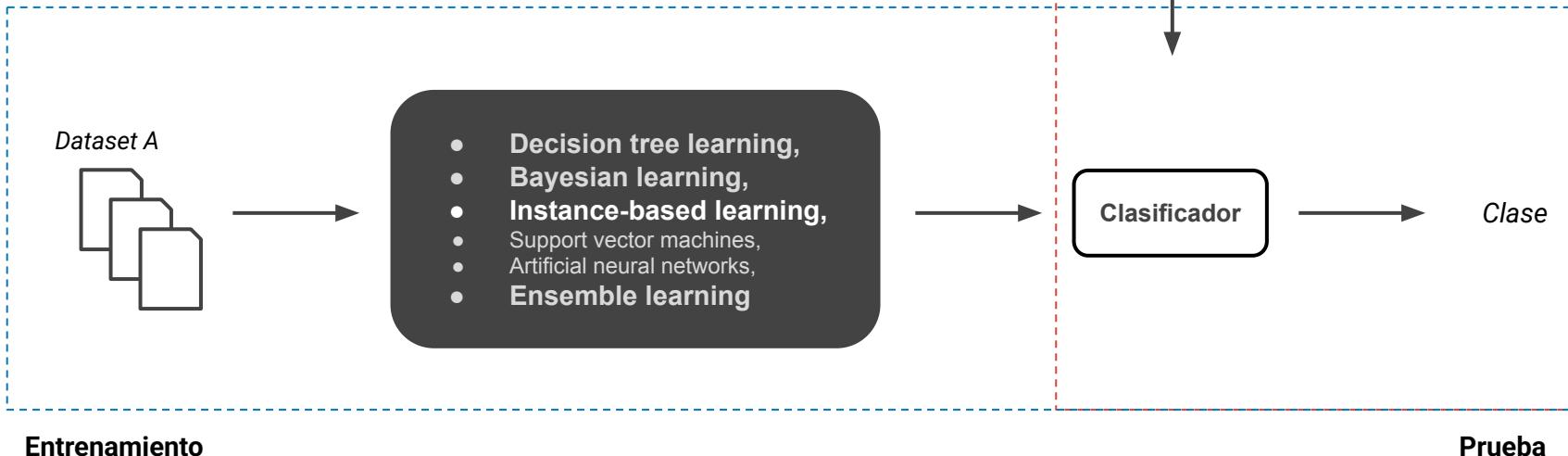
# Clasificación



# Evaluación

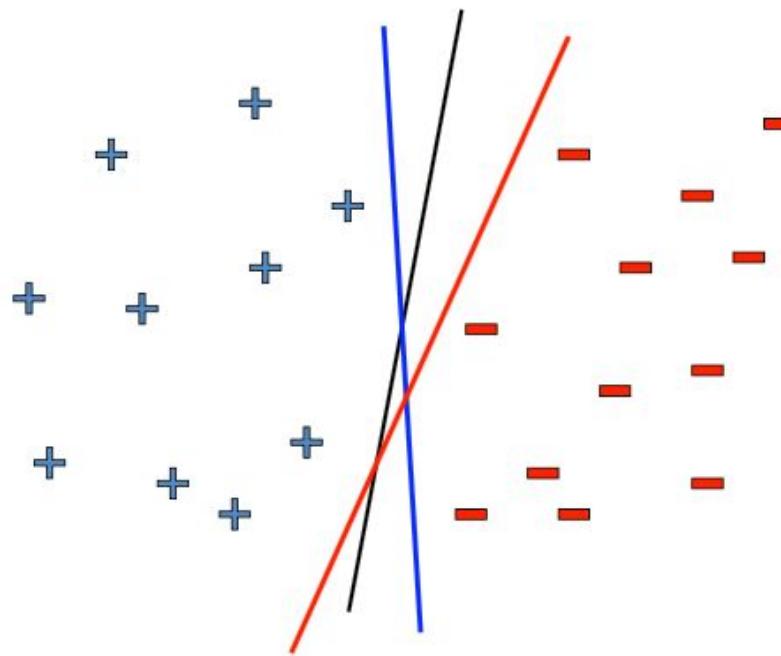
	$k=3$	$k=5$	$k=7$	
14	NO	NO	NO	YES
15	YES	YES	YES	YES
16	YES	YES	YES	YES
17	NO	YES	YES	YES
18	NO	NO	NO	YES
19	NO	YES	YES	YES
20	YES	YES	YES	YES
21	YES	YES	YES	YES
22	YES	YES	YES	YES
23	YES	YES	YES	YES
24	YES	YES	YES	YES
25	YES	YES	NO	YES
26	NO	YES	YES	YES
27	YES	YES	YES	YES
Accuracy Score	78% <sub>.71</sub>	71% <sub>.43</sub>	64% <sub>.29</sub>	

# Algoritmos de inferencia

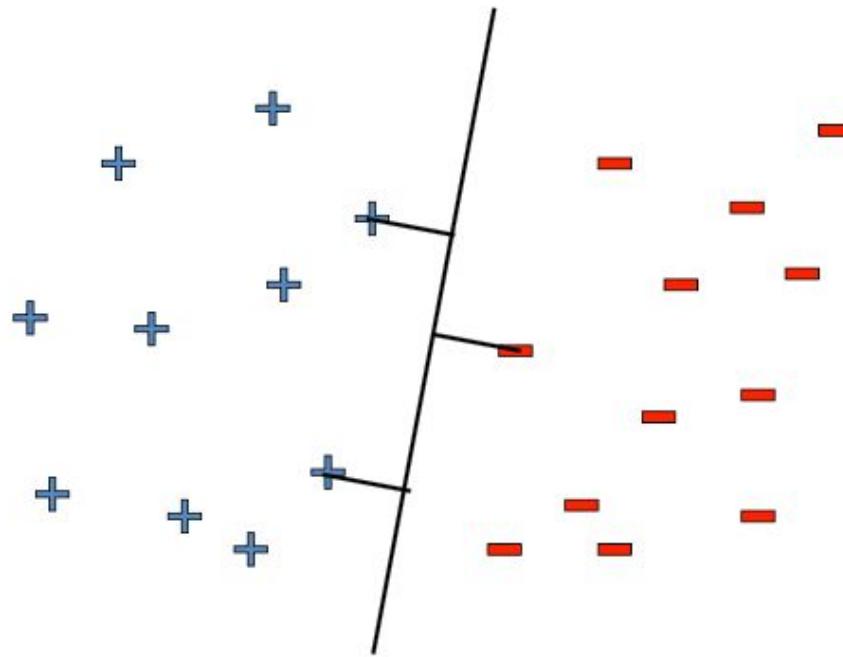




# Clasificadores lineales: ¿qué línea es mejor?

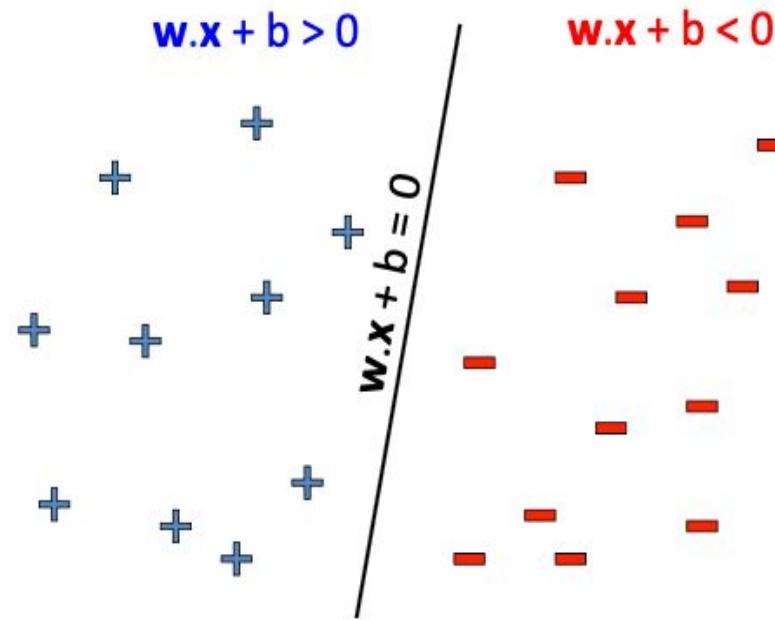


# Elegir el que tenga mayor margen



# Parametrización del límite de decisión

- Confianza:  
 $(w \cdot x_j + b) y_j$
- Etiquetas (class):  
 $y_j \in \{-1, +1\}$



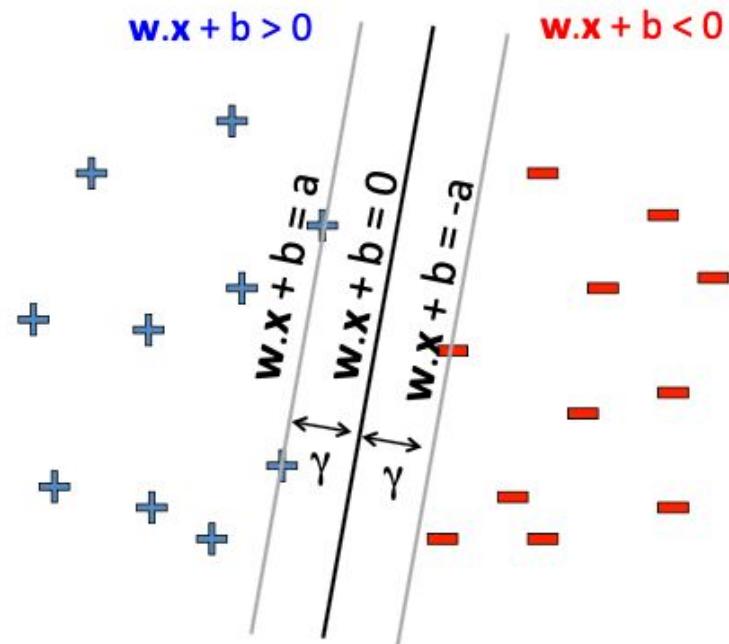
# Maximizando el margen

- Distancia de ejemplos más cercanos desde la línea/hiperplano.  
margen =  $\gamma$

$$\max_{w,b} \gamma = a / \|w\|$$

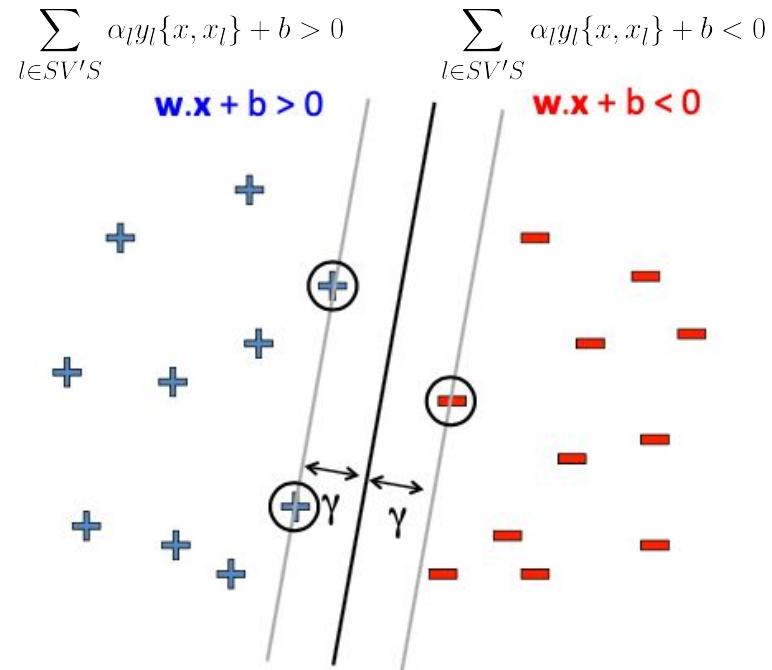
$$(w \cdot x_j + b)y_j \geq a \quad \forall j$$

- Tener en cuenta que 'a' es arbitraria

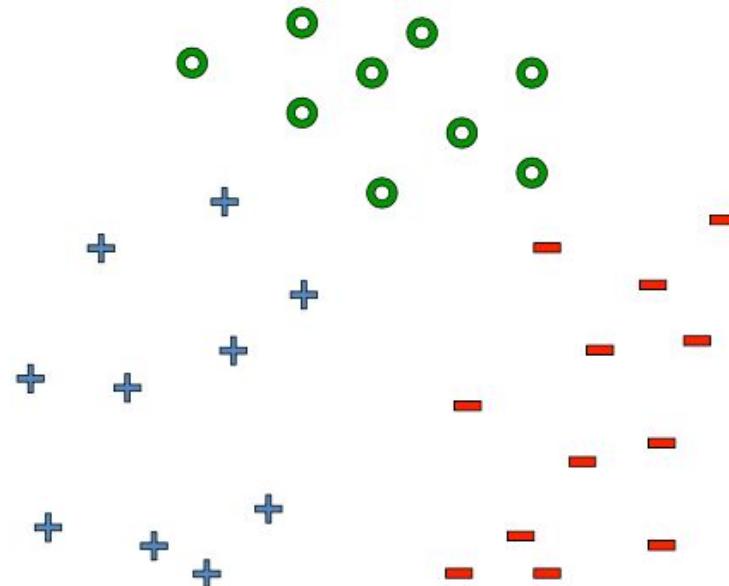


# Vectores de soporte

- El hiperplano lineal se define por "vectores de soporte".
- Mover un poco otros puntos no afecta el límite de decisión.
- Sólo es necesario almacenar los vectores de soporte para predecir etiquetas de nuevos puntos.



# ¿Qué pasa con varias clases?



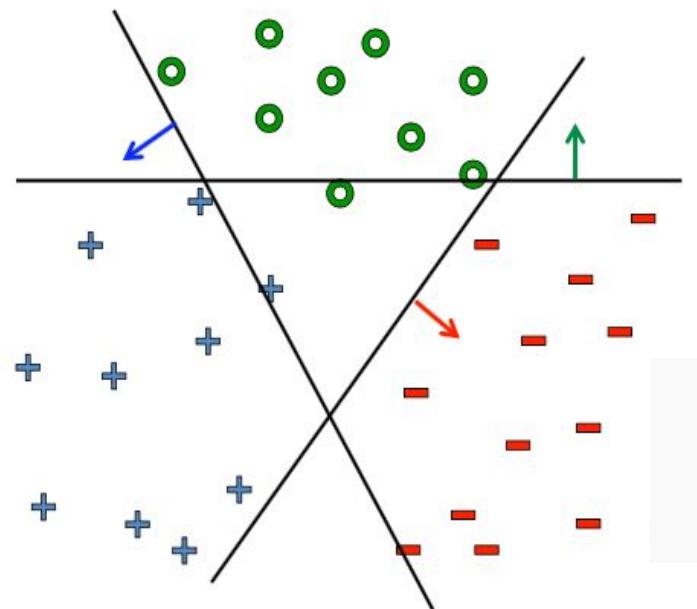
# Solución a problemas multiclase

- Aprender 3 clasificadores por separado:  
Clase k vs el resto

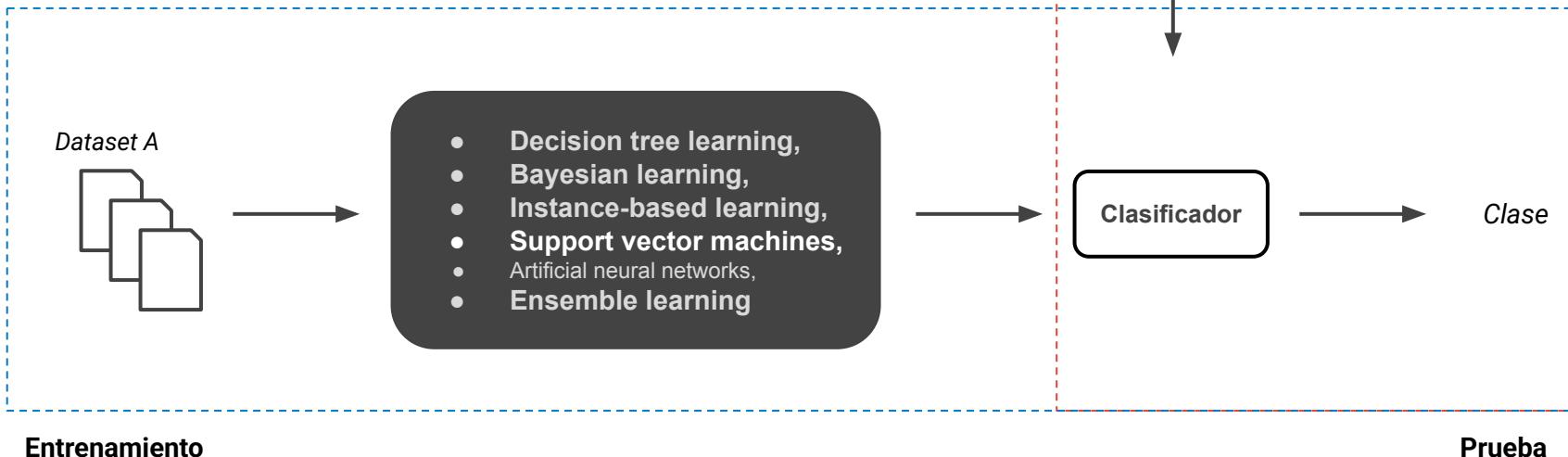
$$(w_k, b_k)_{k=1,2,3}$$

$$y = \arg \max_k w_k \cdot x + b_k$$

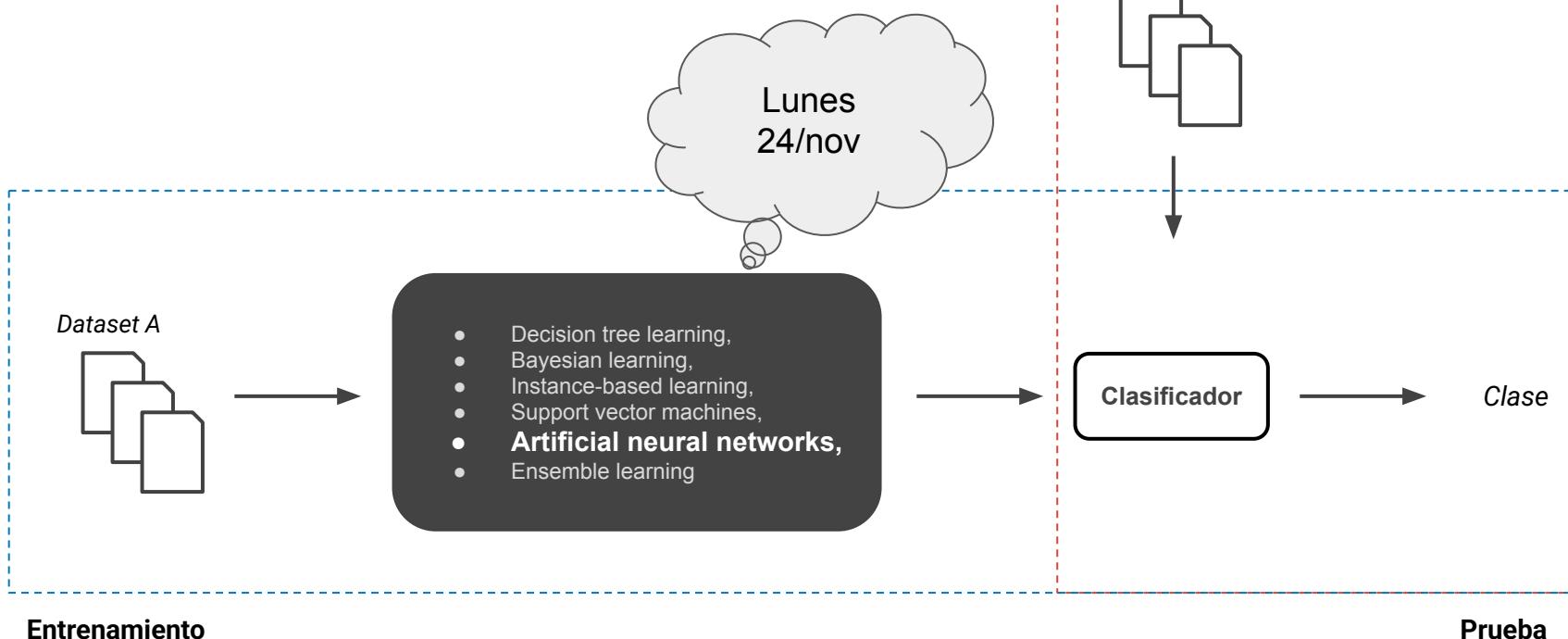
- Pero es posible que  $w_k$  no se base en la misma escala.
- Nota:  $(aw).x + (ab)$  también es una solución.



# Algoritmos de inferencia



# Algoritmos de inferencia



# ¿Preguntas?

[hussein@cicese.mx](mailto:hussein@cicese.mx)

