

flor

KNN o vecinos más cercanos

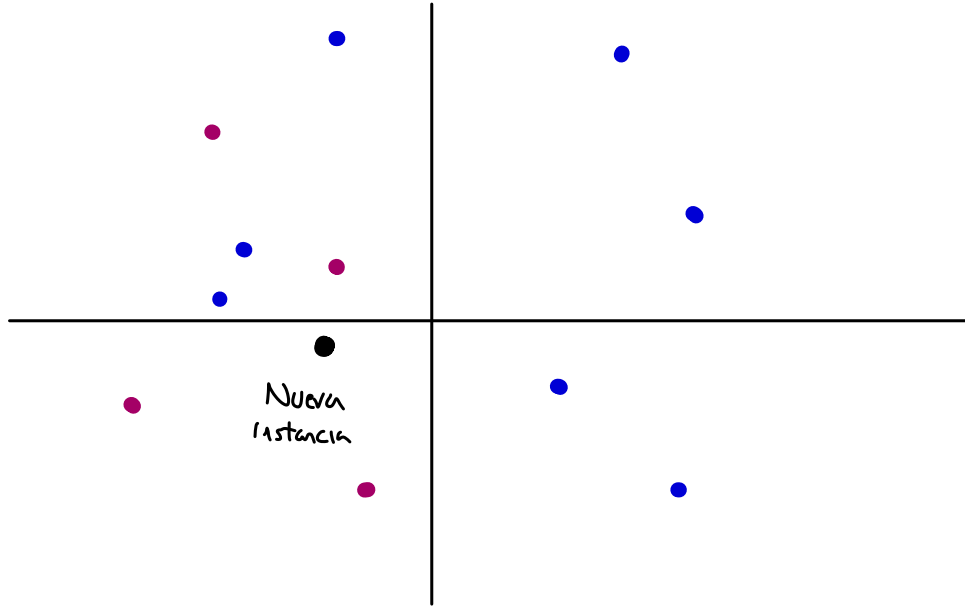
Referencia principal Libro de Tom Mitchell (Libro [2]) en la bibliografía

Referencias adicionales:

machinelearningmastery.com (K nearest neighbors for ML)

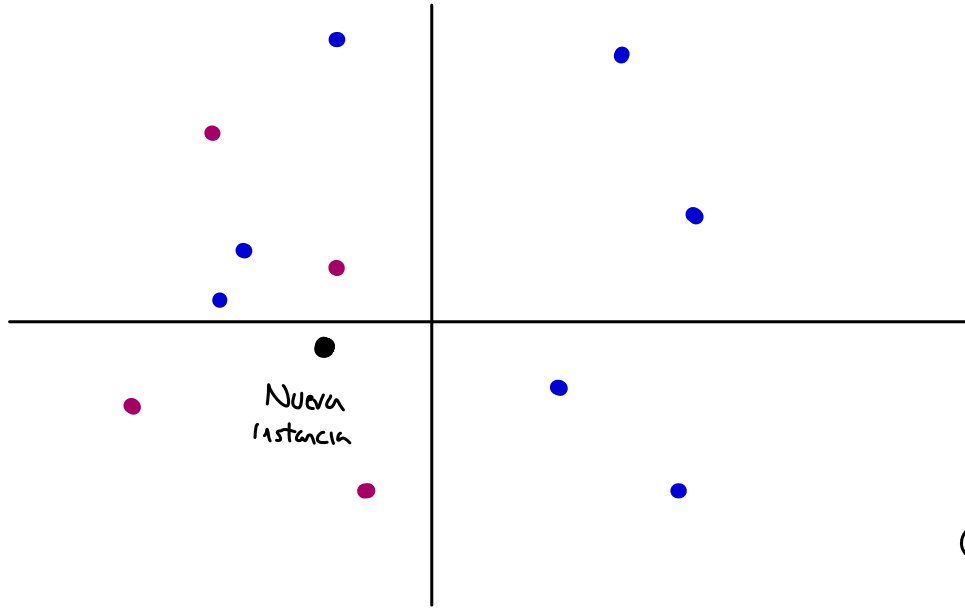
On k-Nearest Neighbor Voronoi Diagrams in the plane  
[Der-Tsai Lee 1982]

# K-Nearest Neighbors (Vecinos más cercanos)



¿Qué etiqueta le damos a la nueva instancia?

# K - Nearest Neighbors (Vecinos más cercanos)

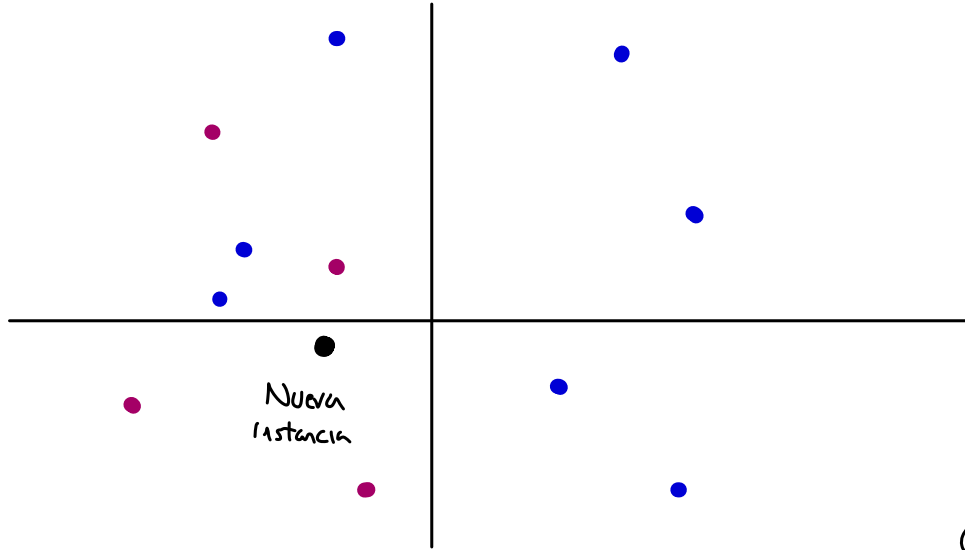


¿Qué etiqueta le damos a la nueva instancia?

Miramos las etiquetas de los vecinos más cercanos.

¿Cuántos de ellos? **K**

# K - Nearest Neighbors (Vecinos más cercanos)



¿Qué etiqueta le damos a la nueva instancia?

Miramos las etiquetas de los vecinos más cercanos

¿Cuántos de ellos?

K

K	etiqueta para la nueva instancia
---	----------------------------------

1

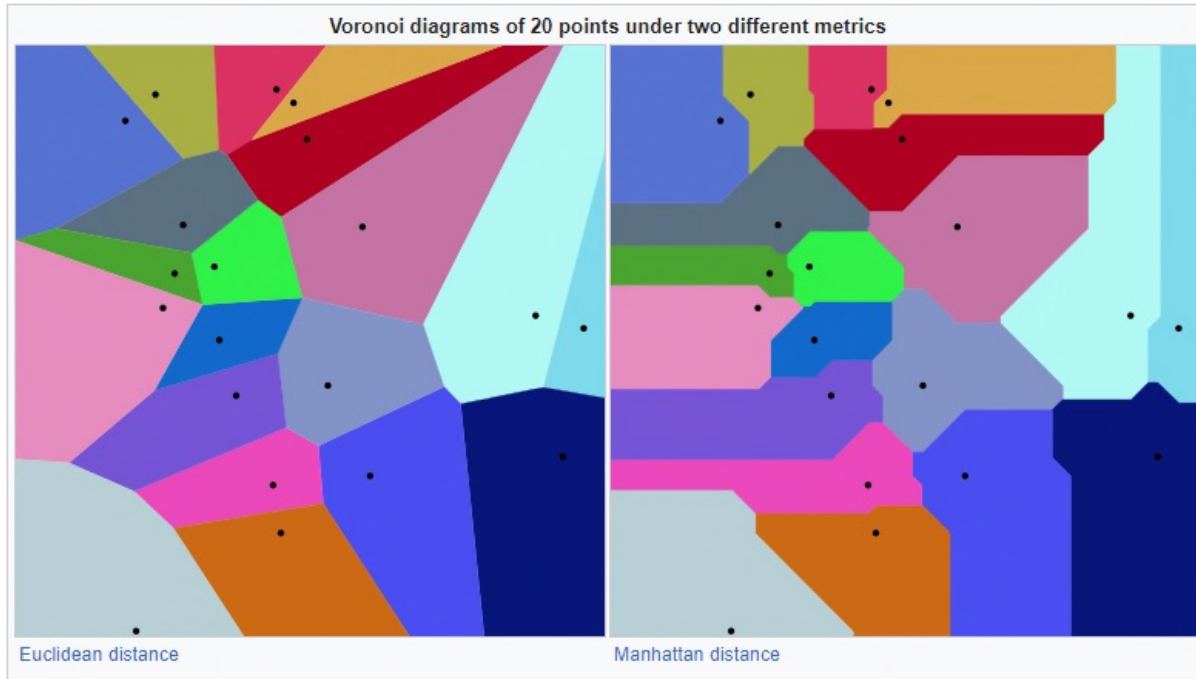
2

3

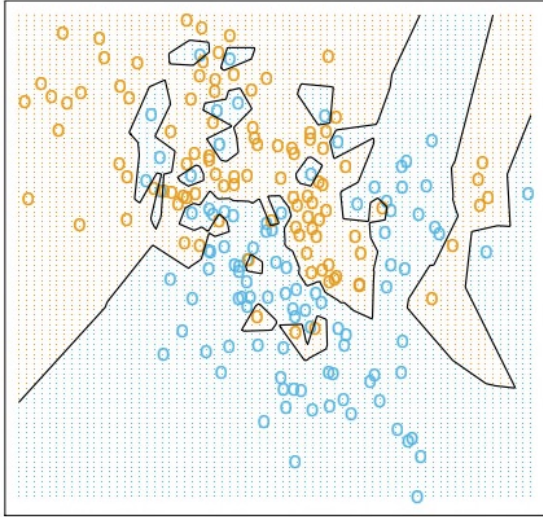
5

Se determina una estrategia para cuando hay empate (pej. Al azar)

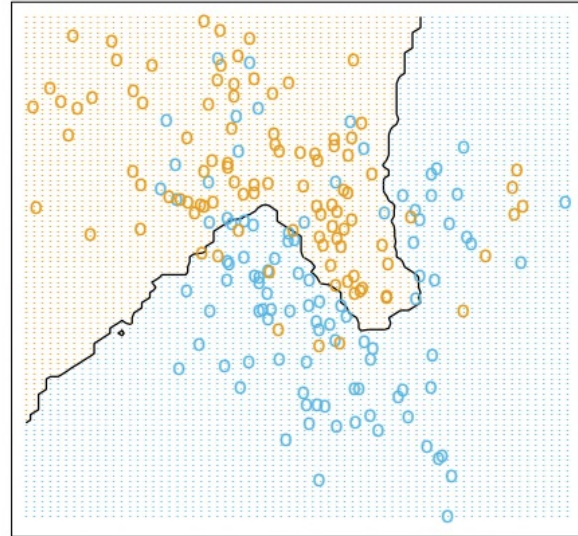
# Diagramas de Voronoi ( $k=1$ )



1-Nearest Neighbor Classifier

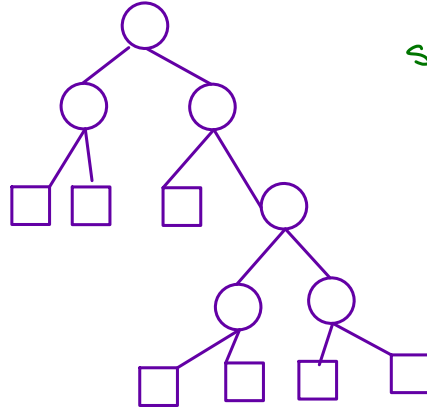
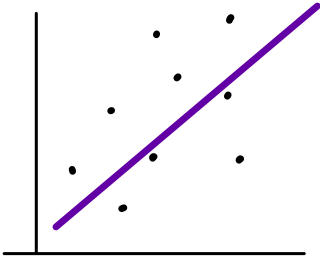


15-Nearest Neighbor Classifier



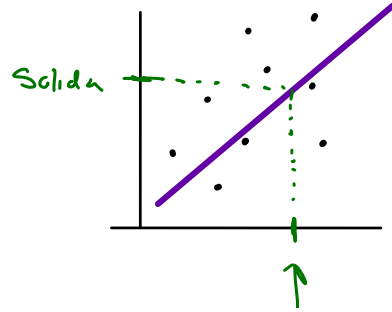
Antes

Los algoritmos primero aprenden  
de los datos para generar un modelo

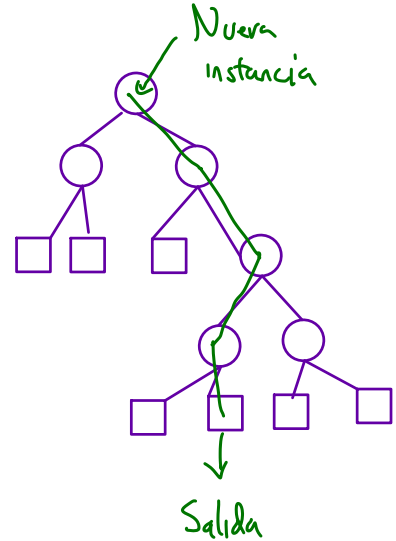


Aprendizaje (Learning)

Luego, para una consulta,  
se le preguntaría al modelo



Nueva instancia



Salida

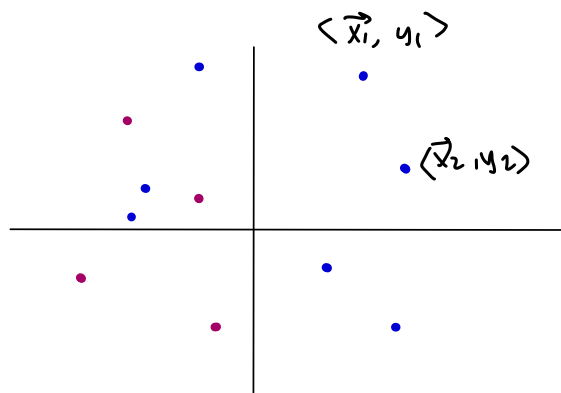
Consulta  
(Query)

$$\text{salida} = f(\text{nueva instancia})$$

↓  
función

Ahora

La fase que antes sería de **entrenamiento**, ahora sólo consiste en almacenar los datos de entrenamiento / registros / instancias



$\langle \vec{x}_1, y_1 \rangle$

$\langle \vec{x}_2, y_2 \rangle$

$\vdots$

$\langle \vec{x}_N, y_N \rangle$

se almacenan



Para la **consulta**, se hace una búsqueda en los datos guardados

Por esta razón, a este método se le llama **Método Basado en instancias**

Instance-Based Method



## El algoritmo (clasificación)

---

Training algorithm:

- For each training example  $\langle x, f(x) \rangle$ , add the example to the list *training\_examples*

Classification algorithm:

- Given a query instance  $x_q$  to be classified,
  - Let  $x_1 \dots x_k$  denote the  $k$  instances from *training\_examples* that are nearest to  $x_q$
  - Return

$$\hat{f}(x_q) \leftarrow \operatorname{argmax}_{v \in V} \sum_{i=1}^k \delta(v, f(x_i))$$

where  $\delta(a, b) = 1$  if  $a = b$  and where  $\delta(a, b) = 0$  otherwise.

---

**TABLE 8.1**

The  $k$ -NEAREST NEIGHBOR algorithm for approximating a discrete-valued function  $f : \mathbb{R}^n \rightarrow V$ .

## Variaciones del algoritmo

Distancia: Euclídea  
Manhattan/Taxista  
Etc

Training algorithm:

- For each training example  $\langle x, f(x) \rangle$ , add the example to the list *training\_examples*

Classification algorithm:

- Given a query instance  $x_q$  to be classified,
  - Let  $x_1 \dots x_k$  denote the  $k$  instances from *training\_examples* that are nearest to  $x_q$
  - Return

cómo definir más  
cerano

$$\hat{f}(x_q) \leftarrow \operatorname{argmax}_{v \in V} \sum_{i=1}^k \delta(v, f(x_i))$$

ver variaciones a continuación →

where  $\delta(a, b) = 1$  if  $a = b$  and where  $\delta(a, b) = 0$  otherwise.

**TABLE 8.1**

The  $k$ -NEAREST NEIGHBOR algorithm for approximating a discrete-valued function  $f : \mathbb{R}^n \rightarrow V$ .

Cómo elegir cuando hay empate en distancias?

## El caso de regresión

La función

$$\hat{f}(x_q) = \underset{v \in V}{\operatorname{argmax}} \sum_{i=1}^k \delta(v, f(x_i))$$

Cambia por

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



Promedio de los valores  
de los  $k$  vecinos más  
ceranos

## Distance-Weighted KNN

Una versión en la que se le da importancia a qué tan cerca están los vecinos

$$\hat{f}(x_q) \leftarrow \operatorname{argmax} \sum_{i=1}^k w_i S(v, f(x_i))$$

$$w_i = \frac{1}{d(x_q, x_i)^2}$$


o bien

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

# Comparaciones en tiempo y espacio de ejecución

Considere un conjunto de datos  
en la recta  $\mathbb{R}$  y ordenados

Complejidad computacional



		Tiempo de ejecución	Espacio en la memoria
1-NN	Learning		
	Query		
K-NN	Learning		
	Query		
Linear Regression	Learning		
	Query		

# Comparaciones en tiempo y espacio de ejecución

Considere un conjunto de datos  
en la recta  $\mathbb{R}$  y ordenados

Complejidad computacional

		Tiempo de ejecución	Espacio en la memoria
1-NN	Learning	1	$n$
	Query	$\log n$	1
K-NN	Learning	1	$n$
	Query	$\log n + k$	1
Linear Regression	Learning	$n$	1
	Query	1	1

Lazy learners

Eager learner

## Otras observaciones sobre K-NN

Desventajas: ▷ Costo computacional en la consulta

▷ Se involucra a todos los atributos por igual. Si la salida realmente depende sólo de algunos atributos, la distancia se ve afectada por aquellos de los que no depende

↘ ¿Cómo sobrellevarlo?

▷ Dándole pesos distintos a atributos distintos en la definición de la distancia

▷ Seleccionando atributos

▷ Sensible a la escala de los datos

▷ Normalizar o estandarizar los datos!

## Ventajas:

- ▷ Algoritmo robusto ante datos ruidosos
- ▷ El aprendizaje es económico computacionalmente
- ▷ Simple
- ▷ No depende de la distribución de los datos