

ACÀMICA

TEMA DEL DÍA

# Evaluación de modelos

Volveremos a ver evaluación de modelos, incorporando una técnica muy importante: Validación Cruzada.

También, estudiaremos una nueva herramienta para evaluar modelos de Clasificación: la Curva ROC



# Agenda

---

Daily

Explicación: Curva ROC

**Break**

Hands-on training

Cierre



# Daily



Daily



## Sincronizando...

### Bitácora



¿Cómo te ha ido?  
¿Obstáculos?  
¿Cómo seguimos?

### Challenge



¿Cómo te ha ido?  
¿Obstáculos?  
¿Cómo seguimos?

# Repaso de la bitácora



# Machine Learning

El Aprendizaje Automático (o Machine Learning) se dedica al estudio de los programas que aprenden a realizar una **tarea** en base a la experiencia.



- La tarea está asociada a una **función objetivo** desconocida. Por ejemplo, “separar puntos naranjas de azules”, “detectar spam” o “calcular el precio de una propiedad”.
- Un programa *aprende* una tarea, si su **performance mejora con la experiencia**.

# Aprendizaje supervisado

Dada una función objetivo desconocida, queremos **aproximarla** mediante un modelo.



- En los problemas de Clasificación, buscamos **aproximar** la/s frontera/s que separan nuestras clases.
- En los problemas de Regresión, buscamos **aproximar** la curva que generó nuestros datos.



# Aprendizaje supervisado

Dada una función objetivo  $\mathbf{f}$  desconocida, queremos aproximarla mediante un modelo, que se suele notar  $\mathbf{f}^1$ .

## Entrenar un modelo

→  
*consiste en*

ajustar sus parámetros (encontrar valores óptimos) dado un conjunto de datos.

Los algoritmos de aprendizaje automático son procedimientos para entrenar modelos a partir de un conjunto de datos

<sup>1</sup>El sombrerito va a arriba de la  $\mathbf{f}$  pero no encontramos cómo escribirlo.

Antes de continuar, un comentario acerca de la nomenclatura...

## PARÁMETRO

Son características de nuestro modelo que el algoritmo de entrenamiento aprende automáticamente. **Ejemplo:** pendiente y ordenada al origen en una regresión lineal, umbrales que usa un árbol de decisión para partir una rama, etc.

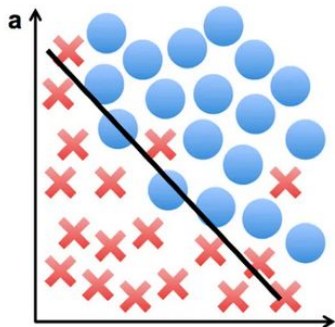
## HIPERPARÁMETRO

Son características de nuestro flujo de trabajo en Machine Learning (que incluye al modelo) que debemos elegir bajo algún criterio. En general, el criterio es mejor performance estadística (¡aunque podría no serlo!).

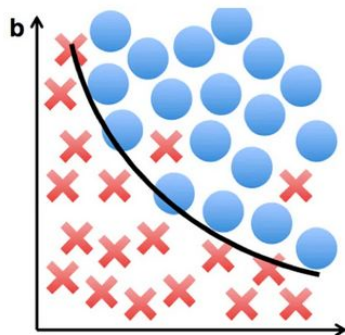
A veces, usamos “parámetro” para referirnos a los hiperparámetros, pero es importante prestar atención si es algo que se “aprende” o se “decide”.

En Scikit-Learn, los hiperparámetros los elegimos cuando creamos un modelo, mientras que los parámetros recién pueden ser accedidos luego de entrenar el modelo.

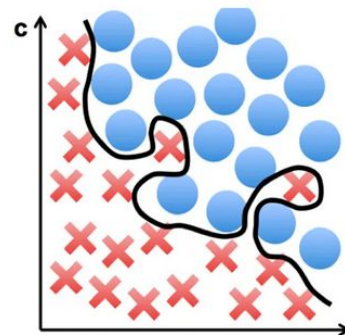
## Pero no es tan fácil... Recordemos estas tres situaciones



El **modelo a** es muy simple y no reproduce correctamente la frontera entre las clases. Llamaremos **underfitting** a esta situación.

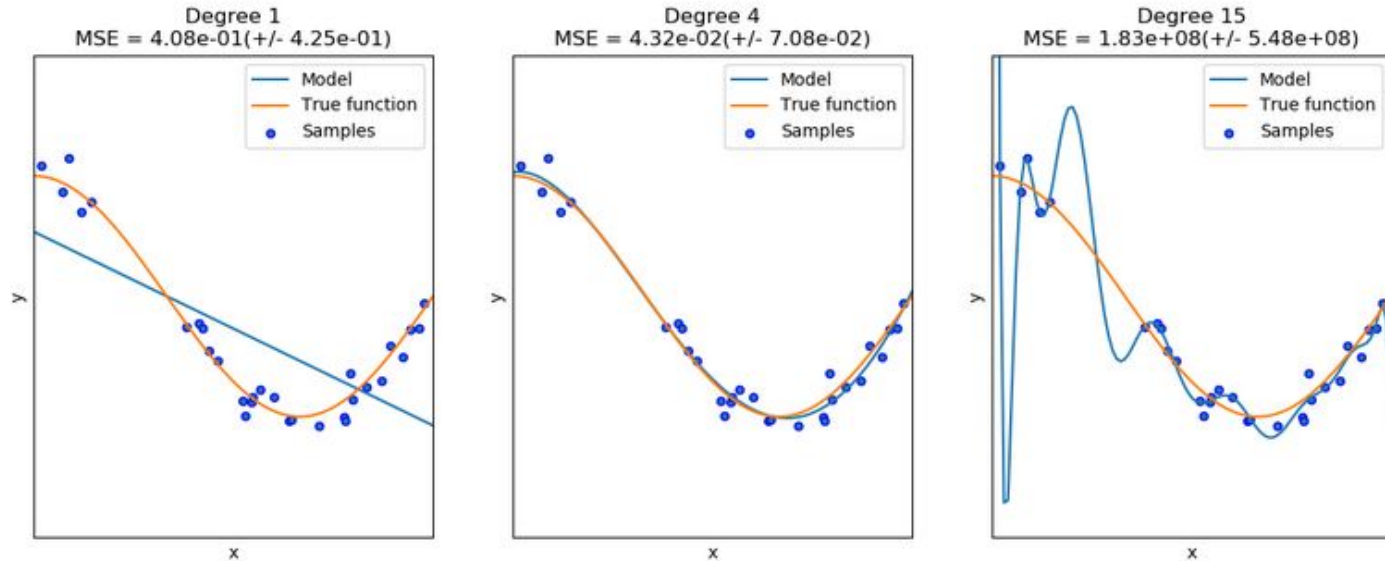


El **modelo b** tiene la complejidad suficiente para encontrar una frontera que parece apropiada para estos datos.



El **modelo c** parece muy flexible y se adaptó demasiado a los datos con los que fue entrenado. Llamaremos **overfitting** a esta situación.

# No es solamente un problema de los modelos de Clasificación



Fuente: [https://scikit-learn.org/stable/auto\\_examples/model\\_selection/plot\\_underfitting\\_overfitting.html#](https://scikit-learn.org/stable/auto_examples/model_selection/plot_underfitting_overfitting.html#)

# ¿Cómo podemos evaluar si el modelo está *aprendiendo* o no de nuestros datos?

Una forma práctica de evaluar si nuestro modelo aprendió o no de nuestro datos es **observar su desempeño frente a nuevas instancias.**

En nuestro flujo de trabajo, tendremos que emular una situación donde el modelo es entrenado con ciertos datos y luego es evaluado con datos nuevos.

## Train/Test Split

1. Separo los datos en dos conjuntos, Train y Test.
2. Entreno con los datos de Train
3. Evalúo el desempeño del modelo los datos de Test.

Evaluar el desempeño sobre el conjunto de Test tiene varios usos:

- 1. Obtenemos una evaluación realista del desempeño de nuestros modelos.**
- 2. Nos permite seleccionar el modelo que mejor desempeña sobre nuestros datos<sup>1</sup>.**

**Pero...**

<sup>1</sup> Veremos mejor esto en el siguiente encuentro.

Evaluar el desempeño sobre el conjunto de Test tiene varios usos:

- 1. Obtenemos una evaluación realista del desempeño de nuestros modelos.**
- 2. Nos permite seleccionar el modelo que mejor desempeña sobre nuestros datos<sup>1</sup>.**

**Pero...**

**¿Y si tuvimos suerte al separar los datos?**

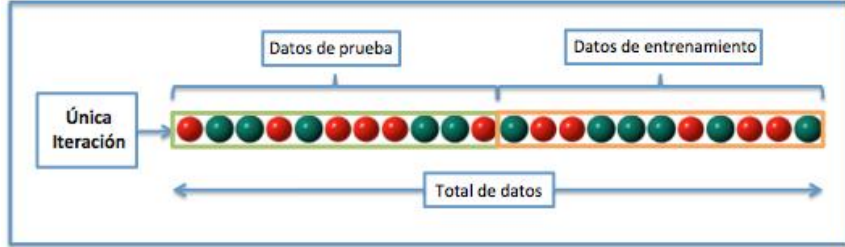
<sup>1</sup> Veremos mejor esto en el siguiente encuentro.

# Validación Cruzada





# Hasta ahora teníamos



Entrenamos  
modelos con  
datos de  
entrenamiento

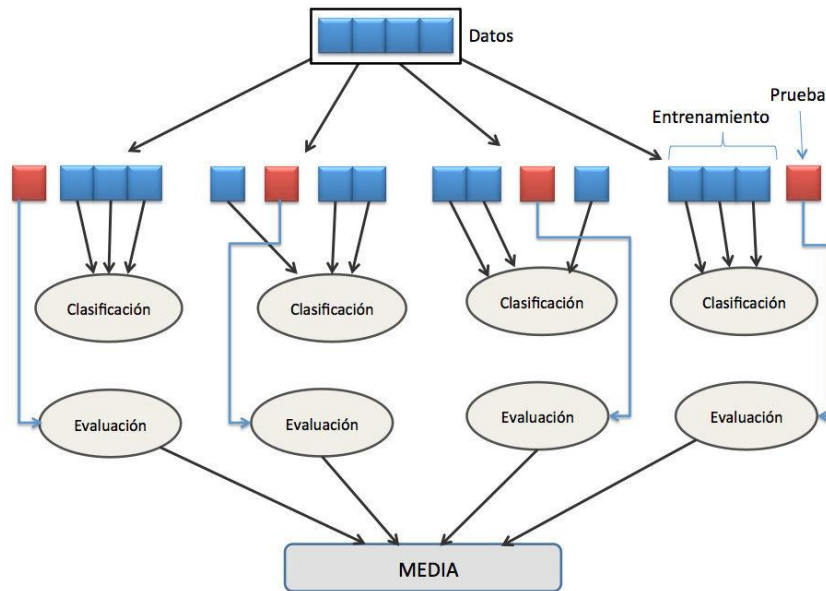
Performance  
sobre datos de  
prueba

El objetivo de la validación cruzada es obtener una evaluación de performance de nuestro modelo que sea independiente de la partición en entrenamiento y prueba de los datos.

¿Y cómo lo hacemos?



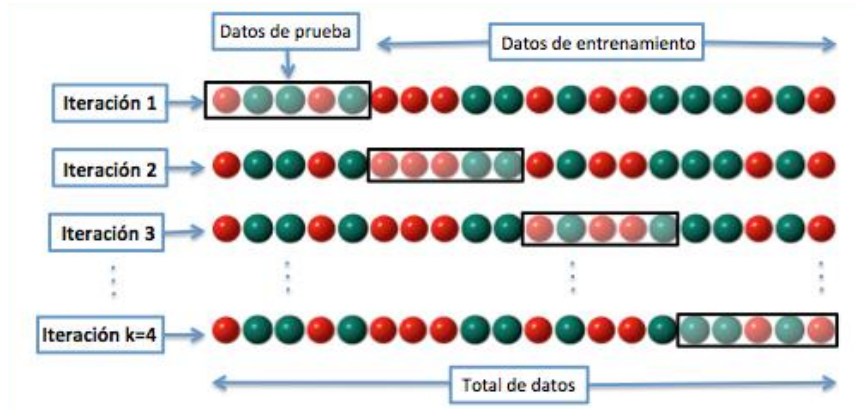
# ¡Haciendo muchas particiones!



De esta forma,  
esperamos que la medida  
de performance sea  
independiente de la  
partición de los datos

# k-fold Cross Validation

Existen diferentes técnicas de validación cruzada.  
La más conocida común es k-fold Cross Validation:



Modelo → Performance 1

Modelo → Performance 2

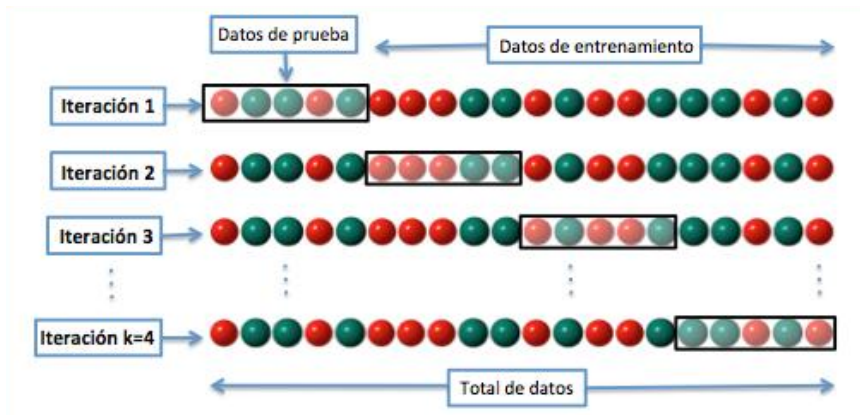
Modelo → Performance 3

Modelo → Performance k = 4

Promedio

# k-fold Cross Validation

Existen diferentes técnicas de validación cruzada.  
La más conocida común es k-fold Cross Validation



Modelo → Performance 1

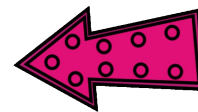
Modelo → Performance 2

Modelo → Performance 3

Modelo → Performance  $k = 4$

Promedio

**Notar que en k-fold CV cada dato aparece una vez en los datos de prueba y  $k-1$  en los datos de entrenamiento.**



# k-fold Cross Validation

- 1) Desordenar los datos
- 2) Separar en K folds (muestras) del mismo tamaño
- 3) Para cada fold que separamos:
  - a) Elegir la fold como Test set, y las K-1 folds restantes como Train set.
  - b) Entrenar y evaluar el modelo.
  - c) Guardar el resultado de la evaluación y descartar el modelo.
- 4) Obtener una medida de performance del modelo como el promedio de las K evaluaciones obtenidas en (3). También es una buena práctica incluir una medida de la varianza de las métricas obtenidas (**¿Por qué?**).

# k-fold Cross Validation

- 1) Desordenar los datos
- 2) Separar en K folds (muestras) del mismo tamaño
- 3) Para cada fold que separamos:
  - a) Elegir la fold como Test set, y las K-1 folds restantes como Train set.
  - b) Entrenar y evaluar el modelo.
  - c) Guardar el resultado de la evaluación y descartar el modelo.
- 4) Obtener una medida de performance del modelo como el promedio de las K evaluaciones obtenidas en (3). También es una buena práctica incluir una medida de la varianza de las métricas obtenidas **(¿Por qué?)**.

¿Qué ocurre cuando  $k = \text{número de datos}$ ?



# k-fold Cross Validation

- La validación cruzada es un **procedimiento de remuestreo** que se utiliza para evaluar modelos de aprendizaje automático en una muestra de datos limitada.
- El **(hiper)parámetro más importante es k** que se refiere al número de grupos en que se dividirá una muestra de datos dada.
- Es un **método popular porque es fácil de entender** y porque generalmente resulta en una **estimación *menos sesgada* o menos optimista** de la habilidad del modelo que otros métodos, como una simple división de train / test.
- **¡No siempre hay que separar al azar!** En algunos casos (por ejemplo, predicción con series de tiempo), la validación cruzada toma otra forma.
- La validación cruzada está **íntimamente relacionada con la optimización de hiperparámetros**, que veremos en el siguiente encuentro.



¡CV es tan importante que viene en todos los entornos de desarrollo de Machine Learning!

## En Scikit-Learn:

- [Documentación](#) en Scikit-Learn sobre Validación Cruzada.
- [Funciones](#) incorporadas en el módulo *model\_selection*

**`sklearn.model_selection.KFold`**

```
class sklearn.model_selection. KFold (n_splits='warn', shuffle=False, random_state=None)
```

[\[source\]](#)



## CHALLENGE BITÁCORA



¡Muéstranos qué hiciste!

¿Qué cosas te costaron más del ejercicio? ¿Cómo las resolviste?

¿Cuál el principal aprendizaje que te llevas?

Si tuvieras que hacerle alguna recomendación a alguien que va a hacer el ejercicio por primera vez, ¿qué le dirías?

- Existen dataset *desbalanceados*, es decir, conjuntos de datos donde hay clases sobrerrepresentadas y otras subrepresentadas. Por ejemplo, el dataset de Iris es un dataset perfectamente balanceado (50 instancias de cada clase), mientras que el dataset del Titanic está levemente desbalanceado. Trabajar con datasets muy desbalanceados no es tarea sencilla, y existen muchas técnicas para abordarlos. Reflexiona sobre qué ocurre con la exactitud, la precisión y la exhaustividad cuando las utilizas para evaluar desempeño sobre dataset binario muy desbalanceado (supongamos, proporción 1 en 1000), en particular cuando te interesa la clase minoritaria. ¿Qué ocurrirá con la Curva ROC y la AUC-ROC?
- ¿Resolviste el Challenge del notebook?

## CHALLENGE BITÁCORA



¿Alguien hizo algo diferente que quiera mostrar?



Un **dataset balanceado** es aquel que tiene - aproximadamente - la misma proporción de instancias de cada clase.  
Por ejemplo, en el caso, binario, alrededor de 50:50 (1:1) de cada clase.

Un **dataset desbalanceado** - en el caso binario - es aquel que tiene muchas instancias de una clase y muy pocas de la otra, dificultando el entrenamiento.  
Por ejemplo, 80:20, 90:10, 99:1, y peor.

**Un poco de desbalance** de clases es esperable, y no afecta a nuestro análisis.

Pero en algunas áreas suelen haber datasets muy desbalanceados:

- Detección de fraudes
- Diagnóstico médico
- Deforestación

Cuando trabajemos con estos datasets, tenemos que tener cuidado con:

- Cómo entrenamos nuestros modelos.
- Qué métricas usamos para evaluarlo.



Cuando trabajemos con estos datasets, tenemos que tener cuidado con:

- Cómo entrenamos nuestros modelos.
- Qué métricas usamos para evaluarlo.

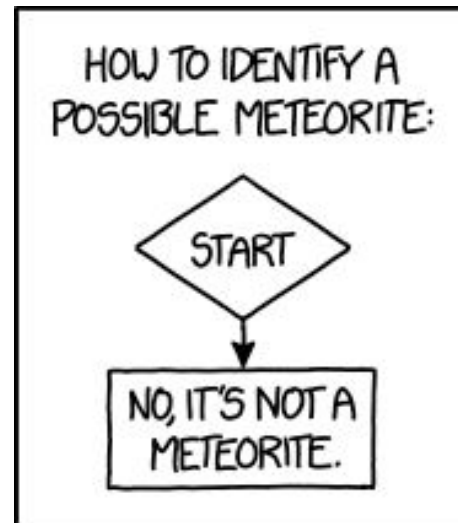
## WARNING

Uno de los malos de la película:

La paradoja de la exactitud (suena mejor en inglés, Accuracy Paradox)

A medida que el desbalance de clases es mayor, la exactitud aumenta, por más que nuestro modelo sea muy malo.

**¿Por qué?** <https://xkcd.com/1723/> \*



## Algunas técnicas para trabajar correctamente con estos datasets:

1

¿Podemos **recolectar nuevos datos**?

---

2

Elegir la **métrica de performance** apropiada para nuestro problema (¡Olvidarse de Exactitud!). Matriz de Confusión, Precisión y Exhaustividad (recall) suelen ser las primeras opciones, pero hay más. ¿Un Falso Positivo tiene el mismo costo que un Falso Negativo?

---

3

**Resamplear** el dataset.

- a. Oversampling: generar nuevas instancias de la clase minoritaria, ya sea copiando instancias preexistentes, o generando instancias sintéticas (ver SMOTE).
  - b. Undersampling: eliminar instancias de la clase sobrerrepresentada.
- 

4

**Probar diferentes modelos** (modelos de ensamble suelen ser buenos) y/o agregarle peso a la clase subrepresentada (fácil desde Scikit-Learn).

---

5

Las **opciones no se terminan acá**. Es un área de continuo desarrollo. Para tener en cuenta: One-Class classification.

# Scores



# Clasificación Binaria • Matriz de confusión

		Clase Predicha	
		Clase Positiva	Clase Negativa
Clase Verdadera	Clase Positiva	TP	FN
	Clase Negativa	FP	TN

¡Tiene toda la información  
que necesitamos!

# Clasificación Binaria • Matriz de confusión

		Clase Predicha	
		Clase Positiva	Clase Negativa
Clase Verdadera	Clase Positiva	TP	FN
	Clase Negativa	FP	TN

¡Tiene **CASI** toda la información que necesitamos!

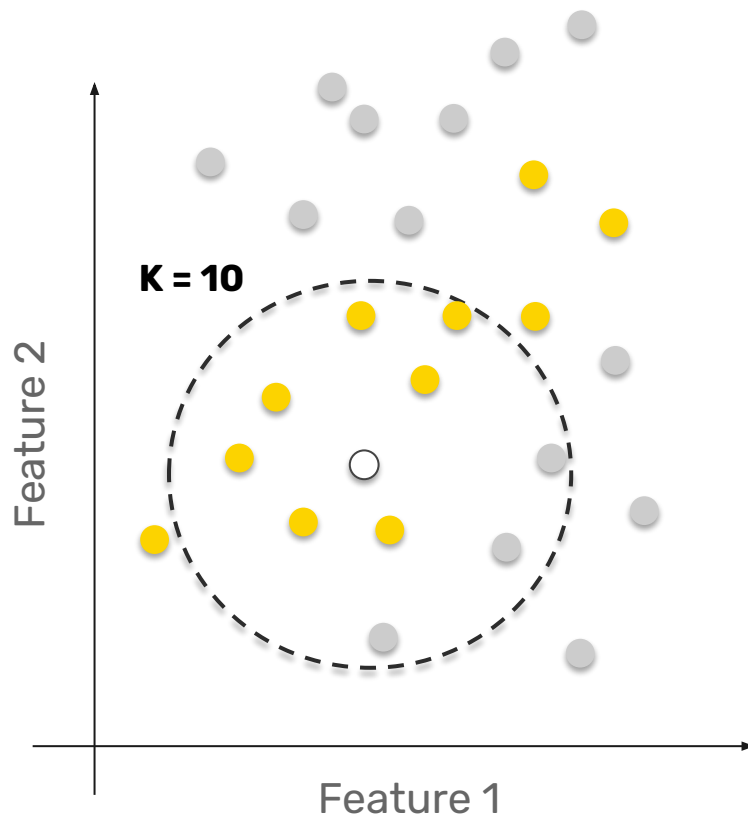
# Clasificación Binaria • Score

Pero, ¿cómo llegamos a los aciertos (TPs), Falsos Positivos, etc.?

Por ejemplo, pensemos en un modelo de vecinos más cercanos, con  $K = 10$ .

En este caso, de 10 vecinos, 7 son amarillos, por lo que la etiqueta correspondiente sería amarilla.

Lo mismo ocurrirá cuando haya más de 5 vecinos de color amarillo. Si, en cambio, hay menos de 5 vecinos de color amarillo, la etiqueta pasa a ser gris.

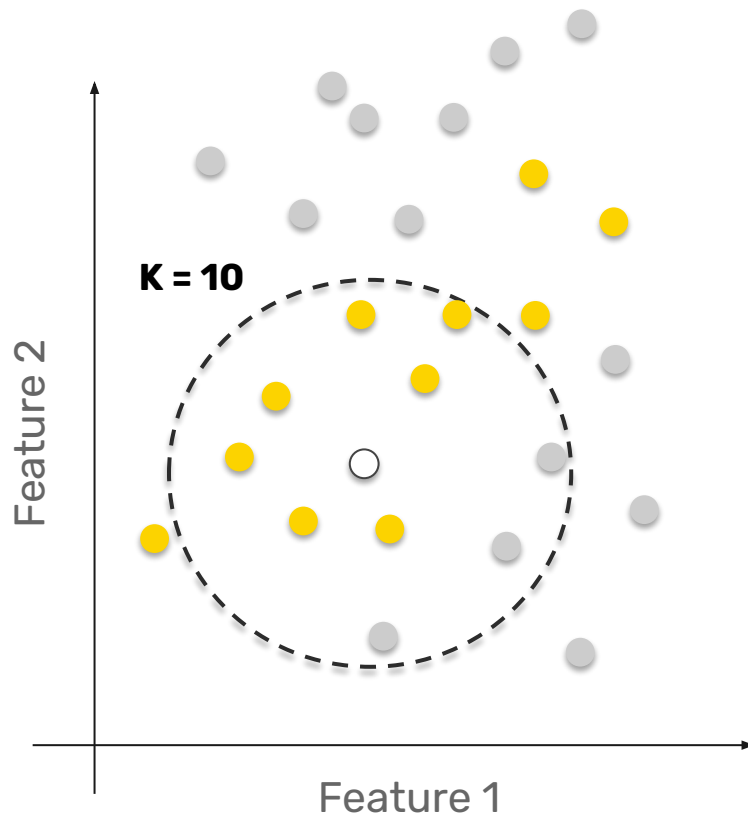


## Clasificación Binaria • Score

Entonces, de una instancia que tiene 10 vecinos amarillos, podemos estar más *seguros* que la etiqueta correspondiente es amarilla que una instancia que solamente tiene 6 vecinos.

Cuando miramos solamente las etiquetas asignadas, esta información la perdemos.

¿Qué podemos hacer?



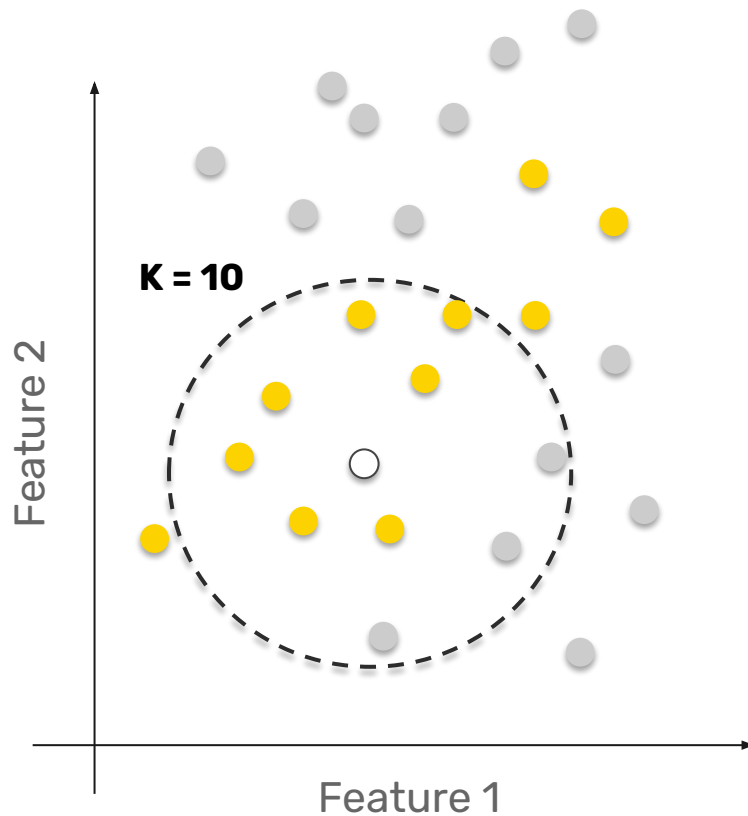
# Clasificación Binaria • Score

Entonces, de una instancia que tiene 10 vecinos amarillos, podemos estar más *seguros* que la etiqueta correspondiente es amarilla que una instancia que solamente tiene 6 vecinos.

Cuando miramos solamente las etiquetas asignadas, esta información la perdemos.

¿Qué podemos hacer?

**¡Generar un Score que represente cuán seguro está el modelo de la etiqueta!**





# Clasificación Binaria • Score

Este razonamiento se puede hacer con (casi) todos los modelos que usemos. En el fondo, lo que un modelo hace para asignar etiquetas es generar un score y poner el umbral *a la mitad*.

**Para pensar:** ¿cómo generan los scores los árboles de decisión?



Como siempre, no lo tenemos que programar.  
En Scikit-Learn, todos los modelos vienen con un método, `predict_proba(X)`, que calcula los scores.

**Importante:** si bien a primer orden estos scores pueden ser interpretados como probabilidades, la realidad es que no lo son, porque no están **Calibrados**. Lo que sí podemos usar son los **Rankings** que generan.

¿Podemos usar estos Scores para caracterizar mejor el desempeño de nuestro modelo y, además, tomar mejores decisiones?

# Curva ROC



# CURVA ROC

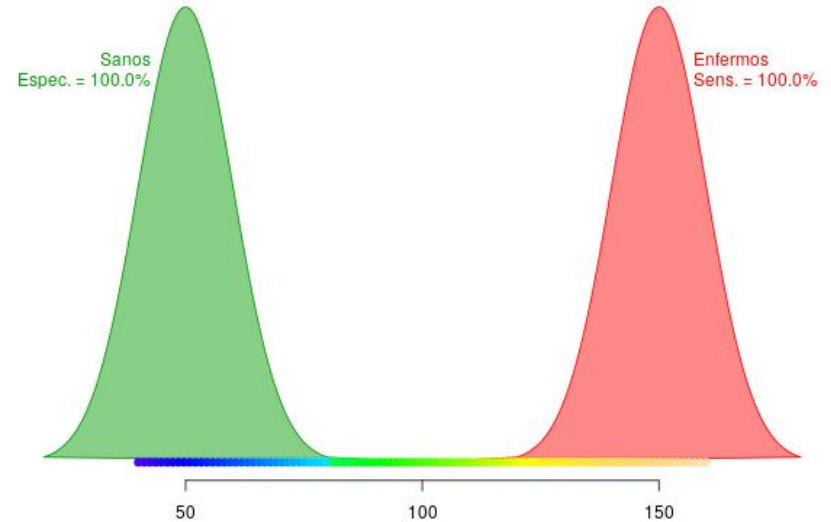
AUC



# Umbrales

¿Qué ocurre si usamos un umbral y asignamos etiquetas según el Score sea superiores o inferiores al valor elegido?

**Hasta ahora, hacemos esto:**

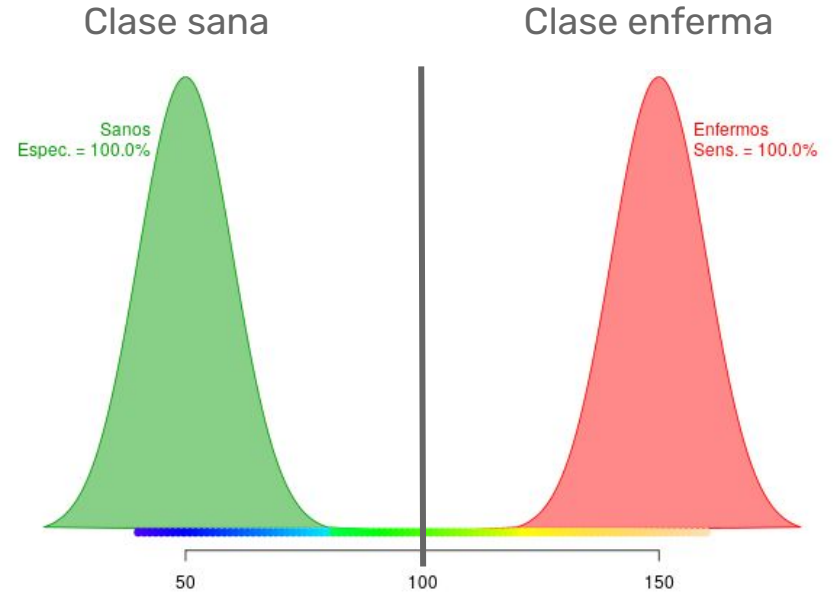


Output (Score) de un modelo de clasificación

# Umbrales

¿Qué ocurre si usamos un umbral y asignamos etiquetas según el Score sea superiores o inferiores al valor elegido?

**Hasta ahora, hacemos esto:**



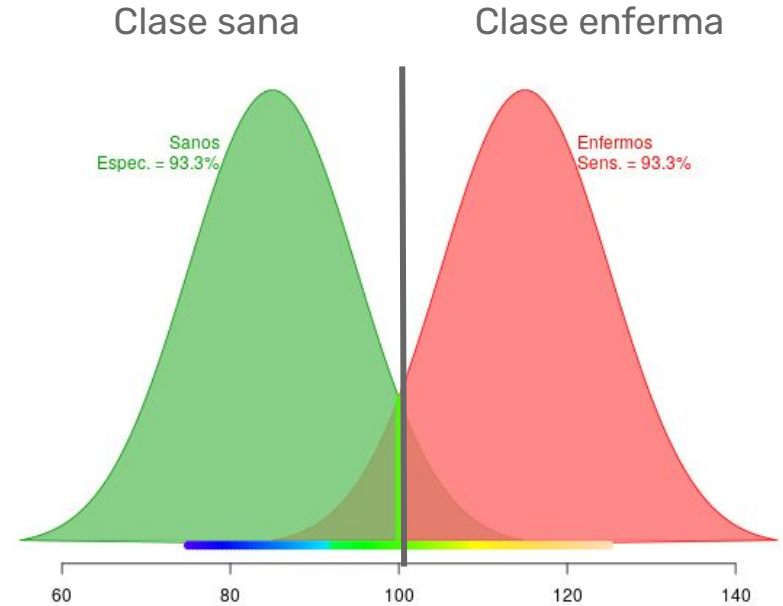
Output (Score) de un modelo de clasificación

Fuente y para jugar un poco: <https://www.bioestadistica.uma.es/analisis/roc1/>

# Umbrales

¿Qué ocurre si usamos un umbral y asignamos etiquetas según el Score sea superiores o inferiores al valor elegido?

**Hasta ahora, hacemos esto:**

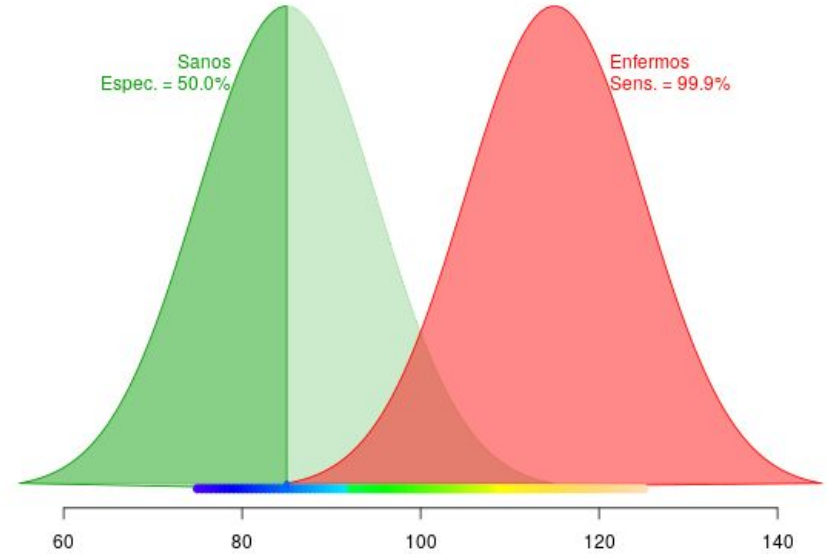


Output (Score) de un modelo de clasificación

Fuente y para jugar un poco: <https://www.bioestadistica.uma.es/analisis/roc1/>

# Umbrales

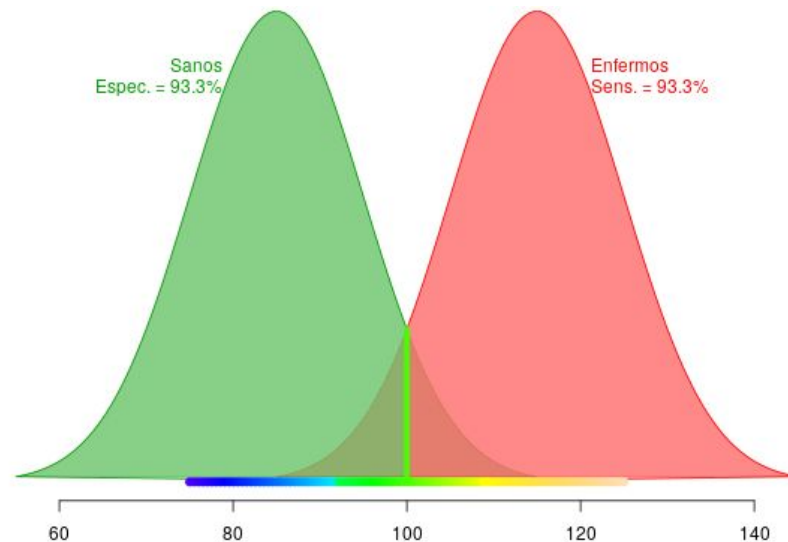
Pero, para nuestro problema,  
tal vez sea mejor hacer esto:





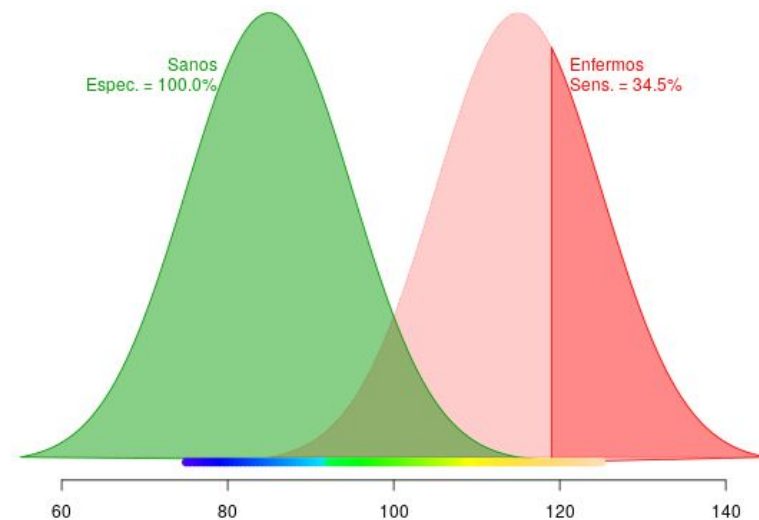
# Umbrales

...o esto (mismo caso que antes):

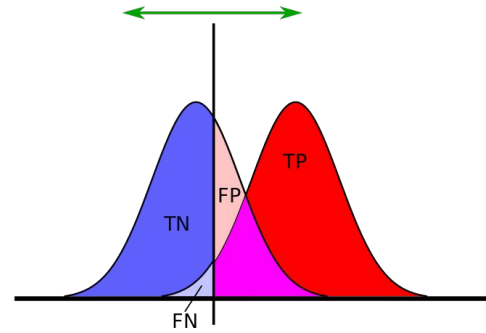


# Umbrales

...o esto:

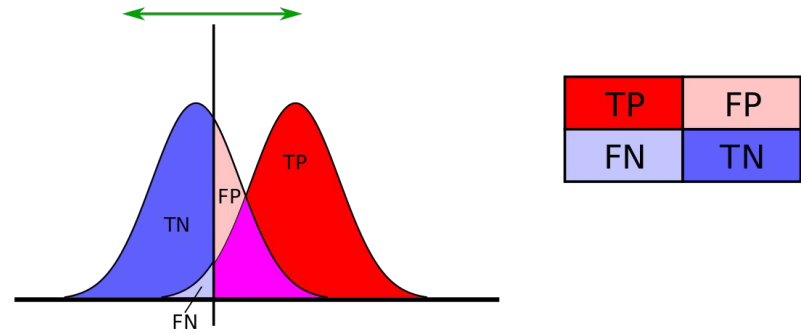


A medida que movemos el umbral, vamos cambiando la cantidad de Aciertos, Falsos positivos, Falsos Negativos y Verdaderos Negativos.



TP	FP
FN	TN

A medida que movemos el umbral, vamos cambiando la cantidad de Aciertos, Falsos positivos, Falsos Negativos y Verdaderos Negativos.



**Podemos cuantificar con:**

$$\text{TPR} = \frac{\text{TP}}{P} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

True Positive Rate → Es la exhaustividad

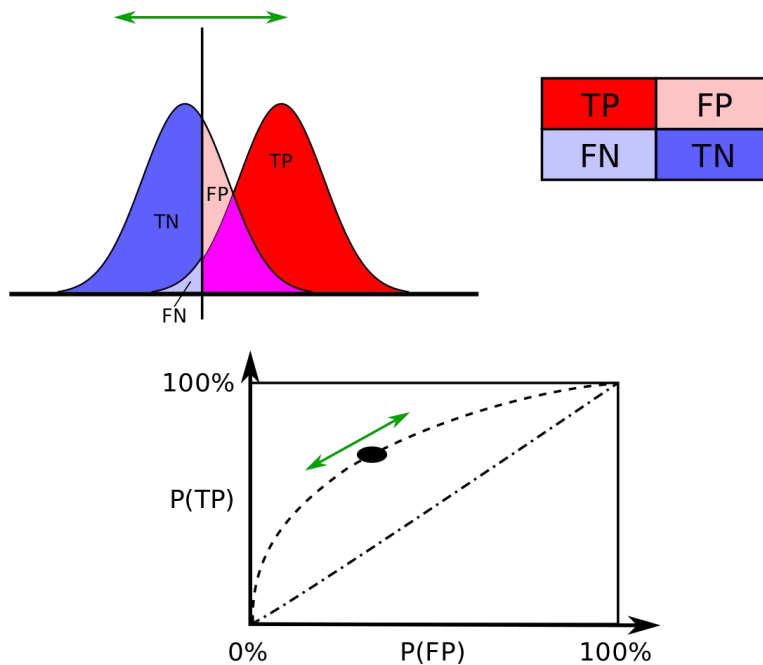
$$\text{FPR} = \frac{\text{FP}}{N} = \frac{\text{FP}}{\text{FP} + \text{TN}}$$

False Positive Rate → NO es la precisión

¡Y hacer una curva de uno contra el otro a medida que variamos el umbral!

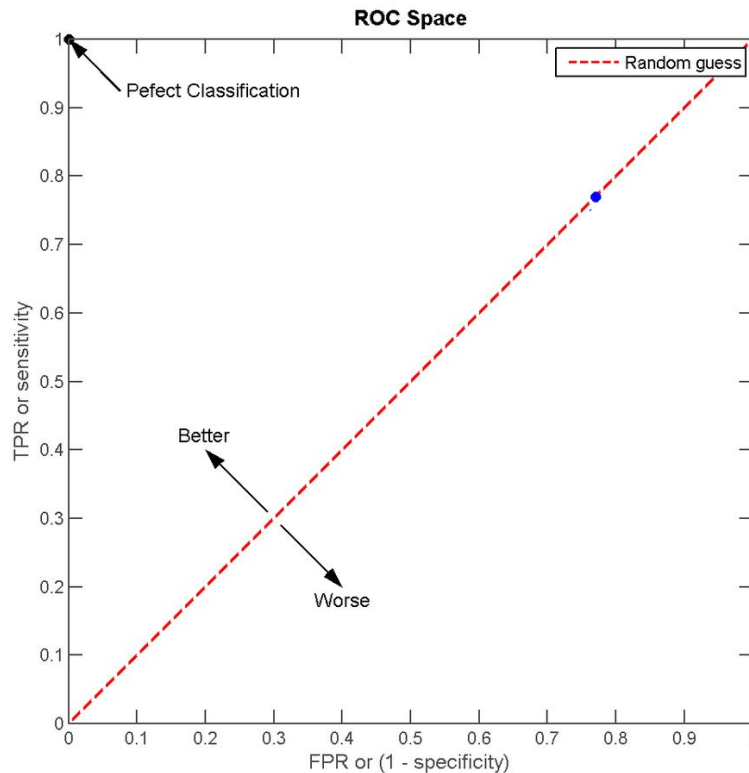
# Curva ROC

La curva ROC es la representación del TPR vs. el FPR para cada valor de corte.



# Curva ROC

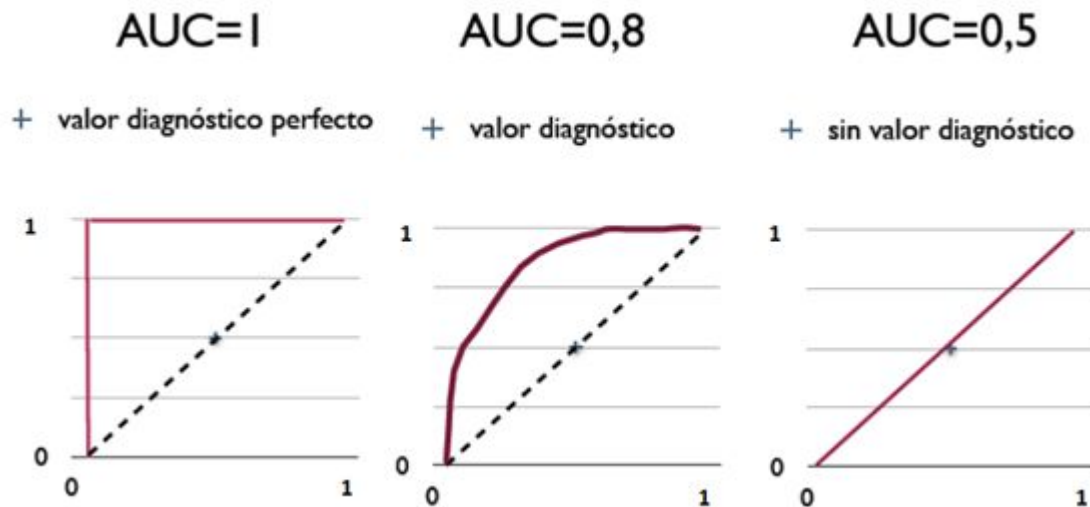
¿Cómo interpretamos  
la calidad de las  
curvas ROC?



# Curva ROC • Área bajo la curva ROC (AUC ROC)

¿Y si queremos cuantificar? La medida de 'cuán buena' es la curva ROC es calculado el área bajo la curva (AUC).

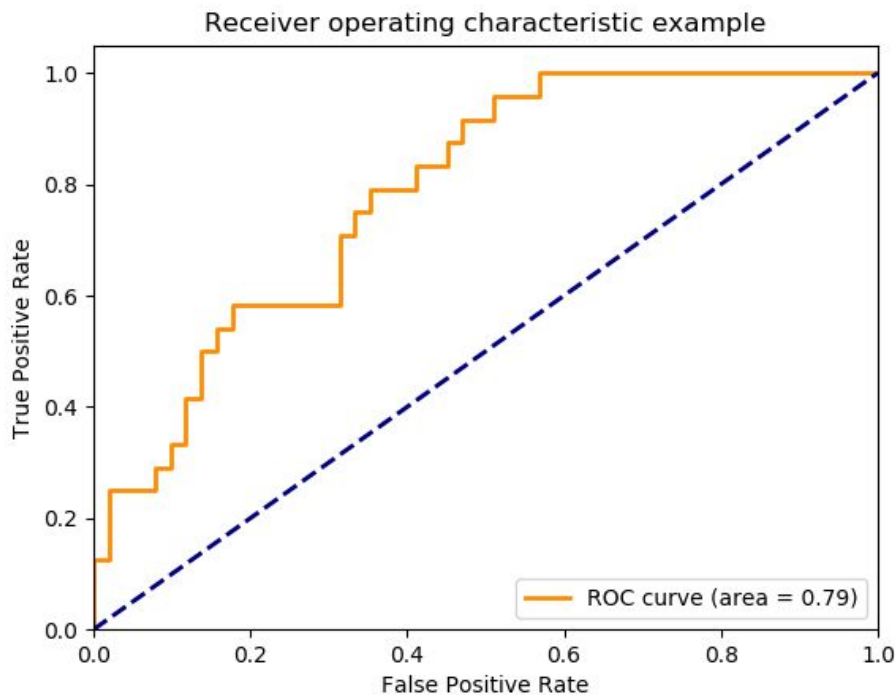
¡No siempre tendremos que graficar la curva!



## ¿Qué valor de umbral elijo?

Esto va a depender de nuestro problema, si queremos favorecer precisión o exhaustividad.

Pero al área bajo la curva es un indicador de cuán bueno es nuestro modelo independientemente de eso.





# Conclusiones

- Una **curva ROC** (Receiver Operating Characteristic) es una **representación gráfica que ilustra la relación entre TPR y el FPR** de un sistema clasificador para diferentes puntos de corte. NO confundir con una curva Precision-Recall.
- Se puede usar **para generar estadísticos** que resumen el **rendimiento o la efectividad de un clasificador**. El indicador más utilizado en muchos contextos es el área bajo la curva ROC o AUC (AUC- Área Bajo la Curva).
- Dato histórico: fue **desarrollada por ingenieros eléctricos en la II Guerra Mundial**, para **medir la eficacia de la detección de objetos enemigos en el campo de batalla mediante señales de radar**. Su uso está muy extendido en medicina para validar técnicas diagnósticas.

# Curva ROC en Scikit Learn

## `sklearn.metrics.roc_curve`

```
sklearn.metrics.roc_curve(y_true, y_score, pos_label=None, sample_weight=None, drop_intermediate=True) \[source\]
```

Compute Receiver operating characteristic (ROC)

Note: this implementation is restricted to the binary classification task.

## `sklearn.metrics.roc_auc_score`

```
sklearn.metrics.roc_auc_score(y_true, y_score, average='macro', sample_weight=None, max_fpr=None) \[source\]
```

Compute Area Under the Receiver Operating Characteristic Curve (ROC AUC) from prediction scores.

Note: this implementation is restricted to the binary classification task or multilabel classification task in label indicator format.

## `sklearn.metrics.auc`

```
sklearn.metrics.auc(x, y, reorder='deprecated') \[source\]
```

Compute Area Under the Curve (AUC) using the trapezoidal rule

This is a general function, given points on a curve. For computing the area under the ROC-curve, see `roc_auc_score`.  
For an alternative way to summarize a precision-recall curve, see `average_precision_score`.

A close-up photograph of a white ceramic cup filled with a latte. The surface of the milk is decorated with intricate latte art, featuring a central heart shape surrounded by concentric, wavy lines. The cup sits on a matching white saucer. In the background, a white napkin and a silver fork are visible, though they are out of focus. The overall lighting is soft and even, highlighting the textures of the coffee and the smooth surface of the cup.

**¡BREAK!**

---



# Hands-on training



## Hands-on training



DS\_Bitácora\_21\_CV\_ROC.ipynb



# Recursos



## Recursos

-  [Python Data Science Handbook](#) - Capítulo 5, *"Hyperparameters and Model Validation"*.
-  [La guía de Scikit-Learn sobre CV](#) y [La guía de Scikit-Learn sobre Curvas de ROC](#) - Son siempre una muy buena referencia.
- Mira [este](#) video sobre Curvas de ROC, si te costó entender el concepto visto en la bitácora, este video te va a ser de mucha ayuda.
- Juega con [este recurso](#), del cual hemos sacado algunas de las imágenes de esta bitácora.



# Para la próxima

---

- Termina el notebook de hoy.
- Lee la bitácora 22 y carga las dudas que tengas al Trello.

En el encuentro que viene uno/a de ustedes será seleccionado/a para mostrar cómo resolvió el challenge de la bitácora. De esta manera, ¡aprendemos todos/as de (y con) todas/as, así que vengan preparados/as.



ACÀMICA