09481: Inteligencia Artificial

Profesor del curso: Breyner Posso, Ing. M.Sc.
e-mail: breyner.posso1@u.icesi.edu.co

Programa de Ingeniería de Sistemas.
Departamento TIC.
Facultad de Ingeniería.
Universidad Icesi.
Cali, Colombia.

Agenda

- Introducción
- Clasificación
- Métricas
- K- vecinos más cercanos (KNN, K-Nearest Neighbors)

Introducción

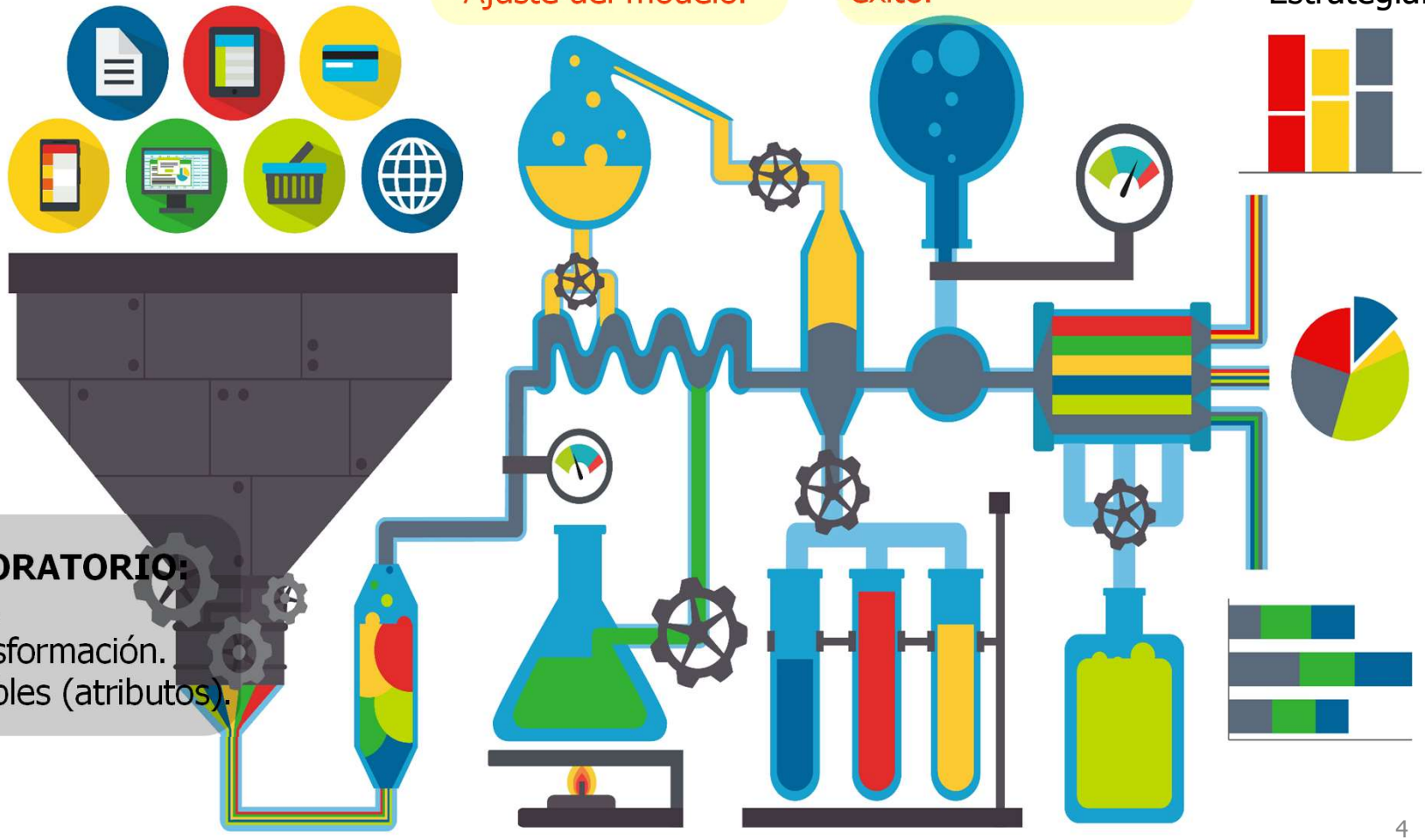
DATOS:
Materia prima.

MODELO:
Implementación del modelo de analítica.
Ajuste del modelo.

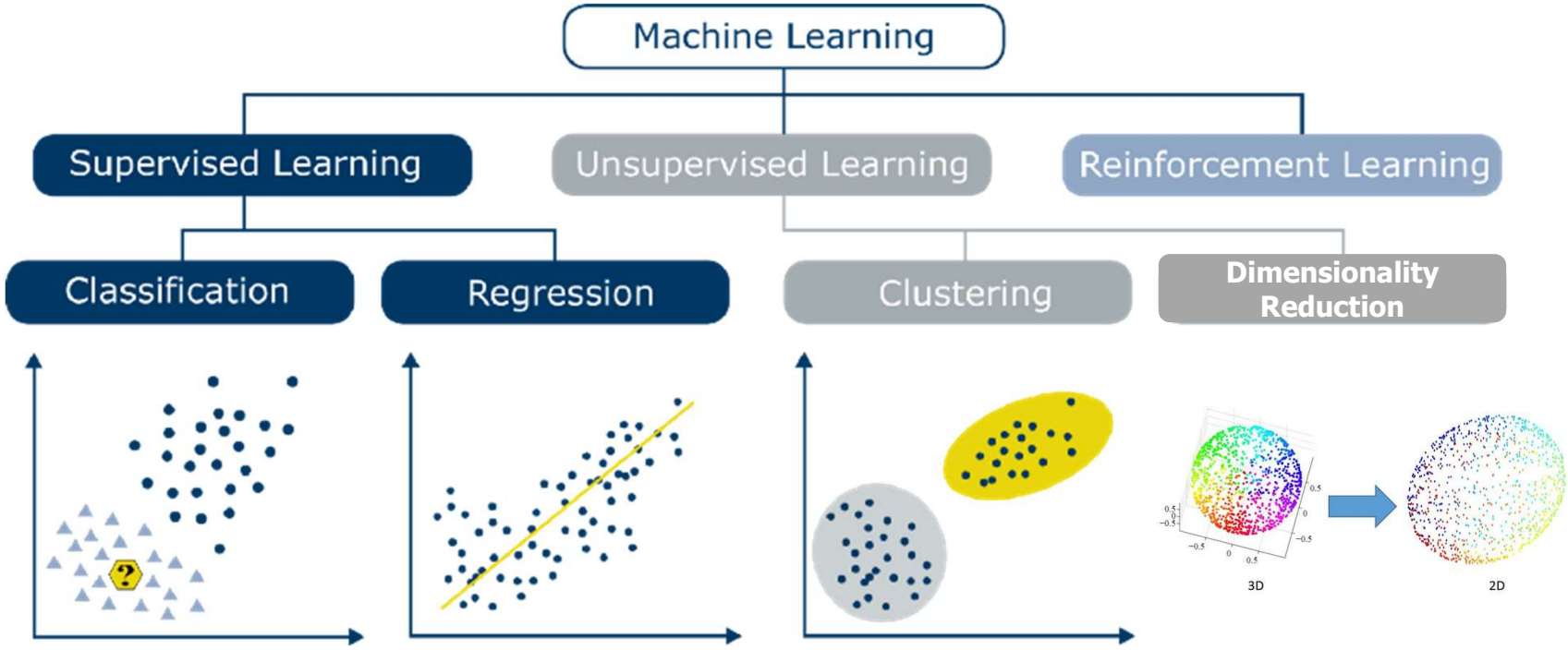
EVALUACION:
Viabilidad del negocio.
Validación criterios de éxito.

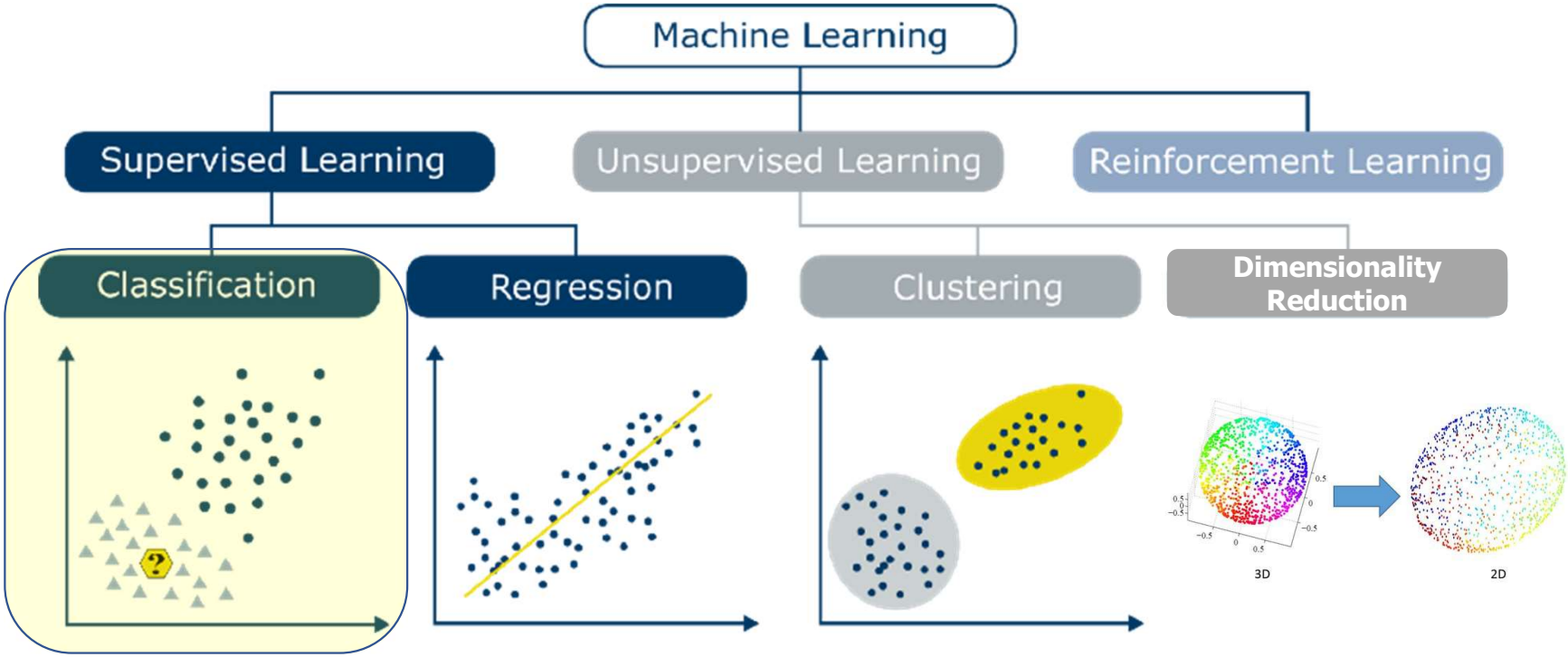
DESPLIEGUE:
Resultados.
Conocimiento.
Estrategia.

PREGUNTAS:
¿Cómo?
¿Cuáles?
¿Cuándo?
¿Por qué? *



ANALISIS EXPLORATORIO:
Limpieza de datos.
Preparación / transformación.
Selección de variables (atributos).


Machine Learning
Supervised Learning
Unsupervised Learning
Reinforcement Learning
Classification
Regression
Clustering
Dimensionality Reduction
3D
2D

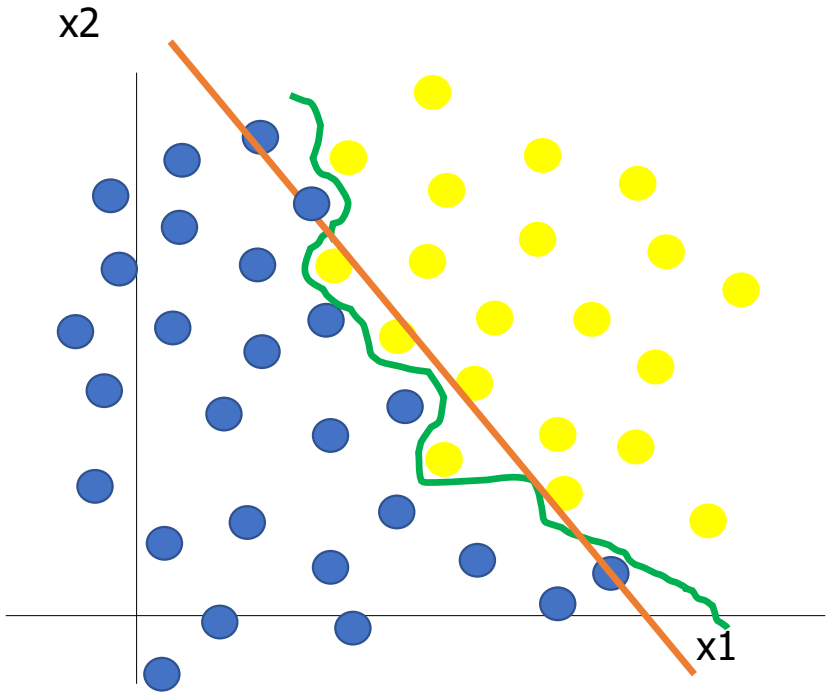

Machine Learning
Supervised Learning
Unsupervised Learning
Reinforcement Learning
Classification
Regression
Clustering
Dimensionality Reduction
3D
2D

Clasificación

Clasificación

Objetivo.

- Ajustar modelos para predecir valores **discretos** (clases o categorías) de la variable objetivo (target) con respecto a una o varias variables independientes (predictores).
- Esta predicción puede incluir la estimación de la **probabilidad** de pertenecer a cada una de las clases o categorías.

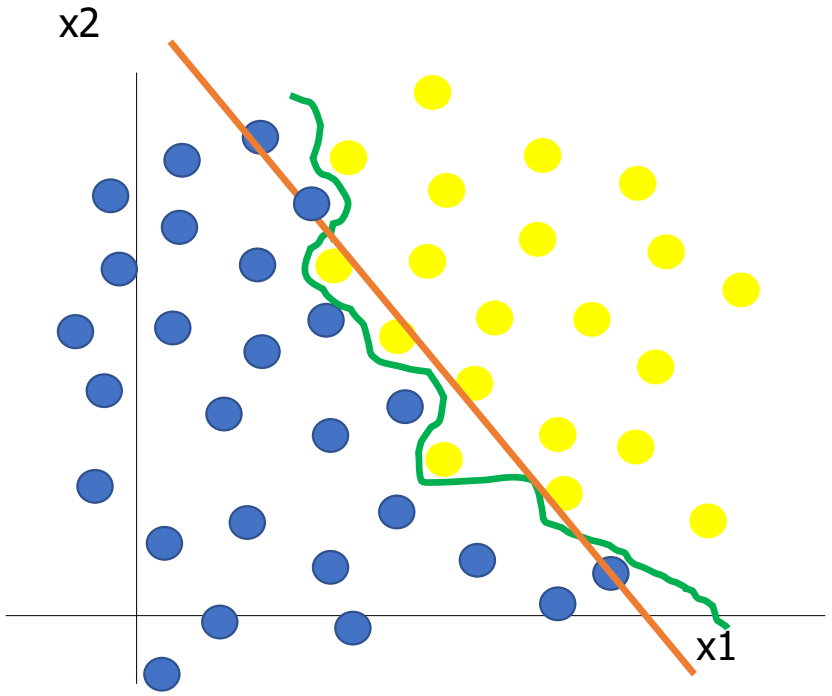


Clasificación

Tipos.

- ***Clasificación binaria***
- Los modelos de clasificación binaria predicen un resultado binario, es decir, una de las dos clases posibles.
- Ejemplos:

- ¿comprará el cliente este producto? [sí, no]
- ¿tipo de tumor? [maligno, benigno]
- ¿es este comportamiento una anomalía? [sí, no]
- ¿nos devolverá este cliente un crédito? [sí, no]



Clasificación

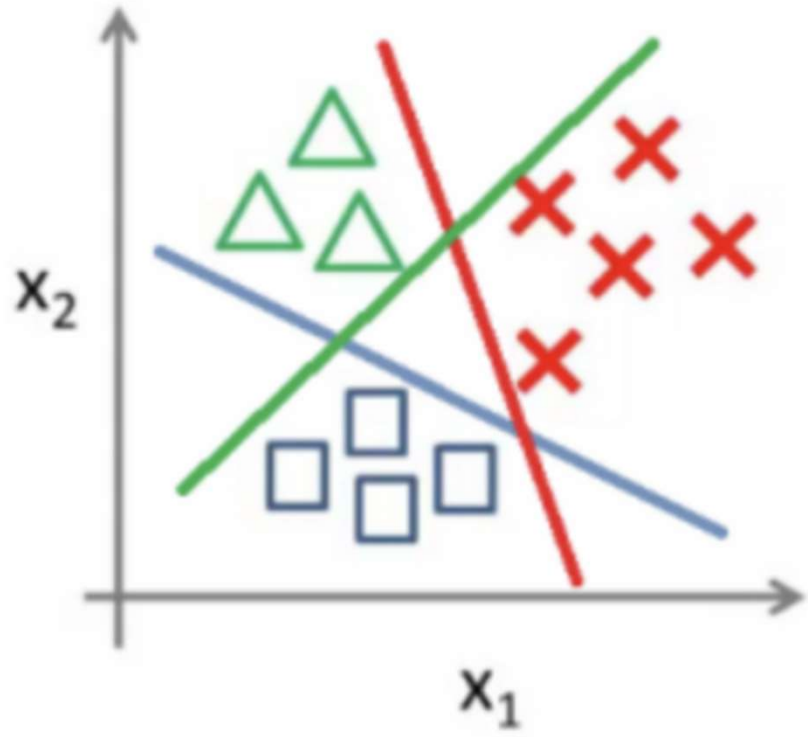
Tipos.

- ***Clasificación multiclase***
- Los modelos de clasificación multiclase predicen una clase entre un conjunto limitado de clases.
- Ejemplos:

- "¿Este producto es un libro, una película o una prenda de ropa?"

- "¿Esta película es una comedia romántica, un documental o un thriller?"

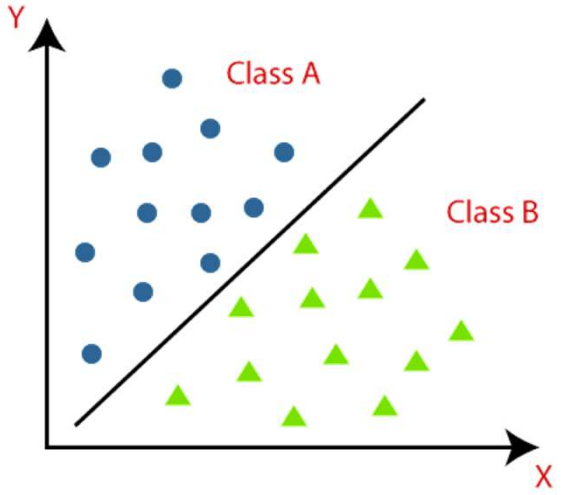
- "¿Qué categoría de productos es más interesante para este cliente?"

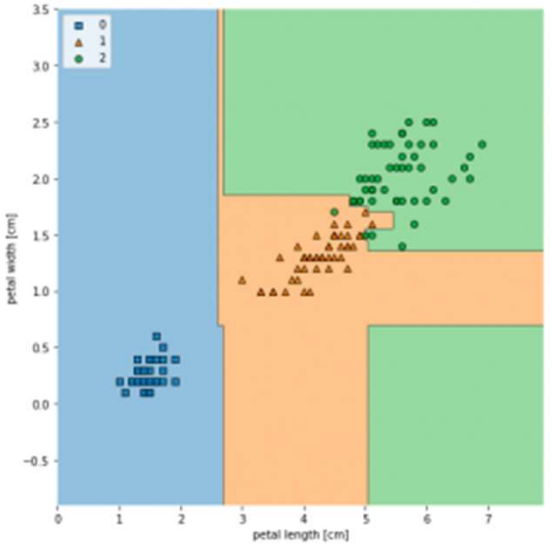


Clasificación

Métodos

- **Linear Models**
  - Logistic Regression
  - Support Vector Machines
- **Non-linear Models**
  - K-Nearest Neighbours
  - Kernel SVM
  - Naïve Bayes
  - Decision Tree Classification
  - Random Forest Classification.





Clasificación

Línea base.

- Medida de evaluación dada por un clasificador que escoge siempre la clase mayoritaria.

Clasificación

Línea base.

- Medida de evaluación dada por un clasificador que escoge siempre la clase mayoritaria.
- Por ejemplo, en un dataset usado para entrenar un clasificador que detecte correos *spam*, se tienen 10000 observaciones de las cuales 8000 son de correos deseados. En este problema la línea base sería el 80%, es decir, que un clasificador con un buen desempeño debería predecir de forma correcta los correos deseados por encima del 80%.

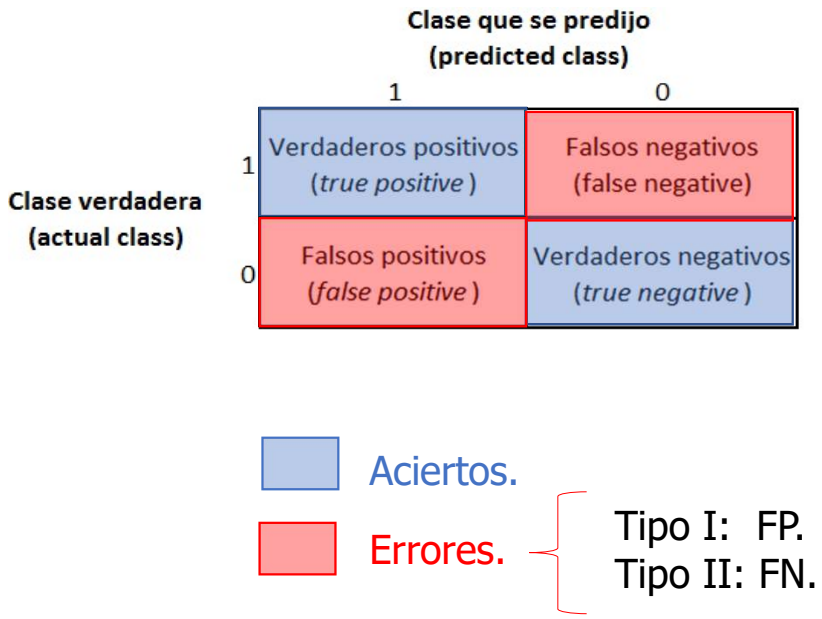
Clasificación binaria

Ejemplo

¿Cómo evaluar el desempeño de este clasificador?

idx	x (Atributos de entrada)	Y (clase verdadera)	Ypred (clase predicha)
1	[redacted]	1	1
2	[redacted]	1	1
3	[redacted]	1	1
4	[redacted]	1	1
5	[redacted]	1	1
6	[redacted]	1	1
7	[redacted]	0	1
8	[redacted]	1	0
9	[redacted]	1	0
10	[redacted]	0	0

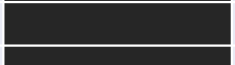
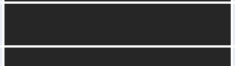
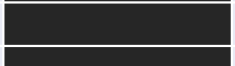
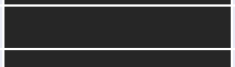
Clasificación binaria: matriz de confusión



Clasificación binaria

Ejemplo

¿Cómo evaluar el desempeño de este clasificador?

idx	x (Atributos de entrada)	Y (clase verdadera)	Ypred (clase predicha)
1		1	1
2		1	1
3		1	1
4		1	1
5		1	1
6		1	1
7		0	1
8		1	0
9		1	0
10		0	0

		Clase predicha	
		1	0
Clase Verdadera	1	**TP**	**FN**
	0	**FP**	**TN**

Clasificación binaria

Ejemplo

¿Cómo evaluar el desempeño de este clasificador?

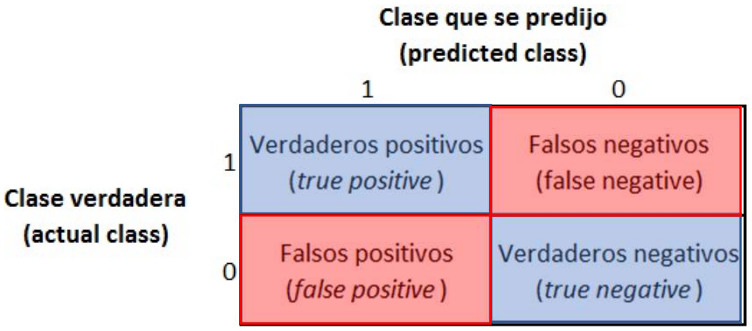
idx	x (Atributos de entrada)	Y (clase verdadera)	Ypred (clase predicha)
1	[redacted]	1	1
2	[redacted]	1	1
3	[redacted]	1	1
4	[redacted]	1	1
5	[redacted]	1	1
6	[redacted]	1	1
7	[redacted]	0	1
8	[redacted]	1	0
9	[redacted]	1	0
10	[redacted]	0	0

		Clase predicha	
		1	0
Clase Verdadera	1	**TP**	**FN**
	0	**FP**	**TN**

		Clase predicha	
		1	0
Clase Verdadera	1	**6**	**2**
	0	**1**	**1**

Clasificación binaria

- **Ejercicio:** identifique los elementos de la matriz de confusión y la importancia relativa de los errores de clasificación según el dominio de aplicación



- Caso 1: clasificación de correos electrónicos en spam vs. no-spam:

TP, TN:

FP: , consecuencia:

FN: , consecuencia:

- Caso 2: diagnóstico de una enfermedad grave como cáncer?

TP, TN:

FP: , consecuencia:

FN: , consecuencia:

Clasificación binaria: métricas de desempeño

		Predicted class: 1	Predicted class: 0	Row totals
True class	1	*True positives* (TP)	False negative (FN)	P=TP+FN
	0	*False positive* (FP)	*True negative* (TN)	N=FP+TN

$$TP\ rate = \frac{TP}{P}$$

$$accuracy = \frac{TP + TN}{P + N}$$

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{P}$$

$$F_1\ measure = \frac{2}{\frac{1}{precision} + \frac{1}{recall}}$$

$$specificity = \frac{TN}{FP + TN}$$

Métricas de desempeño: entre más cercano a **1,** mejor el clasificador.

$$FP\ rate = \frac{FP}{N}$$

$$error\ rate = \frac{FP + FN}{P + N}$$

Métricas de error: entre más cercano a **0,** mejor el clasificador.

En el caso de scikit-learn, puede calcular estas y otras métricas usando las funciones del módulo descrito en el siguiente enlace: https://scikit-learn.org/stable/modules/classes.html#module-sklearn.metrics

sklearn.metrics.confusion_matrix
sklearn.metrics.accuracy_score
sklearn.metrics.precision_score
sklearn.metrics.recall_score
sklearn.metrics.f1_score
....

Clasificación binaria: métricas de desempeño

- Exactitud (*accuracy*) = (VP+VN)/(VP+VN+FP+FN)
- Tasa de error= 1-exactitud = (FP+FN)/(VP+VN+FP+FN): probabilidad de error.
- Precisión= VP / (VP+FP): ¿qué proporción de todas las predicciones de pertenencia a la clase que hizo el clasificador fueron correctas? *P*(realidad=1|predicción=1)
- Sensibilidad (r*ecall* o *TP rate)*= VP / (VP+FN): ¿qué proporción de todos los ejemplos que pertenecían a la clase en la realidad, identificó el clasificador? *P*(predicción=1|realidad = 1)
- *Especificidad (o TNR): = VN / (VN+FP)*: ¿qué proporción de todos los ejemplos que en realidad NO pertenecían a la clase, pudo identificar bien? *P*(predicción=0|realidad=0)
- Métrica $F_1 = 2 * \frac{precision * recall}{precision + r}$

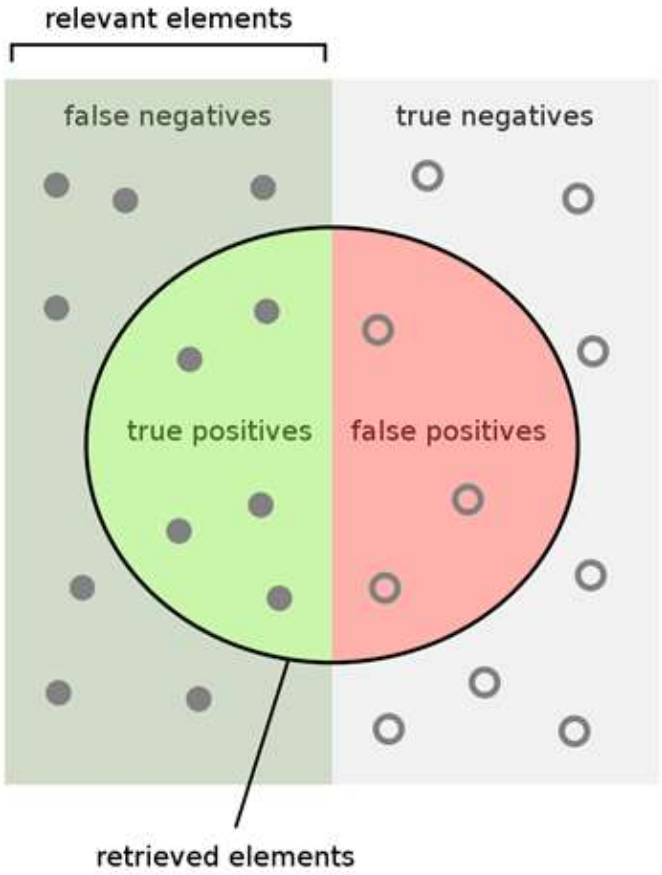
		Predicción	
		1	0
Realidad	1	VP	FN - Tipo II
	0	FP - Tipo I	VN

Convención
1: pertenece a la clase.
0: no pertenece a la clase.

Nota: para entender cómo se extienden estas métricas a la clasificación multi-clase, por favor leer el artículo:

Marina Sokolova, Guy Lapalme, *A systematic analysis of performance measures for classification tasks*, Information Processing & Management, Volume 45, Issue 4, 2009, Pages 427-437, ISSN 0306-4573, https://doi.org/10.1016/j.ipm.2009.03.002.

Clasificación binaria: métricas de desempeño



How many retrieved items are relevant?

$$\text{Precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$

How many relevant items are selected? e.g. How many sick people are correctly identified as having the condition.

$$\text{Sensitivity} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}} = \text{Recall}$$

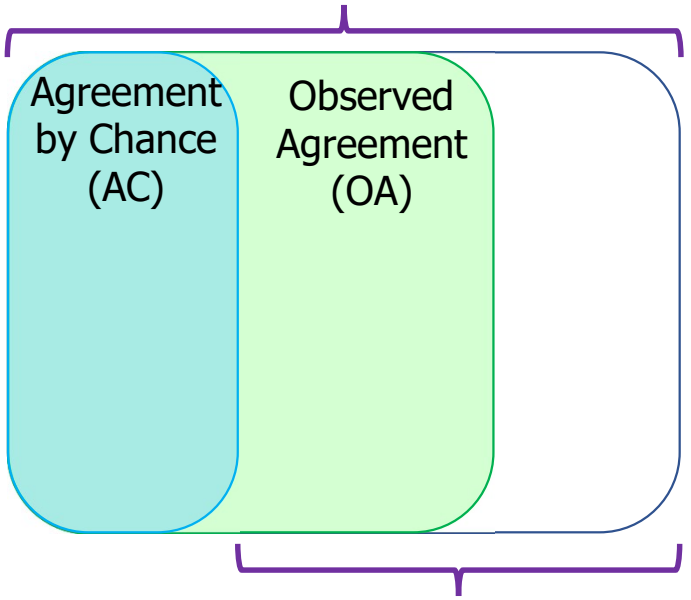
How many negative selected elements are truly negative? e.g. How many healthy people are identified as not having the condition.

$$\text{Specificity} = \frac{\text{true negatives}}{\text{true negatives} + \text{false positives}}$$

Clasificación: métrica de desempeño Kappa

- Coeficiente de concordancia **Kappa.**
- La Kappa de Cohen es una métrica que se usa a menudo para evaluar el acuerdo entre dos **evaluadores**. También se puede utilizar para evaluar el rendimiento de un modelo de clasificación.
- Tiene en cuenta la concordancia entre las predicciones y las clases reales.
- Al valor de la concordancia observada (OA), que es igual a la exactitud, se le sustrae el efecto de concordancia por suerte (AC).
- Para está métrica los valores son <= 1.
- Es muy útil cuando las clases no están balanceadas.
  - Diagnóstico de enfermedades raras.
  - Clientes que aceptan productos de crédito.
- Kappa = $\frac{OA-AC}{1-AC}$

Un modelo perfecto tendría una exactitud =100%



¿Qué tan bueno es el modelo actual comparado con el acuerdo por chance?

Clasificación: métrica de desempeño Kappa

Cohen's kappa measures the agreement between two raters who each classify *N* items into *C* mutually exclusive categories. The definition of κ is:

$$\kappa \equiv \frac{p_o - p_e}{1 - p_e} = 1 - \frac{1 - p_o}{1 - p_e},$$

where p_o is the relative observed agreement among raters, and p_e is the hypothetical probability of chance agreement, using the observed data to calculate the probabilities of each observer randomly seeing each category. If the raters are in complete agreement then $\kappa = 1$. If there is no agreement among the raters other than what would be expected by chance (as given by p_e), $\kappa = 0$. It is possible for the statistic to be negative,[6] which implies that there is no effective agreement between the two raters or the agreement is worse than random.

Fuente: Wikipedia

¿Cómo calcular kappa a partir de una matriz de confusión de un clasificador binario?

		Predicted class: 1	Predicted class: 0	Row totals
True class	1	*True positives* (TP)	False negative (FN)	P=TP+FN
	0	*False positive* (FP)	*True negative* (TN)	N=FP+TN

Total = P+N = TP+FN+FP+TN

$$Kappa = \left(\frac{OA - AC}{1 - AC} \right)$$

$$OA = accuracy = \frac{TP + TN}{Total}$$

$$AC = \left(\frac{FP + TN}{Total} \right)\left(\frac{FN + TN}{Total} \right) + \left(\frac{TP + FN}{Total} \right)\left(\frac{TP + FP}{Total} \right) = \frac{(FP + TN)(FN + TN) + (TP + FN)(TP + FP)}{Total \times Total}$$

%(true class==0) %(predicted class==0) %(true class==1) %(predicted class==1)

Nota: en sci-kit learn puede usar: sklearn.metrics.cohen_kappa_score.
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.cohen_kappa_score.html

Clasificación: métricas de desempeño

		Predicciones		
		+	-	TOTAL
reales	+	10	4	14
	-	3	2	5
	TOTAL	13	6	19

OA = 0,63
AC = 0,59
Kappa = 0,11

Accuracy (OA) = (10+2)/19=0,63
(AC) = ((13/19) * (14/19)) + ((6/19) * (5/19)) = 0,59
Kappa = (OA-AC)/(1-AC) =0,11

		Predicciones		
		+	-	TOTAL
reales	+	0	3	3
	-	0	97	97
	TOTAL	0	100	100

OA = 0,97
AC = 0,97
Kappa = 0,00

Accuracy (OA) = (0+97)/100=0,97
(AC) = ((0/100) * (3/100)) + ((100/100) * (97/100)) = 0,97
Kappa = (OA-AC)/(1-AC) =0

		Predicciones		
		+	-	TOTAL
reales	+	1475	988	2463
	-	556	1981	2537
	TOTAL	2031	2969	5000

OA = 0,69
AC = 0,50
Kappa = 0,38

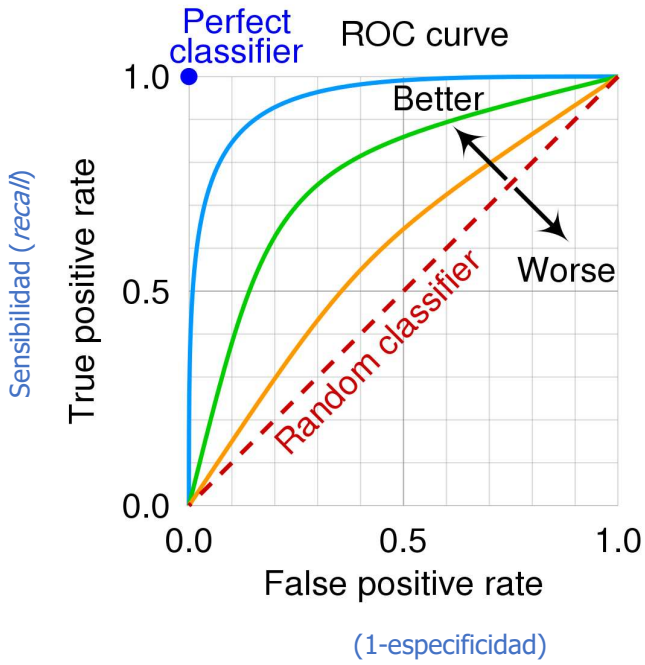
Nota: En años recientes, el uso de Kappa está siendo cuestionado como métrica de desempeño en problemas de clasificación.

Delgado, R., & Tibau, X. A. (**2019**). Why Cohen's Kappa should be avoided as performance measure in classification. *PloS one*, *14*(9)
https://doi.org/10.1371/journal.pone.0222916
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6762152/

Clasificación: métricas de desempeño

Comparación de varios modelos:

- Validación cruzada con exactitud o error de clasificación.
- Métrica F_1.
- AUC: Área bajo la curva *ROC (Receiver Operating Characteristic)*. Útil además para encontrar el umbral óptimo en clasificación binaria cuando la salida del modelo es continua (ej: una probabilidad).
- Dependiendo del contexto del problema, se debe escoger la métrica apropiada.

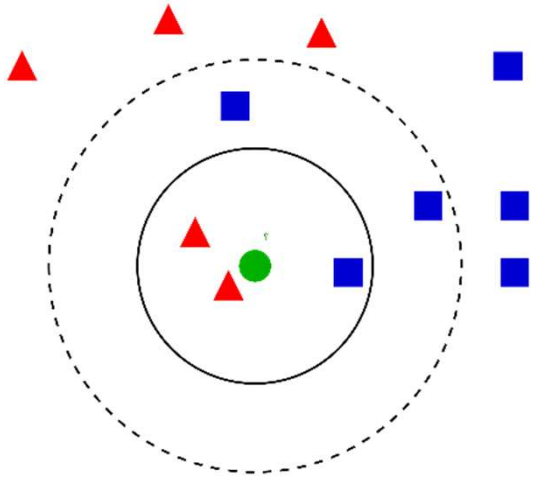


Fuente: Wikipedia

K-vecinos más cercanos (KNN: K-Nearest Neighbors)

KNN

- Algoritmo de aprendizaje supervisado para **clasificación** y **regresión.**
- **Sencillo**: asignar la clase o valor agregado de las instancias conocidas que se encuentran mas cerca de la instancia a predecir**.**
- Basado en las **instancias** de aprendizaje, no en un modelo subyacente probabilístico/estadístico**.**
- Aprendizaje **perezoso**: en realidad el algoritmo sólo se ejecuta en el momento que se requiere predecir una nueva instancia a partir de una predicción local**.**

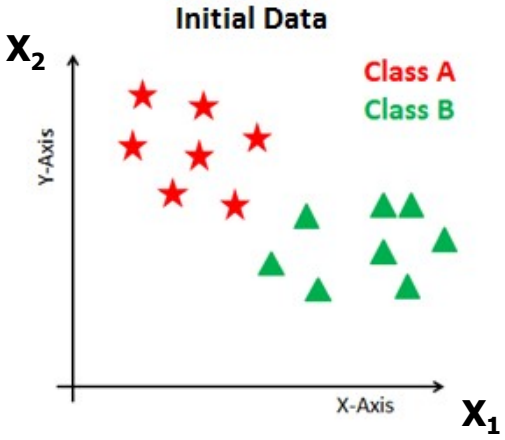


Fuente: Wikipedia.

KNN

Algoritmo

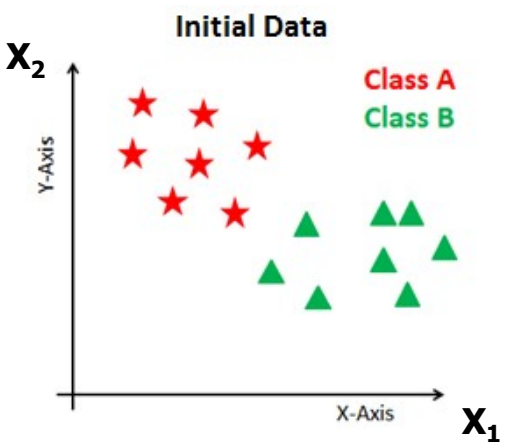
(1)



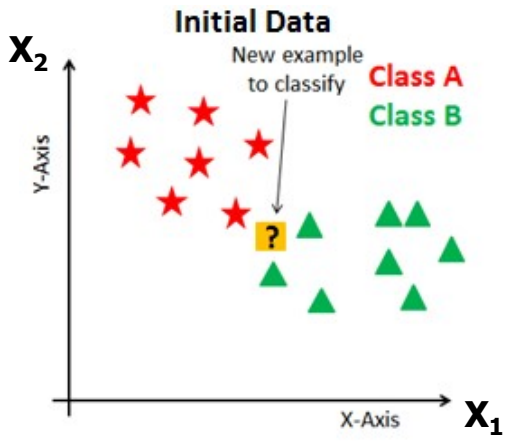
KNN

Algoritmo

(1)



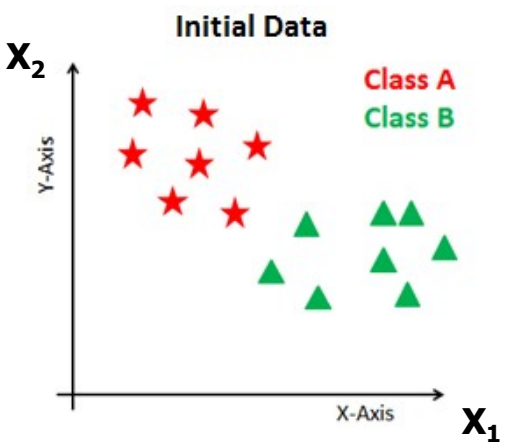
(2)



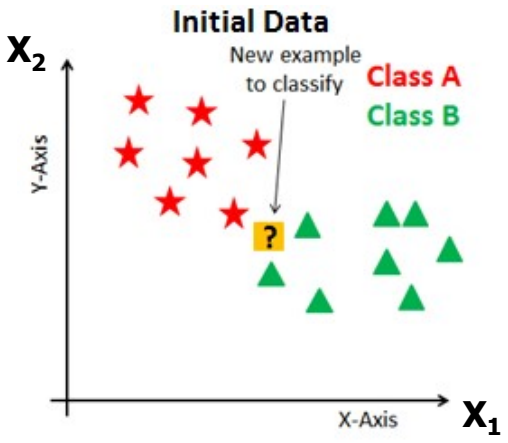
KNN

Algoritmo

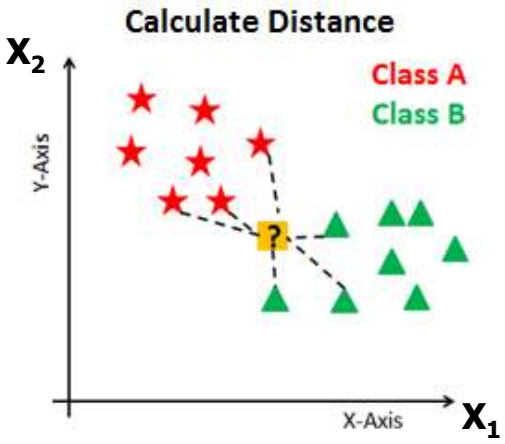
(1)



(2)



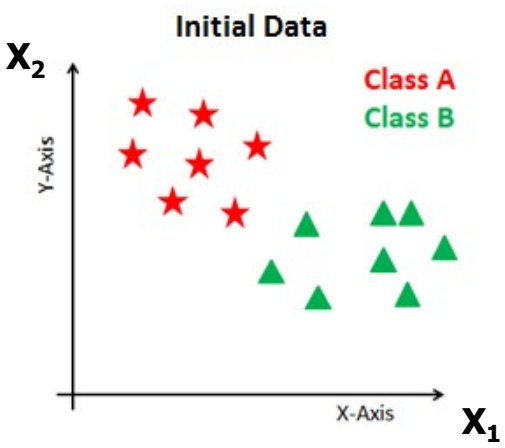
(3)



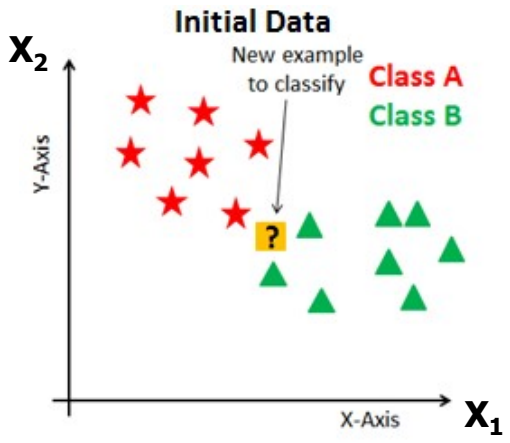
KNN

Algoritmo

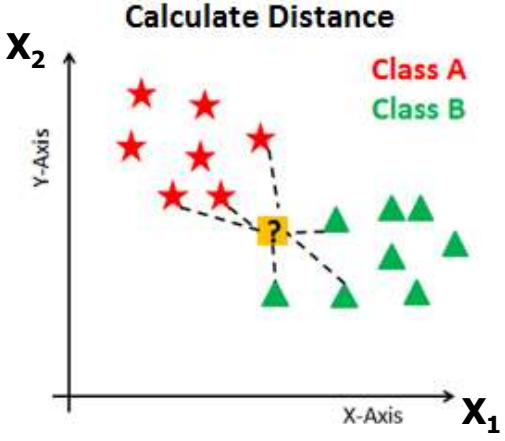
(1)



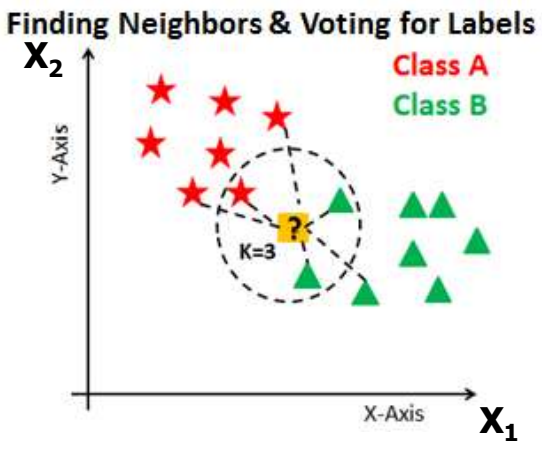
(2)



(3)

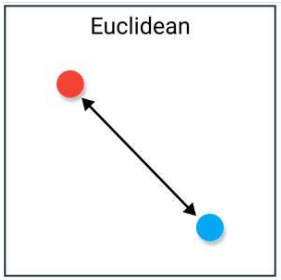


(4)

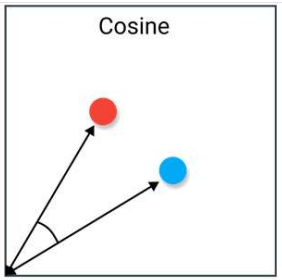


KNN: distancias y similitudes

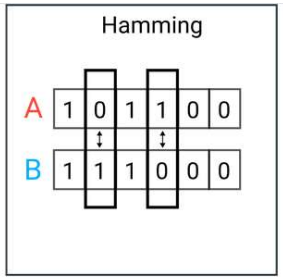
- KNN requiere una medida de **similitud** o **distancia** que se debe definir con antelación para encontrar los vecinos más cercanos.
- Para ser una medida **distancia**, una función *d* **debe** tener las siguientes propiedades:
  d(x, y) >= 0
  d(x, y) = *d* (y, x)
  d (x, y) = 0 *si y sólo si* x = y
  d (x, z) <= *d* (x, y) + *d* (y, z) conocida como la desigualdad del triángulo.
- Nota: Cuanto más distante, menos parecido!



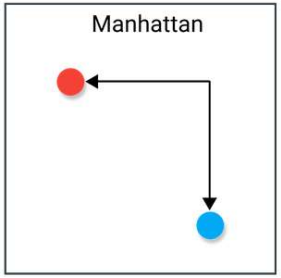
$$D(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$



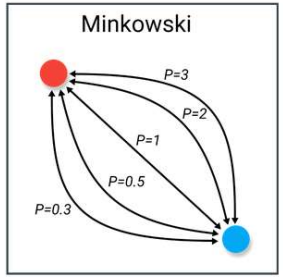
$$similarity(x,y) = \cos(\theta) = \frac{x \cdot y}{\|x\|\|y\|}$$



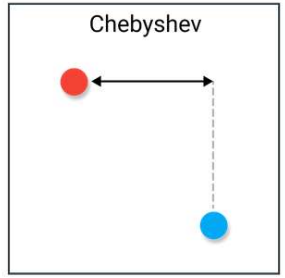
Número de valores diferentes entre dos vectores de la misma longitud.



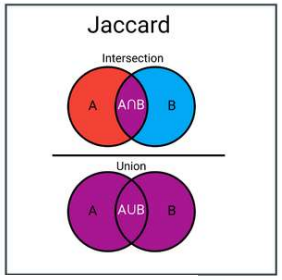
$$D(x,y) = \sum_{i=1}^{k}|x_i - y_i|$$



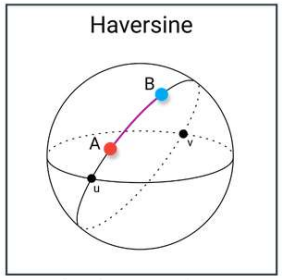
$$D(x,y) = \left(\sum_{i=1}^{n}|x_i - y_i|^p\right)^{\frac{1}{p}}$$



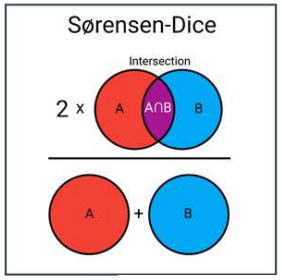
$$D(x,y) = \max_i(|x_i - y_i|)$$



$$D(x,y) = 1 - \frac{|x \cap y|}{|y \cup x|}$$



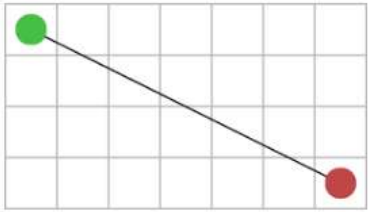
$$d = 2rarcsin\left(\sqrt{sin^2\left(\frac{\varphi_2 - \varphi_1}{2}\right) + cos(\varphi_1)cos(\varphi_2)sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right)$$



$$D(x,y) = \frac{2|x \cap y|}{|x| + |y|}$$

KNN: distancias y similitudes

Euclidiana: no es invariante en la escala, lo que significa que las distancias calculadas pueden estar sesgadas según las unidades de las características. Normalice los datos antes de usar esta medida de distancia en KNN. Apropiada en espacios de baja dimensionalidad.

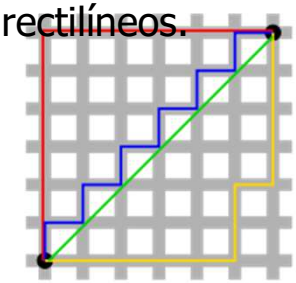


$$\mathrm{d}(\mathbf{p},\mathbf{q}) = \mathrm{d}(\mathbf{q},\mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

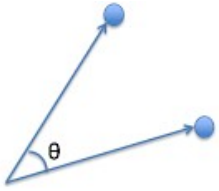
Manhattan: basada en una organización en bloques rectilíneos.

$$\sum_{i=1}^{n} |x_i - y_i|$$



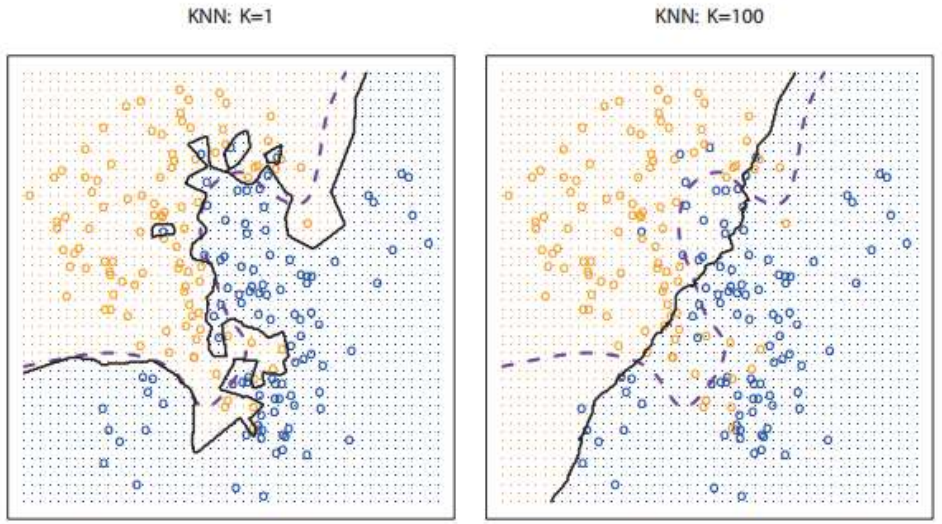
- **Similitud del coseno**: corresponde al coseno del ángulo entre las dos instancias. Es apropiada para espacios de alta dimensionalidad y en problemas de **big data.**

$$sim(\boldsymbol{x},\boldsymbol{y}) = \cos(\theta_{\boldsymbol{x},\boldsymbol{y}}) = \frac{\boldsymbol{x}.\boldsymbol{y}}{\|\boldsymbol{x}\|\|\boldsymbol{y}\|} = \frac{\sum_i x_i * y_i}{\sqrt{\left(\sum_i (x_i * x_i)\right) * \left(\sum_i (y_i * y_i)\right)}}$$



KNN: K vecinos

- **Parámetro *K*:** número de vecinos más cercanos a considerar para establecer la clase o valor de una nueva instancia.
  - El resultado puede ser drásticamente diferente para varios valores de *K*.
  - Un valor de *K* grande suavizará los límites entre clases/valores (alto sesgo, baja varianza).
  - Un valor de *K* pequeño resultará en límites muy flexibles (bajo sesgo, alta varianza).
  - El valor óptimo de *K* se encuentra empíricamente.

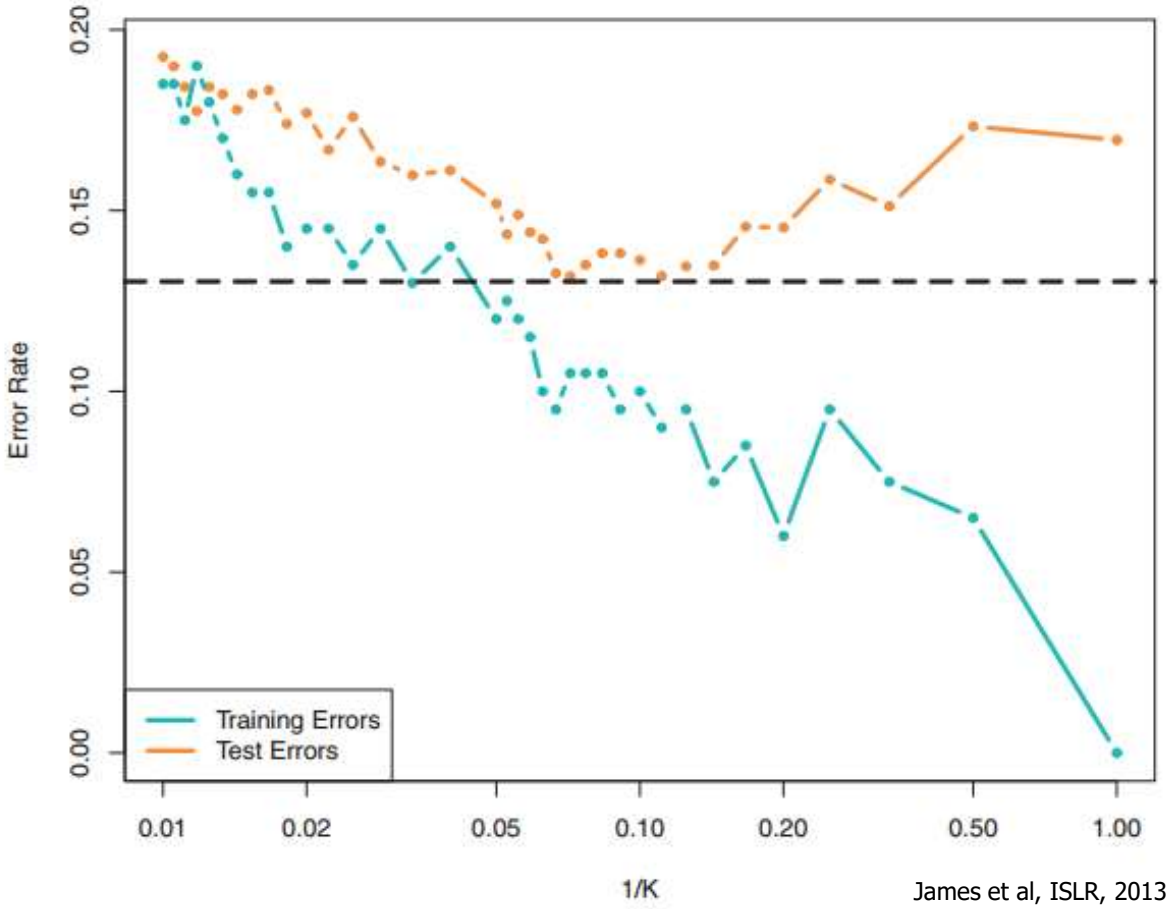


James et al, ISLR, 2013

Línea punteada: frontera de decisión de Bayes.
Línea continua: frontera de decisión de KNN.

KNN: K vecinos

- **Sobre aprendizaje o sobre ajuste (*overfitting*)**: este problema debe tenerse en cuenta al momento de escoger el valor de *K*.
- Valores pequeños de k pueden conducir a o*verfitting*.
- Valores grandes de k pueden conducir a *underfitting*.

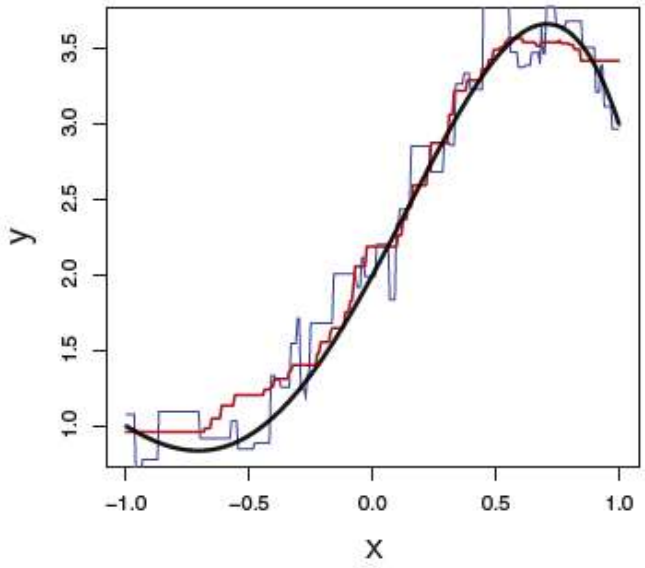


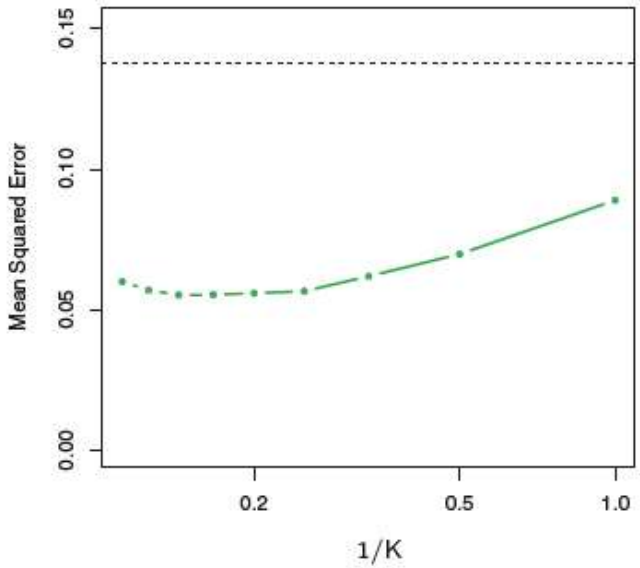
James et al, ISLR, 2013

KNN: K vecinos

En el caso de utilizar KNN para regresión las mismas consideraciones anteriores aplican:

- En el panel izquierdo: se aplica KNN con un valor de K=1 (azul) y K=9 (rojo).
- En el panel derecho, se puede ver el valor de *RMSE* para diferentes valores de $1/K$ (en verde). También se puede ver, por comparación el nivel de error de la regresión lineal simple (punteada en negro).





James et al, ISLR, 2013

KNN: consideraciones

- Perezoso (*lazy learning*).
- No paramétrico (no tiene pesos que aprender) y no lineal (hace referencia a la frontera de decisión).
- **Método local, no generalizable (NO hay un modelo construido como tal)**:
  - Puede encontrar particularidades muy específicas de ciertas regiones.
  - Su uso (sobre todo en regresión) sólo permite estimaciones en los rangos de las variables del conjunto de aprendizaje (la extrapolación no tiene mucho sentido).
- Sufre de la maldición de la **dimensionalidad.**
- Muy sensible a la **unidad de medida** de los atributos (se deben **normalizar** las variables para evitar diferencias en sus importancias finales), y a atributos que no aportan poder predictivo (e.g.: el color de los ojos no debería considerarse para predecir la edad de una persona).
- **Variaciones**: KNN ponderado por la distancia, basado en un radio dado.

CNN (*Condensed Nearest Neighbors*)

- Dificultad de aplicación de KNN cuando se tienen **muchos registros.**
- No todos los registros son necesarios para una correcta clasificación.
- Aproximación de KNN utilizando un conjunto de datos reducido.
- Estrategia: escoger **prototipos** que permitan una clasificación con K=1 lo más parecida al resultado utilizando el conjunto de datos completo.
- Algoritmo: Siendo ***X*** el conjunto de datos inicial y ***U*** el conjunto reducido:
  - Identificar todos los elementos x de ***X*** cuyo vecino más cercano sea de clase diferente.
  - Retirar los x identificados (son prototipos) de ***X*** y agregarlos a ***U***.
  - Repetir hasta que no se agreguen más prototipos a ***U***.

Lecturas complementarias

KNN

https://www.codecademy.com/learn/introduction-to-supervised-learning-skill-path/modules/k-nearest-neighbors-skill-path/cheatsheet

KNN Algorithm for Machine Learning

https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning

Referencias

- *Introduction to Statistical Learning with Applications in R (ISLR), G. James, D. Witten, T. Hastie & R. Tibshirani, 2014.*

- *Data Mining (4th Edition)*, Ian Witten, Eibe Frank, Mark A. Hall & Christopher J. Pal, Elsevier, 2016.

- *Machine Learning,* Tom M. Mitchell, McGraw-Hill, 1997.

- *Data Science for Business*, Foster Provost & Tom Fawcett, O'Reilly, 2013.