

# **Tecnologías para Inteligencia Artificial (247101009)**

## **Tema 2. Regresión logística y K-NN**

---

Javier Vales Alonso

**Máster Universitario en Ingeniería Telemática**

2020

Universidad Politécnica de Cartagena

## Modelos de clasificación lineal

- Regresión logística

- Regresión logística multi-clase

## Medidas de rendimiento

- Medidas de rendimiento en regresión

- Medidas de rendimiento en clasificación

## Modelos no-paramétricos

- $K$ -NN

## ¿Cómo estudiar esta unidad?

1. Haga una primera lectura de las diapositivas de la unidad. Concéntrese en ver las ideas generales y hacer una primera revisión de las matemáticas.
2. Haga una revisión a fondo de las matemáticas con las diapositivas y resuelva en el notebook los ejercicios indicados. Intente comprender todos los desarrollos involucrados. En caso de dudas, lea las referencias sugeridas (ver referencias en Tema 0) o contacte con el profesor.
3. Finalmente, envíe el notebook a través de AV.

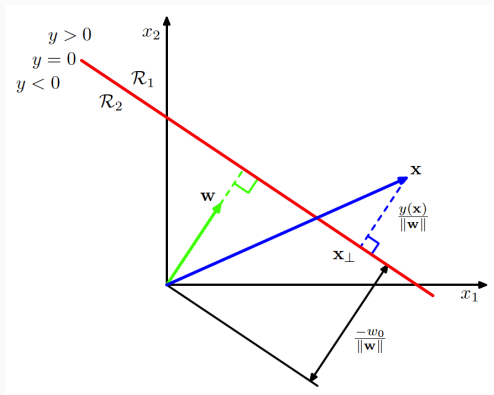
# Modelos de clasificación lineal

---

En los problemas de *clasificación*, el objetivo es asignar a una nueva instancia  $\mathbf{x}$  (vector de entrada  $D$ -dimensional) una de las  $K$  posibles clases  $\mathcal{C}_k$  para  $k=1, \dots, K$ .

Las regiones asignadas a diferentes clases están separadas por *fronteras de decisión*. El término **clasificación lineal** se utiliza cuando estas fronteras de decisión son hiperplanos. Si un conjunto de datos puede separarse sin errores de clasificación por algún hiperplano, diremos que los datos son *separables linealmente*.

## Modelos de clasificación lineal (II)



Una frontera de decisión lineal (rojo) para un espacio de entrada bidimensional es un hiperplano unidimensional (una línea). Si  $D=3$  sería un plano. Un hiperplano  $(D - 1)$ -dimensional está dado por la ecuación  $w_0 + \mathbf{w}^T \mathbf{x} = 0$ , siendo  $\mathbf{x}$  un vector  $D$ -dimensional.

## Modelos de clasificación lineal (III)

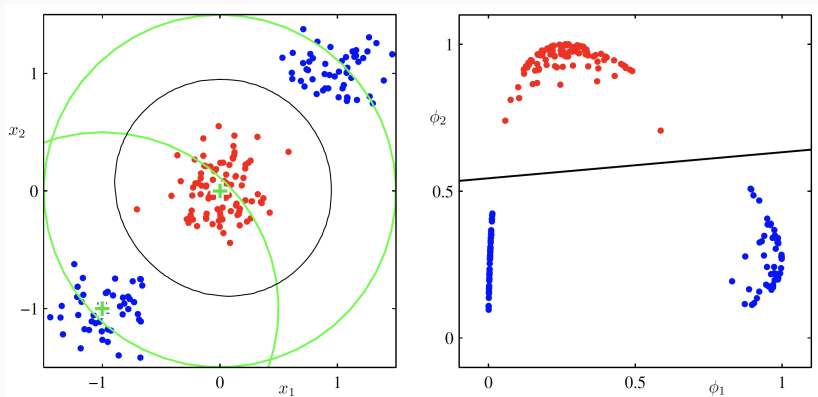
Como en los problemas de regresión lineal, consideraremos habitualmente la transformación de los datos a un espacio de características dado por una base de  $M$  funciones (posiblemente no lineales)  $\{\phi_m(\mathbf{x})\}$ , para  $m = 0, \dots, M-1$  con  $\phi_0(\mathbf{x})=1$ .

Es decir, en notación vectorial, la transformación al espacio de características la escribiremos como:

$$\phi(\mathbf{x}) = (\phi_0(\mathbf{x}), \dots, \phi_{M-1}(\mathbf{x}))^T$$

Si está claro por el contexto escribiremos simplemente  $\phi$  por  $\phi(\mathbf{x})$ . Observe que si trabajamos con esta transformación, aunque los límites de decisión sean lineales en el espacio de características, no lo serán en el espacio de entrada.

## Modelos de clasificación lineal (IV)



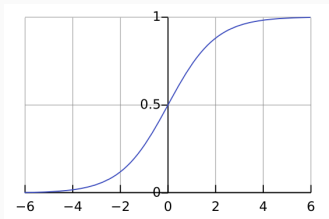
Ejemplo de conjunto de datos lineal no separable, que se puede transformar (utilizando Funciones de base gaussiana  $\phi_1$  y  $\phi_2$ ) desde el espacio de entrada original (izquierda) a un espacio separable de características (derecha).



# Regresión logística

La regresión logística es un **modelo lineal de clasificación binaria** que caracteriza la probabilidad  $p(C_1|\phi)$  usando la función *sigmoide logística* sobre una combinación lineal del vector de características:

$$p(C_1|\phi) = \sigma(\mathbf{w}^T \phi) \quad p(C_2|\phi) = 1 - p(C_1|\phi) \quad (1)$$



$$\sigma(a) = \frac{1}{1 + e^{-a}}$$

$$\frac{d\sigma(a)}{da} = \sigma(a)(1 - \sigma(a))$$

Como se muestra, esta función toma valores en  $[0, 1]$ , lo que hace que sea una buena candidata para modelar una probabilidad. En machine learning, las funciones que, como la sigmoide, asignan una salida tomando como entrada el vector de características se llaman *funciones de activación*.

## Regresión logística (II)

Dado un conjunto de datos etiquetado  $\mathbf{X}=\{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ ,  $\mathbf{t} = \{t_1, \dots, t_N\}$ , con  $t_n=1$  si  $\mathbf{x}_1 \in \mathcal{C}_1$ , o 0 en caso contrario. Si la probabilidad tiene la forma dada en la eq. (1) el modelo depende de  $M$  parámetros  $\{w_m\}$  y la verosimilitud es:

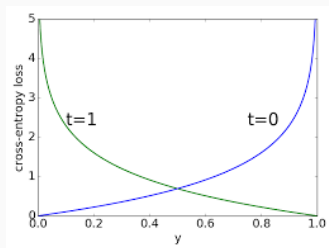
$$p(\mathbf{t}|\mathbf{w}) = \prod_{n=1}^N y_n^{t_n} (1 - y_n)^{1-t_n}$$

Donde  $y_n$  es la predicción  $\sigma(\mathbf{w}^T \phi(\mathbf{x}_n))$ . El logaritmo de la verosimilitud actúa como coste para la regresión logística:

$$J(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{w}) = -\sum_{n=1}^N [t_n \ln y_n + (1 - t_n) \ln(1 - y_n)] \quad (2)$$

Esta función de coste se llama *cross-entropy*, y se usa a menudo para medir la similitud entre dos distribuciones de probabilidad.

## Regresión logística (III)



Cuando la etiqueta  $t_n$  es 1 (clase  $\mathcal{C}_1$ ), la cross-entropy penaliza valores pequeños de  $y_n$  (curva verde). Para datos etiquetados como  $\mathcal{C}_2$  la cross-entropy es alta para valores de  $y_n$  cercanos a 1 (curva azul).

Por lo tanto, el gradiente de la función de error (ver propiedad de la derivada de la función logística en diapositiva 8) es:

$$\nabla J(\mathbf{w}) = \sum_{n=1}^N (y_n - t_n) \phi(\mathbf{x}_n)^T$$

## LRegresión logística (IV)

En regresión lineal los pesos óptimos se pueden calcular con una expresión cerrada. Sin embargo, para regresión logística esto ya no es posible debido a la presencia de la función sigmoide.

Al ser  $J(\mathbf{w})$  concava, tiene un mínimo único, que se puede encontrar eficientemente por métodos iterativos como

Newton-Raphson:

$$\mathbf{w}^{(\text{new})} = \mathbf{w}^{(\text{old})} - \mathbf{H}^{-1} \nabla J(\mathbf{w}^{\text{old}})$$

siendo  $\mathbf{H}$  la Hessiana de  $J(\mathbf{w})$ , esto es,  $(H)_{ij} = \frac{\partial^2 J(\mathbf{w})}{\partial w_i \partial w_j}$

Con la notación matricial, el gradiente y la Hessiana de  $J(\mathbf{w})$  son:

$$\begin{aligned}\nabla J(\mathbf{w}) &= \Phi^T (\mathbf{y} - \mathbf{t}) \\ H = \nabla \nabla J(\mathbf{w}) &= \Phi^T \mathbf{R} \Phi\end{aligned}$$

siendo  $\mathbf{y} = (y_1, \dots, y_N)^T$  y  $\mathbf{R}$  una matriz diagonal  $N \times N$  tal que elemento  $(\mathbf{R})_{nn} = y_n(1 - y_n)$ . Por lo tanto, la regla de actualización para Newton-Raphson es:

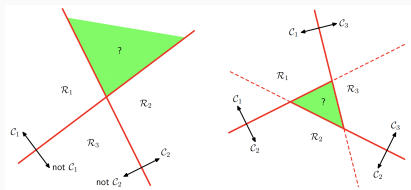
$$\mathbf{w}^{(\text{new})} = \mathbf{w}^{(\text{old})} - (\Phi^T \mathbf{R} \Phi)^{-1} \Phi^T (\mathbf{y} - \mathbf{t})$$

Este método se conoce en la literatura como *Iteratively Reweighted Least Squares* (IRLS).

# Regresión logística multi-clase

Una manera simple de abordar la clasificación multi-clase podría ser considerar colecciones de clasificadores binarios:

- *One-vs-all*. Para cada clase se construye un clasificador binario asumiendo que todos los puntos fuera de la clase dada pertenecen a la clase alternativa.
- *One-vs-one*. Para cada par de clases, un clasificador binario es construido.



Ambos esquemas tienen regiones de incertidumbre (verde)

## Regresión logística multi-clase (II)

Este problema puede evitarse construyendo un modelo multi-clase a través de la **función de activación softmax** que permite probabilidades multi-clase:

$$p(\mathcal{C}_k|\phi) = y_k(\phi, \mathbf{W}) = \frac{e^{a_k}}{\sum_{j=1}^K e^{a_j}} = \frac{e^{\mathbf{w}_k^T \phi}}{\sum_{j=1}^K e^{\mathbf{w}_j^T \phi}}$$

Este modelo esta parametrizado por una colección de vectores de pesos  $\mathbf{W} = \{\mathbf{w}_k\}$ .

## Regresión logística multi-clase (III)

Procediendo como en el caso binario, la función de error se puede expresar como:

$$J(\mathbf{w}_1, \dots, \mathbf{w}_K) = -\ln p(\mathbf{T} | \mathbf{w}_1, \dots, \mathbf{w}_K) = -\sum_{n=1}^N \sum_{k=1}^K (\mathbf{t}_n)_k \ln(\mathbf{y}_n)_k$$

que se llama cross-entropy (multi-clase)<sup>1</sup>.

Entonces, los gradientes con respecto a cada  $\mathbf{w}_k$  son:

$$\nabla_{\mathbf{w}_j} J(\mathbf{w}_1, \dots, \mathbf{w}_K) = -\sum_{n=1}^N [(\mathbf{y}_n - \mathbf{t}_n)_j] \phi(\mathbf{x}_n)$$

---

<sup>1</sup>Aquí el target se expresa en notación *1-de-K*, e.g., si  $K=5$  entonces  $\mathbf{t}=(0, 1, 0, 0, 0)^T$ , e  $\mathbf{y}_n$  denota el vector  $(y_1(\phi(\mathbf{x}_n)), \dots, y_K(\phi(\mathbf{x}_n)))^T$ .



## Medidas de rendimiento

---

## Medidas de rendimiento en regresión

Antes de continuar, es importante comentar las medidas de rendimiento comunes utilizadas en la literatura de ML.

Para los problemas de regresión vimos que el error cuadrático (*squared error*) se usaba para definir la función de error. Como medida de rendimiento adicional se suele proporcionar el error cuadrático medio (*mean squared error*), definido simplemente como:

$$\text{MSE} = \frac{1}{N} \sum_{n=1}^N (y(\mathbf{x}_n) - t_n)^2 \quad (3)$$

o su raíz (RMSE) para trabajar con un error en las mismas unidades que la salida del regresor

Además, también son posibles otras distancias en lugar de las euclideas (ver diapositiva 24 para algunos ejemplos).

La evaluación de los clasificador es rica en opciones. Además de la cross-entropy discutida anteriormente, se emplea habitualmente la *accuracy*. Se define como la ratio de instancias bien clasificadas entre el número total de instancias (para un conjunto de pruebas dado).

Observe que **para un clasificador de  $K$  clases, la *accuracy* mínima es  $1/K$**  (la obtenida por un clasificador puramente aleatorio).

## Medidas de rendimiento en clasificación (II)

Si el conjunto de prueba está sesgado (por ejemplo, si el 90 % de las instancias pertenece a la clase 1 y 10 % a la clase 2), la *accuracy* es una pésima elección: **un clasificador trivial que siempre elige la clase 1 tendrá una *accuracy* del 90 %.**

Una manera más adecuada de evaluar un clasificador es a través de su **matriz de confusión** que muestra para la fila  $i$  y columna  $j$  el ratio de instancias que pertenece a la clase  $i$  y han sido etiquetadas como clase  $j$ . La diagonal da la proporción de clasificación correcta para cada clase.

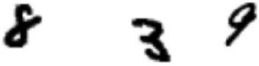


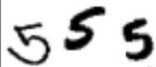
A partir de la matriz de confusión se obtienen varias medidas de rendimiento adicionales. La siguiente diapositiva muestra una figura resumen.

# Medidas de rendimiento en clasificación (III)

		True condition			
Total population		Condition positive	Condition negative	Prevalence = $\frac{\sum \text{Condition positive}}{\sum \text{Total population}}$	Accuracy (ACC) = $\frac{\sum \text{True positive} + \sum \text{True negative}}{\sum \text{Total population}}$
Predicted condition	Predicted condition positive	True positive	False positive, Type I error	Positive predictive value (PPV), Precision = $\frac{\sum \text{True positive}}{\sum \text{Predicted condition positive}}$	False discovery rate (FDR) = $\frac{\sum \text{False positive}}{\sum \text{Predicted condition positive}}$
	Predicted condition negative	False negative, Type II error	True negative	False omission rate (FOR) = $\frac{\sum \text{False negative}}{\sum \text{Predicted condition negative}}$	Negative predictive value (NPV) = $\frac{\sum \text{True negative}}{\sum \text{Predicted condition negative}}$
		True positive rate (TPR), Recall, Sensitivity, probability of detection, Power = $\frac{\sum \text{True positive}}{\sum \text{Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm = $\frac{\sum \text{False positive}}{\sum \text{Condition negative}}$	Positive likelihood ratio (LR+) = $\frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) = $\frac{\text{LR+}}{\text{LR-}}$  $F_1 \text{ score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$
		False negative rate (FNR), Miss rate = $\frac{\sum \text{False negative}}{\sum \text{Condition positive}}$	Specificity (SPC), Selectivity, True negative rate (TNR) = $\frac{\sum \text{True negative}}{\sum \text{Condition negative}}$	Negative likelihood ratio (LR-) = $\frac{\text{FNR}}{\text{TNR}}$	

Medidas de rendimiento para clasificadores binarios. En la literatura de ML, las más usadas son el *precision* (inglés) y el *recall*.

## Medidas de rendimiento en clasificación (IV)

		Predicted	
		Negative	Positive
Actual	Negative		
	Positive		

**TN** (True Negative) is indicated by a green bubble pointing to the top-left cell (Actual Negative, Predicted Negative).

**FP** (False Positive) is indicated by a pink bubble pointing to the top-right cell (Actual Negative, Predicted Positive).

**FN** (False Negative) is indicated by a pink bubble pointing to the bottom-left cell (Actual Positive, Predicted Negative).

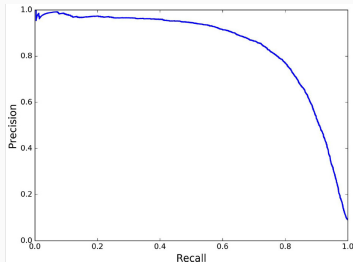
**TP** (True Positive) is indicated by a green bubble pointing to the bottom-right cell (Actual Positive, Predicted Positive).

**Precision** (e.g., 3 out of 4) is indicated by an upward arrow pointing to the Positive Predicted column.

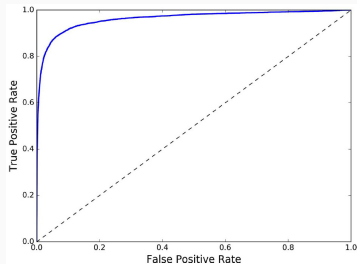
**Recall** (e.g., 3 out of 5) is indicated by a leftward arrow pointing to the Positive Actual row.

Ejemplo de cálculo del *precision* y del *recall* para un clasificador binario para un ejemplo con el data set del MNIST orientado a detectar las clases '5' y 'no 5'.

# Medidas de rendimiento en clasificación (V)



Los clasificadores tienen un compromiso entre el *precision* y el *recall* (un *precision* alto está asociado a un bajo el *recall* y viceversa), Este tipo de gráfico se llama *curva de precision vs. recall*.



La **curva característica de funcionamiento del receptor (ROC)** muestra el *recall* versus la tasa de falsos positivos. Como se puede ver también hay un balance entre ambos.

# Modelos no-paramétricos

---



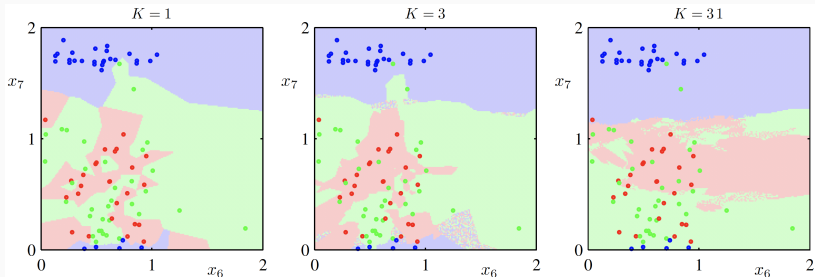
## Modelos no-paramétricos

Los modelos que hemos estudiado establecen una hipótesis que depende de un conjunto de pesos  $\mathbf{w}$  que se determinan con la minimización de una función de error.

Existe también la posibilidad de construir **predictores no paramétricos**. Éstos suelen ser **modelos basados en instancias** (ver Tema 0), cuya operación consiste en comparar mediante alguna función el punto a clasificar/predecir con un conjunto (parcial o total) de los puntos del data set.

Ahora estudiaremos el método llamado *K*-Nearest neighbors (*K*-NN) que emplea como función de comparación la distancia entre instancias. En el Tema 5 estudiaremos el uso de *kernels* para obtener predictores basados en instancias para modelos lineales.

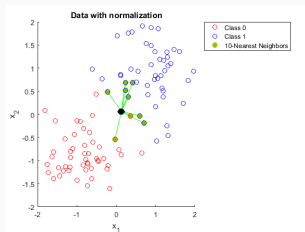
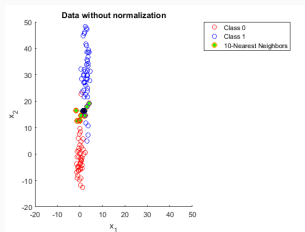
$K$ -NN es un método de clasificación **no lineal** que asigna a  $\mathbf{x}$  la clase con el voto mayoritario entre sus  $K$  vecinos más cercanos.



Regiones de decisión que usan  $K$ -NN con  $K=1, 3$  y  $31$ . Los límites de decisión no son lineales. La clase  $\mathcal{C}_k$  es seleccionado como el que maximiza la probabilidad posterior  $\frac{K_k}{K}$ .

### Notas de implementación:

- Distancias no euclideas se usan con frecuencia, por ejemplo, Manhattan o Cosine (ver [link](#)).
- Para evitar efectos de escalado no uniforme entre las distintas características (ver figura), es obligatorio para realizar un proceso de **normalización** al data set.



### Notas de implementación:

- Si la dimensionalidad del espacio de características es alta se recomienda usar alguna técnica de reducción de dimensionalidad (Tema 4).
- $K$ -NN es sensible a la presencia de *outliers*.
- $K$ -NN también se usa en regresión (por ejemplo, calculando un promedio de los *targets* de los vecinos ponderados por las distancias al punto a predecir).
- Al ser un método basado en instancias para clasificar un nuevo punto, **es necesario calcular la distancia a TODOS los puntos del conjunto de entrenamiento**, lo que obliga a almacenarlos y a una predicción que puede ser lenta.