DEPARTAMENTO
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

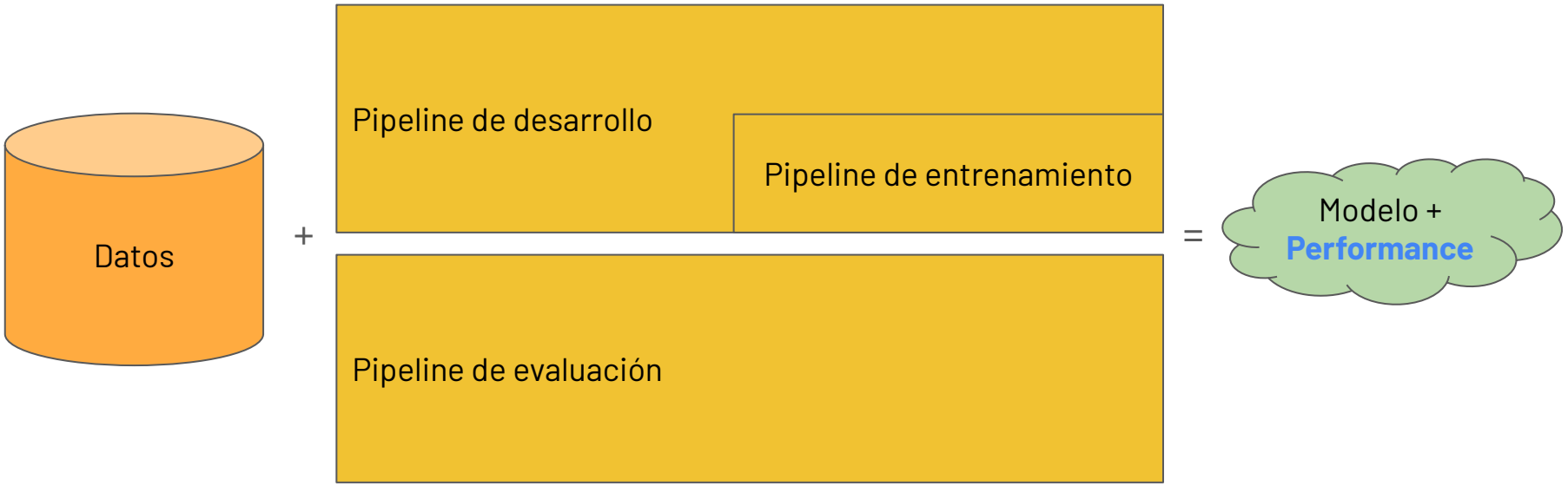
Aprendizaje Automático

1C-2024

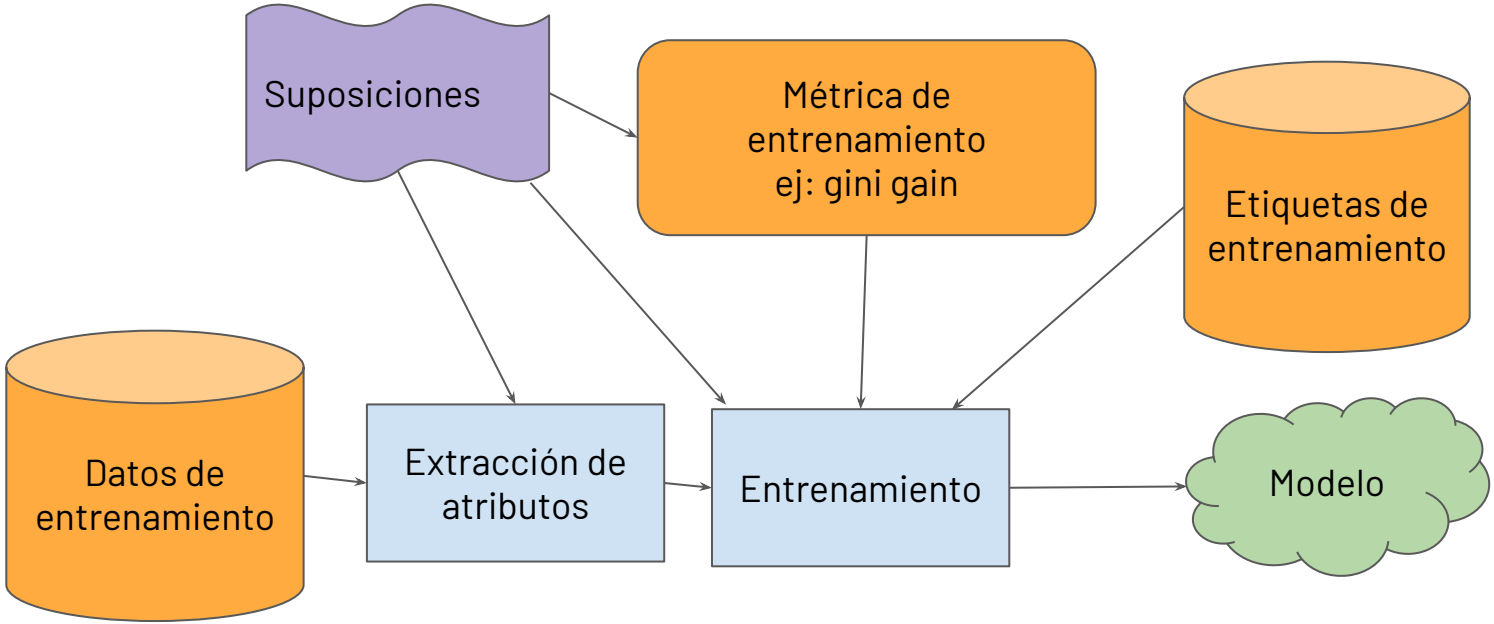
Clase 3:

Evaluación y Selección de Modelos
Métricas

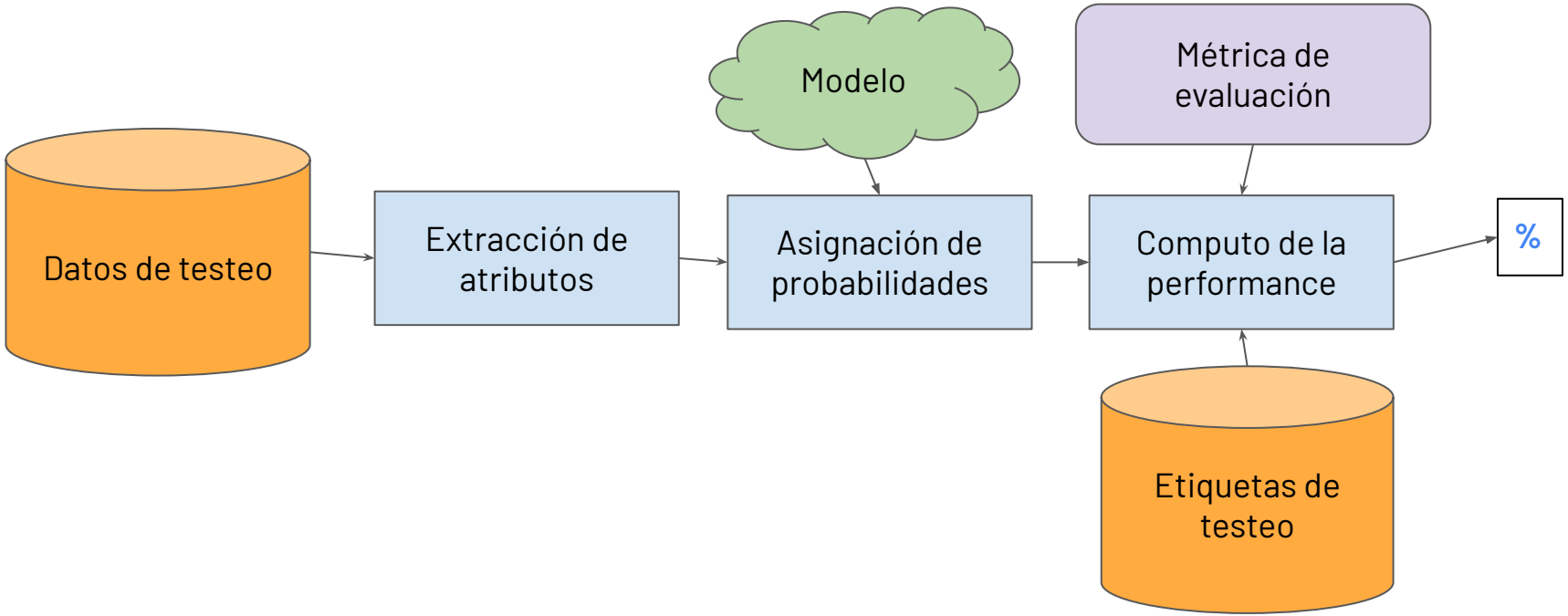
Visión global



Pipeline de entrenamiento



Pipeline de evaluación



¿Por qué enfocarse en la metodología de evaluación?

Casi todo lo que hacemos en Machine Learning depende en gran medida de la evaluación:

- ➢ **Comparamos** configuraciones de algoritmos mediante mediciones con alguna métrica.
- ➢ **Estimamos** la performance "en la realidad" (in the wild) y la reportamos a personas interesadas.

- ★ Es **fácil errar** en la creación de un modelo (entrenamiento), pero también es **fácil darse cuenta.**
- ★ Es **fácil errar** en la evaluación y **no es fácil darse cuenta**.
- ➢ No es raro ver un paper publicado o ver un sistema en producción con problemas en la evaluación. Pero lo que se publica / productiviza no funciona.

Evaluar bien significa entender bien el caso de uso,

- entender qué me importa del problema y qué no,
- entender qué métrica/s refleja/n lo que quiero capturar,
- entender qué mecanismo de evaluación usar,
- entender cómo no hacer/se trampa
- entender cómo hago que no me hagan trampa.

Selección y evaluación de modelos

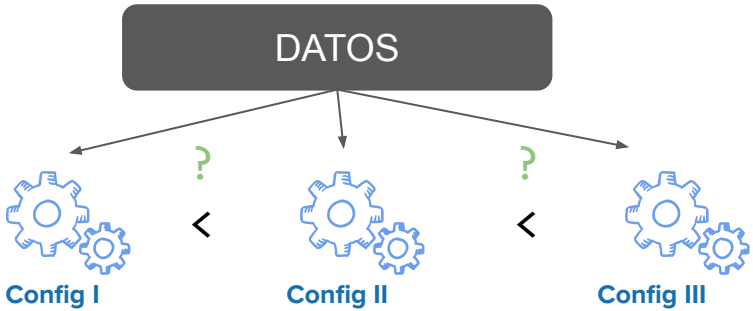
Dos preguntas distintas:

→ ¿Cómo **seleccionar** el mejor modelo entre varias opciones?

→ ¿Cómo **funcionará** mi modelo seleccionado en la "realidad"?

Warning! A veces "**modelo**" se utiliza para referirse a la **configuración** y no la **hipótesis** resultante.

Model Selection: Explorar distintas **configuraciones** de modelos (algoritmo + hiperparámetros + atributos) para poder seleccionar el mejor



Model Assessment (evaluación del modelo): Una vez **seleccionada la mejor configuración,** estimar la performance que tendrá de la mejor manera posible.

¿Performance real?

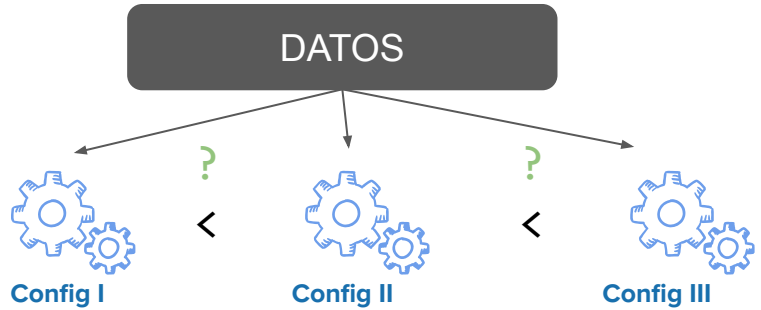
Parte I
Selección de modelos

Búsqueda del mejor modelo

Terminología

hiper-parámetros: Se refiere valores que se especifican manualmente en el algoritmo de aprendizaje antes de ejecutarlo. Por ejemplo en árboles: *Criterio de elección de atributos en cada nodo (Gini Gain, Information Gain), cantidad de hijos (árboles binarios vs. n-arios), criterio de parada (ej: max_depth), estrategia de poda, etc.*

parámetros: Se refiere a valores internos del modelo resultante que se aprenden (ajustan) a partir de ejecutar el algoritmo sobre un dataset. Representan las reglas aprendidas. Ej: *las triplas <nood_id, atributo, corte> obtenidos. "Lo que se guarda si hago un **modelo.zip**"*



¿Cómo buscar la mejor configuración?

Para poder elegir un modelo final, es natural explorar distintas combinaciones de **hiper-parámetros**, distintos **algoritmos** de aprendizaje y distintas **formas de extraer o procesar atributos.**

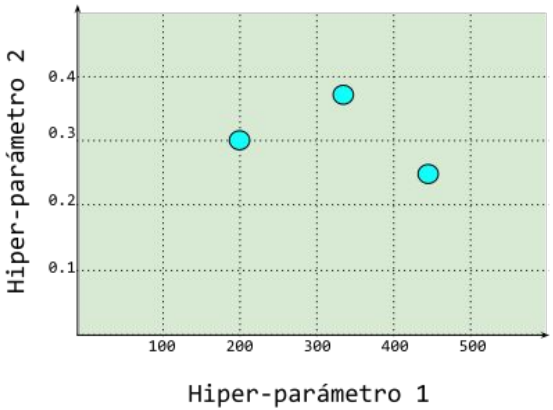
Búsqueda del mejor modelo

Métodos de exploración de combinaciones

Conf 83
Prof_max: 8
Criterio: X
Min gain: 0.008
...
→ Acc: 0.83

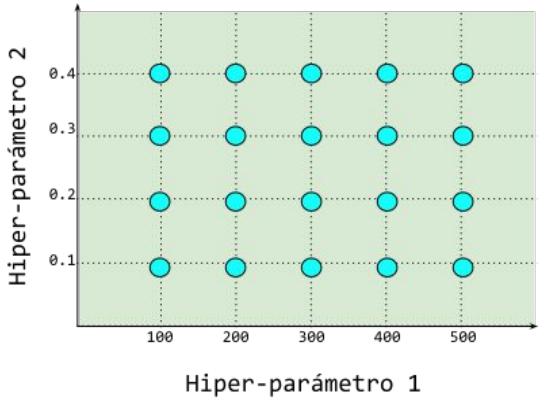
Manual Search

Setear a mano hiper-parámetros que pensamos que van a funcionar bien.



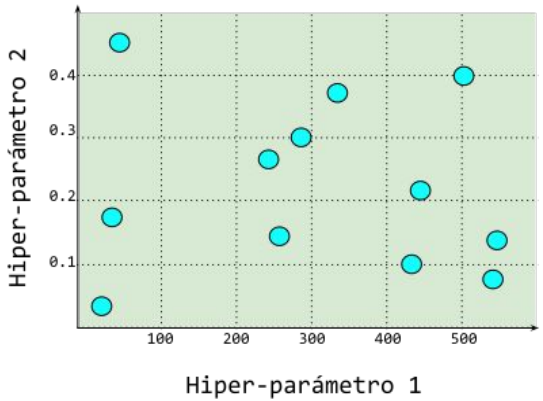
Grid Search

Plantear opciones y explorar todas las combinaciones.



Random Search

Se plantean distintas opciones y se exploran combinaciones al azar.



Bergstra, James, and Yoshua Bengio. "Random search for hyper-parameter optimization." *Journal of Machine Learning Research* 13.Feb (**2012**)

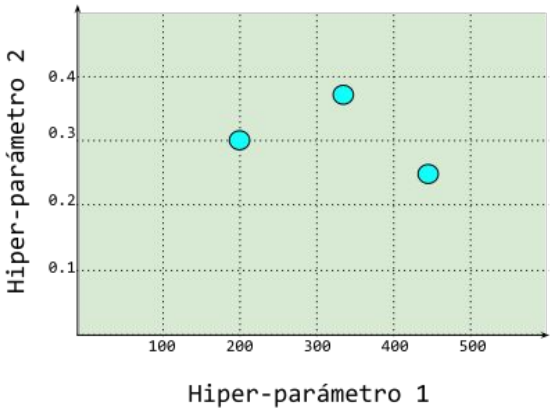
Búsqueda del mejor modelo

Métodos de exploración de combinaciones

Conf 83
Prof_max: 8
Criterio: X
Min gain: 0.008
...
→ Acc: 0.83

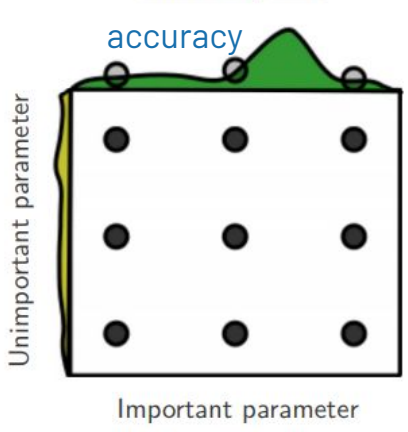
Manual Search

Setear a mano hiper-parámetros que pensamos que van a funcionar bien.



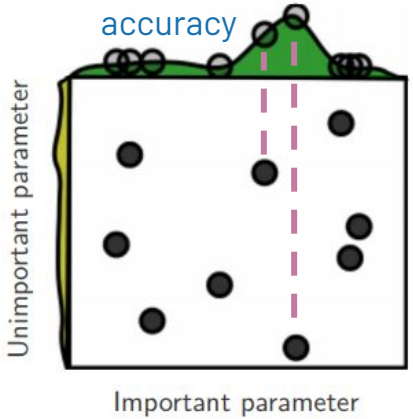
Grid Search

Plantear opciones y explorar todas las combinaciones.



Random Search

Se plantean distintas opciones y se exploran combinaciones al azar.



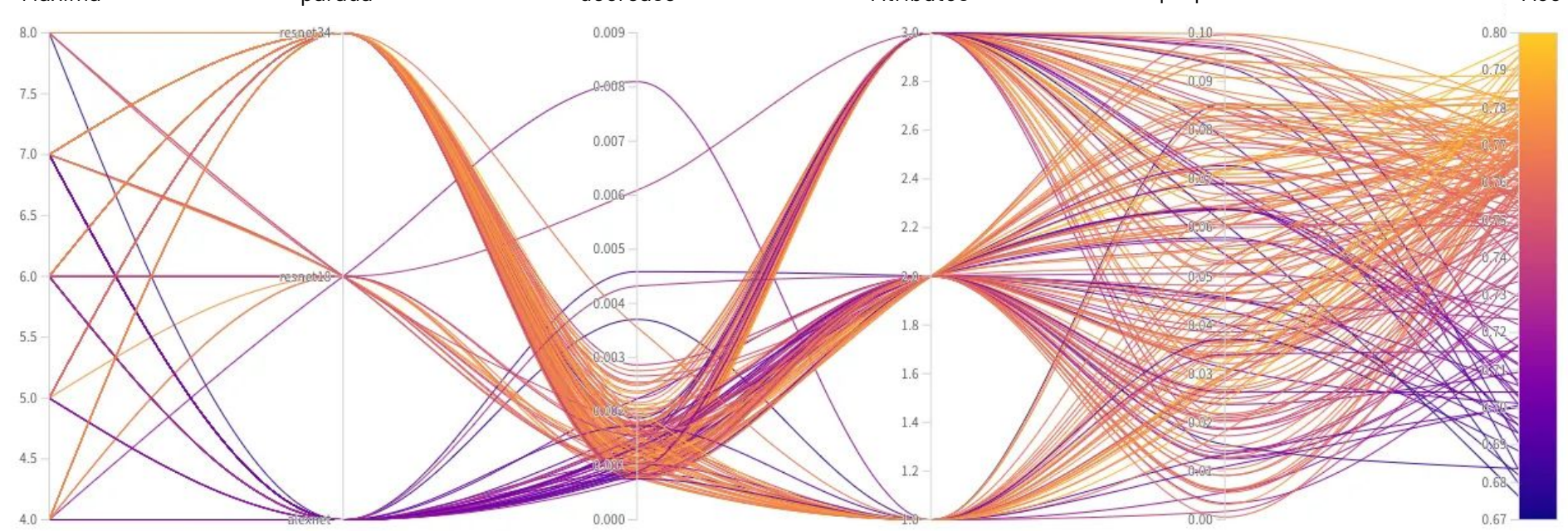
Bergstra, James, and Yoshua Bengio. "Random search for hyper-parameter optimization." *Journal of Machine Learning Research* 13.Feb (**2012**)

Búsqueda del mejor modelo

Ejemplo de visualización de resultados

Conf 83
Prof_max: 8
Criterio: X
Min gain: 0.008
...

→ Acc: 0.83



Parte II
Evaluación de una configuración dada

La expresión "**test**" es ambigua.
En esta materia evitaremos usarla.

Estimación de performance
Validación Cruzada

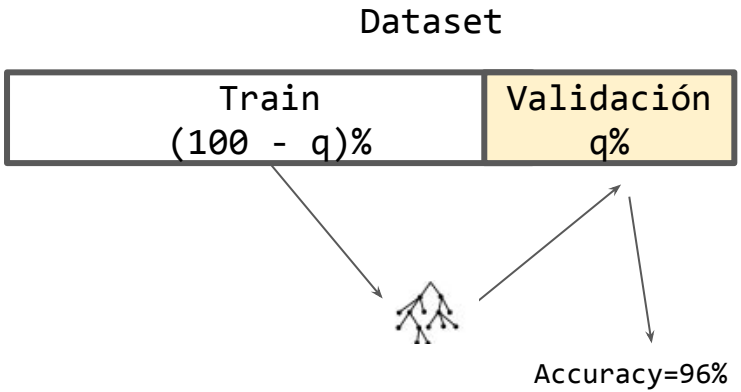
¿Cómo estimar la performance de una configuración?

Medir accuracy sobre datos de entrenamiento → mala idea. ¿Por qué?

Surge la necesidad de separar un **q%** de datos, para validar los modelos: datos de validación (o test).

Los datos se deben separar al azar(*), para evitar cualquier orden o estructura subyacente en los datos.

Validación Cruzada (cross validation)



(*) Esto no siempre es así. Veremos en un par de slides

Estimación de performance

K-Fold Cross Validation

- ¿Qué puede pasar si tenemos mala suerte al separar los datos para entrenamiento/validación? Estamos midiendo sólo sobre el **q%** (en general 5%, 10% o 20%) de nuestros datos. ¿El resultado será confiable? La estimación de performance del modelo podría no ser realista.

- Estos problemas surgen **especialmente** cuando tenemos **pocos datos**, si no, la validación cruzada suele alcanzar.

- Con pocos datos, sería más útil poder probar nuestro algoritmo de aprendizaje con todos nuestros datos. Surge la idea de validación Cruzada de "k" iteraciones: **K-Fold Cross Validation**

Estimación de performance

K-Fold Cross Validation

Dado **L** (un algoritmo de aprendizaje con ciertos hiperparámetros ya establecidos) y el dataset **D:**

1. Separamos **D** en **K subconjuntos** a los que llamamos **folds**: $D_1, D_2, D_3, \ldots, D_K$.

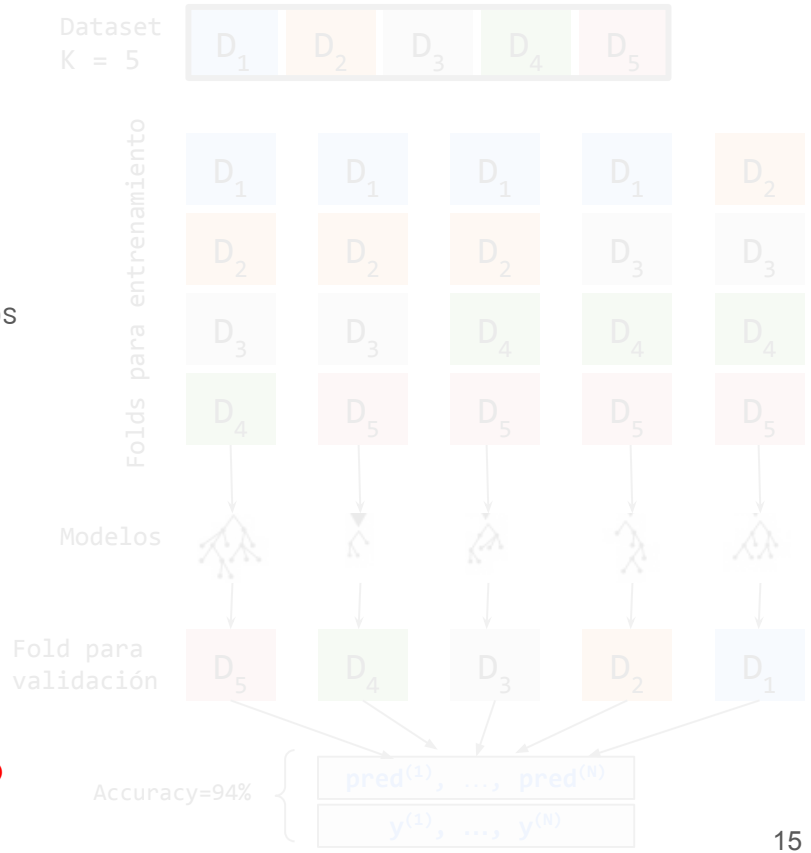
2. Construimos **K modelos** que serán entrenados utilizando todos los datos (salvo los del k-ésimo fold). Es decir, $f_{L,D-Dk} = L(D-D_k)$.

3. Para cada $x^{(i)} \in D$:

   - $pred^{(i)} = f_{L,D-Dk}(x^{(i)})$
     `# predecimos utilizando el modelo que no vio ese dato en entrenamiento.`

   - `predicciones[i] = pred`$^{(i)}$
     `# juntamos las predicciones como si vinieran todas del mismo modelo`

4. Computamos alguna métrica del error sobre el conjunto entero[(*)] **predicciones** vs **y**



(*) Algunos autores sugieren un abordaje distinto, en donde se calcula la métrica para cada fold de validación por separado y después se promedia el resultado (no lo

Estimación de performance

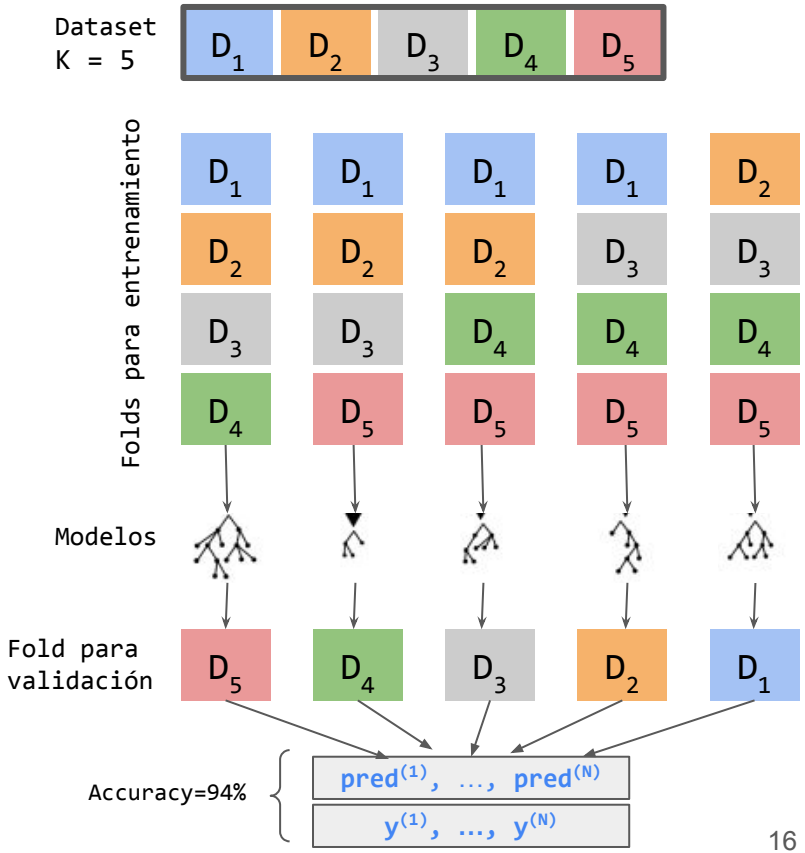
K-Fold Cross Validation

¿Qué sucede si **K = N**? (esto se llama leave one out cross validation)

- ¿Costo computacional?
- ¿Qué tan similares / distintos son los K modelos en ese caso?

Perfecto, tengo **K** modelos. **¿Cuál uso de ahora en más?**

- Hacer una votación ("Ensamble" de modelos)
- Usar el "mejor" (viendo los resultados por fold, pero con el riesgo de haber elegido un fold de validación "más fácil").
- ¿Re-entrenar utilizando todos los datos? OK, pero veremos **algunos riesgos** en breve.



Estimación de performance

Variaciones de K-Fold Cross Validation

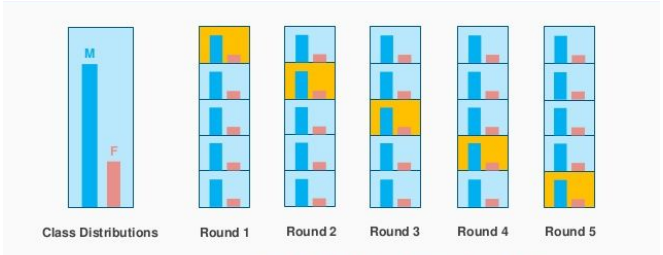
¿Es buena idea hacer **K folds al azar**?

- ¿Las clases están desbalanceadas?
- ¿Orden temporal de los datos?
- ¿Orden espacial? (datos regionales)
- ¿Qué sucede con instancias no independientes?
  - **Ej:** En reconocimiento del habla, un mismo hablante no debe aparecer en ambos conjuntos (¿o sí?).
  - **Ej2:** imágenes médicas de pacientes. ¿vienen de máquinas distintas? Si se usa máquina X hay más chances de enfermedad?
  - **Ej3:** : detección de emociones

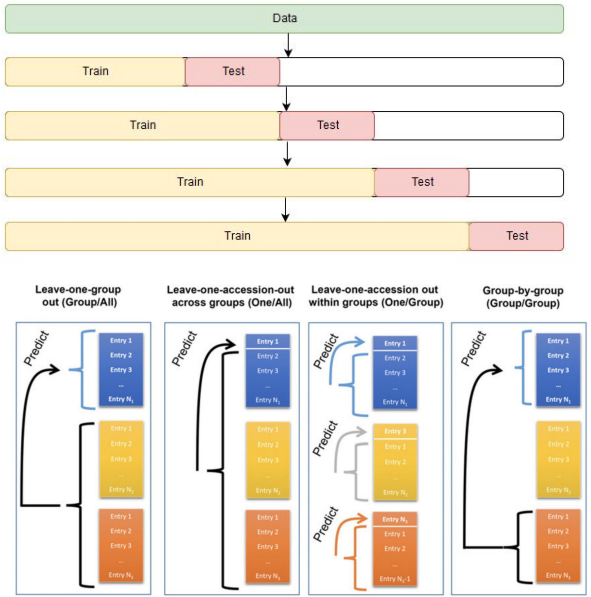
Pensamiento clave **¿Qué queremos aprender?**

Preguntarse entonces: ¿Tengo la misma cantidad de información que tendré en test (o en la realidad)? ¿Quiero mejorar mi predicción a costa de incluir esa información? (ej: máquina de imagenes médica)

Stratified K-fold cross validation.



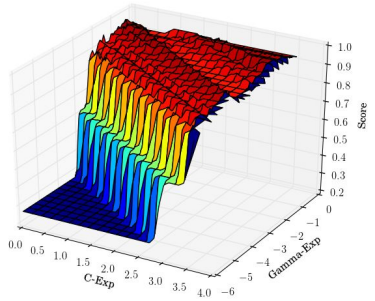
Temporal series K-fold cross validation.



Estas técnicas y más en Müller, A. C., & Guido, S. (2016). Introduction to machine learning with Python: a guide for data scientists.

Volviendo a la selección del mejor modelo

Una vez probadas distintas combinaciones de estructura del modelo e hiper-parámetros obtenemos un **puntaje por cada combinación**



(Utilizando cross validation)

Una vez seleccionados los mejores hiper-parámetros, **podemos entrenar un modelo utilizando todos nuestros datos.**

Escenario frecuente:

Construimos nuestro modelo con la combinación de atributos + algoritmos + hiperparámetros con mejor desempeño.

Lo ponemos a funcionar con datos nuevos, y los resultados son peores.

¿Qué falló?

A otro nivel, repetimos el mismo error de antes. Evaluamos un modelo sobre los mismos datos que usamos para construirlo.

- Entonces, sobreestimamos la performance de nuestro modelo.
- ¿Solución?

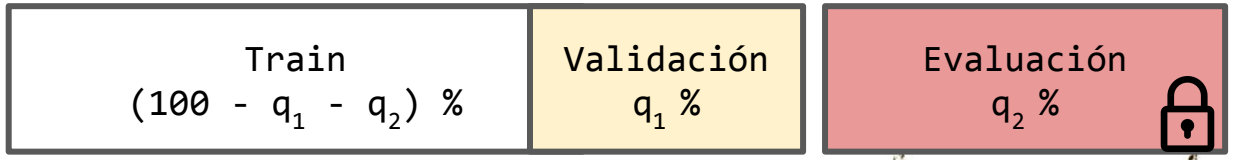
Parte III
Evaluación del modelo final

Desarrollo - Evaluación

- Lo antes posible separamos un **conjunto para evaluación.**
- Serán datos que no se toquen **hasta el final**.
- Todas las pruebas y ajustes se hacen sobre el conjunto para **Desarrollo**.
- Cuando **termina el desarrollo**, se **evalúa** sobre los datos de evaluación separados.
- La estimación de performance será la más realista.
- Ese valor es el que se reporta. **¡No volver atrás!**

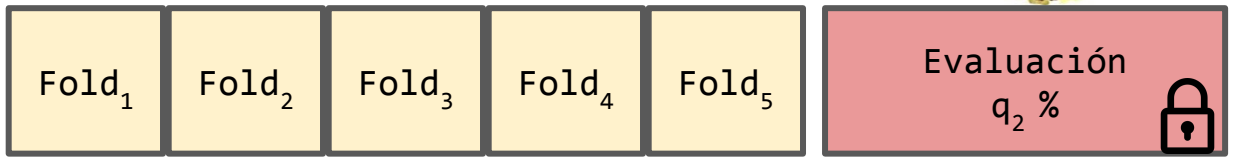
También llamado "**test set**", "**eval set**", "**hold out set**", "**held-out set**", "**control set**" depende la bibliografía.

Caso general



Desarrollo

Caso pocos datos
(K-fold cross val)



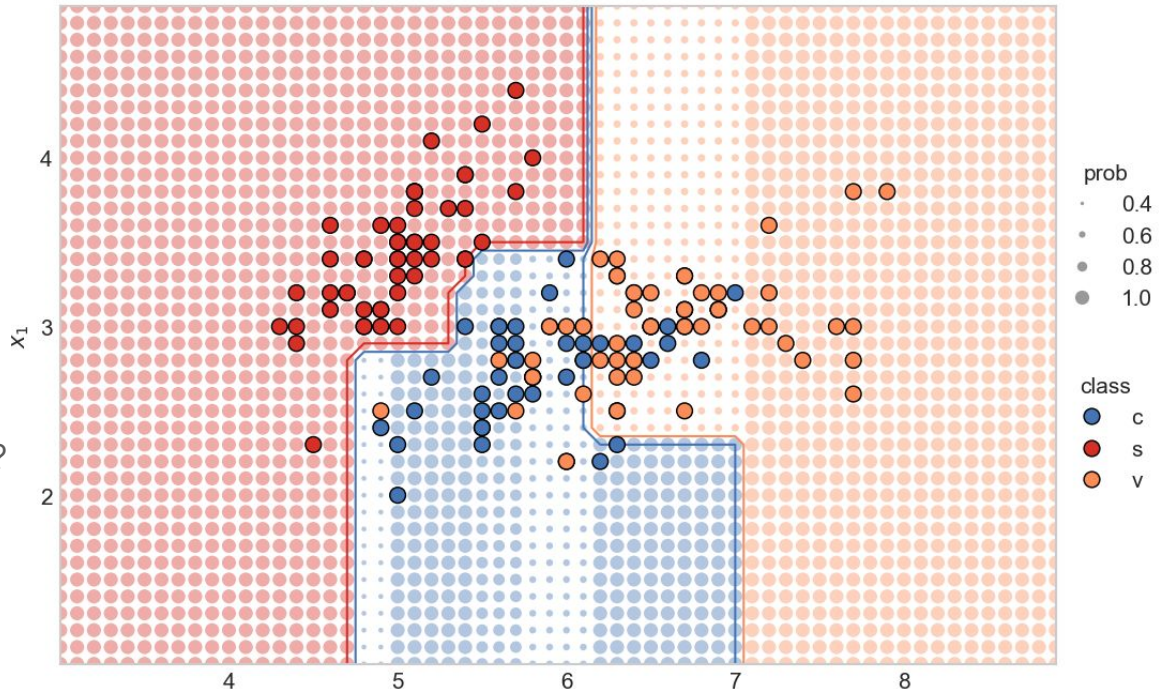
Desarrollo

Intervalo

Parte IV
Medidas de Performance

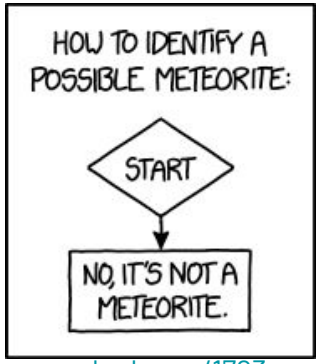
¿Cómo medimos la performance de un modelo particular?

1. ¿Qué tan buenas son las **asignaciones** a clases?
2. ¿Qué tan buenas son las **probabilidades** que se asignan?
3. ¿Funciona de igual manera **para cada clase**?
4. ¿Cómo hago un reporte con **un solo valor** que resuma la performance de mi clasificador?



Ejemplo de fronteras de decisión de un árbol de altura máxima 4.
Código para generar estos gráficos:
https://www.tvhahn.com/posts/beautiful-plots-decision-boundary/

Medidas de Performance



xkcd.com/1723

Un modelo tiene una *accuracy* (eficacia) del 99%.

- O sea, de cada 100 instancias, clasifica bien 99.

¿Qué significa esto?

- Según la tarea y la distribución de clases en el dominio, 99% puede ser muy bueno o pésimo.
- No dice nada sobre el tipo de aciertos y errores que comete el modelo.
- Ejemplos:
  - Un **filtro de spam** que marca todos los emails como spam.
  - Un **detección de fraude** que siempre dice “no fraude”
  - Un **Identificación** de meteoritos que siempre dice “NO”
- Nos interesa medir mejor, poder también asignar distintos costos a cada tipo de error y también poder tomar en cuenta los “scores” que devuelve un sistema.

Medidas de Performance

Caso binario: Matriz de confusión

Caso binario: una clase "claramente" **POSITIVA** y una clase "claramente" **NEGATIVA**.

Ejemplo: Detección COVID.

```
COVID? :: Atributos -> {Si, No}
```

Ejemplo resultado (sobre un val set):

Instancia	1	2	3	4	5	6
Clase real	Sí	Sí	No	Sí	Sí	No
Predicha	Sí	No	Sí	Sí	No	No
Resultado	OK	Err	Err	OK	Err	OK
Resultado Detallado	TP	FN	FP	TP	FN	TN

Matriz de Confusión		Clase Predicha	
	Total (P+N) 3841	COVID (PP) 2743	NO COVID (PN) 1098
Clase Real	COVID (P) 2795	2739 TP	56 FN
	NO COVID (N) 1046	4 FP	1042 TN

- **TP**: True Positives (verdaderos positivos)
- **TN**: True Negatives (verdaderos negativos)
- **FP**: False Positives (falsos positivos)
- **FN**: False Negatives (falsos negativos)

Para los siguientes sistemas,
¿qué tipo de error **FP** vs **FN** parece más dañino?

1. **Filtro de spam:** que descartará directamente los emails sospechosos.
2. **Detección de fraude:** prepara un listado de casos sospechosos para ser revisados por humanos.

Medidas de Performance

Caso binario: Precision y Recall

Precision: De las instancias predichas como positivas ¿qué porcentaje lo eran?
También llamado Positive Predictive Value (PPV)

Recall: De las instancias positivas, ¿qué porcentaje fueron predichas como tal?
También llamado Sensitivity, Hit Rate o True Positive Rate **(TPR)**

La precisión y el recall pueden interpretarse como probabilidades condicionales (estimadas)

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \mathbb{P}(C = P | \hat{C} = P)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \mathbb{P}(\hat{C} = P | C = P)$$

Matriz de Confusión		**Clase Predicha**	
	Total (P+N) 3841	COVID (PP) 2743	NO COVID (PN) 1098
Clase Real	COVID (P) 2795	2739 **TP**	56 **FN**
	NO COVID (N) 1046	4 **FP**	1042 **TN**

- **TP**: True Positives (verdaderos positivos)
- **TN**: True Negatives (verdaderos negativos)
- **FP**: False Positives (falsos positivos)
- **FN**: False Negatives (falsos negativos)

Para los siguientes sistemas,
¿qué métrica priorizararían? ¿**Precision** o **Recall**?

1. **Filtro de spam:** que descartará directamente los emails sospechosos.
2. **Detección de fraude:** prepara un listado de casos sospechosos para ser revisados por humanos.

Medidas de Performance

Caso binario: F-score

Precision (de los que dije positivo, cuántos lo eran):
TP / (**TP** + **FP**)
Recall (de los positivos, cuántos acerté)
TP / (**TP** + **FN**)

Vemos que alto **Recall** y alto **Precision** son dos características deseables del sistema. Pero un sistema con **Recall** 100% y **Precision** 0% o viceversa, no es un buen sistema.
¿ $Promedio = (recall + precision)/2$? → Mala idea (googleen por qué).

En general se utiliza: F_β (el "F-score")

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}}$$

F_1 es lo que se conoce como **media armónica.**

Efecto: los valores más bajos se penalizan proporcionalmente al inverso de su magnitud: cuanto más bajo sea el valor, mayor será la penalización.

Para los siguientes sistemas,
¿qué métrica usarían entre $F_{0.5}$ y F_2 ?

1. **Filtro de spam:** que descartará directamente los emails sospechosos.
2. **Detección de fraude:** prepara un listado de casos sospechosos para ser revisados por humanos.

Medidas de Performance

Caso binario: F-score

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}}$$

Criticas (entre otras) (fuente https://en.wikipedia.org/wiki/F-score)

- David Hand and others criticize the widespread use of the **F_1** score since it gives equal importance to precision and recall**.** In practice, different types of mis-classifications incur different costs. In other words, **the relative importance of precision and recall is an aspect of the problem.**[21]
- According to Davide Chicco and Giuseppe Jurman, the **F_1 score is less truthful and informative than** the Matthews correlation coefficient (MCC) in binary evaluation classification.[22]
- David Powers has pointed out that **F_1 ignores the True Negatives and thus is misleading for unbalanced classes**, while kappa and correlation measures are symmetric and assess both directions of predictability - the classifier predicting the true class and the true class predicting the classifier prediction, proposing separate multiclass measures Informedness and Markedness for the two directions, noting that their geometric mean is correlation.[23]
- Another source of critique of **F_1, is its lack of symmetry. It means it may change its value when dataset labeling is changed - the "positive" samples are named "negative" and vice versa.** This criticism is met by the P4 metric definition, which is sometimes indicated as a symmetrical extension of F_1.[24]

Umbral de decisión

Volviendo al ejemplo de COVID -> {Sí, No}

Instancia	1	2	3	4	5	6
Clase real	Sí	Sí	No	Sí	Sí	No
Predicha	Sí	No	Sí	Sí	No	No
Resultado	**OK**	Err	Err	**OK**	Err	**OK**
Resultado Detallado	**TP**	**FN**	**FP**	**TP**	**FN**	**TN**

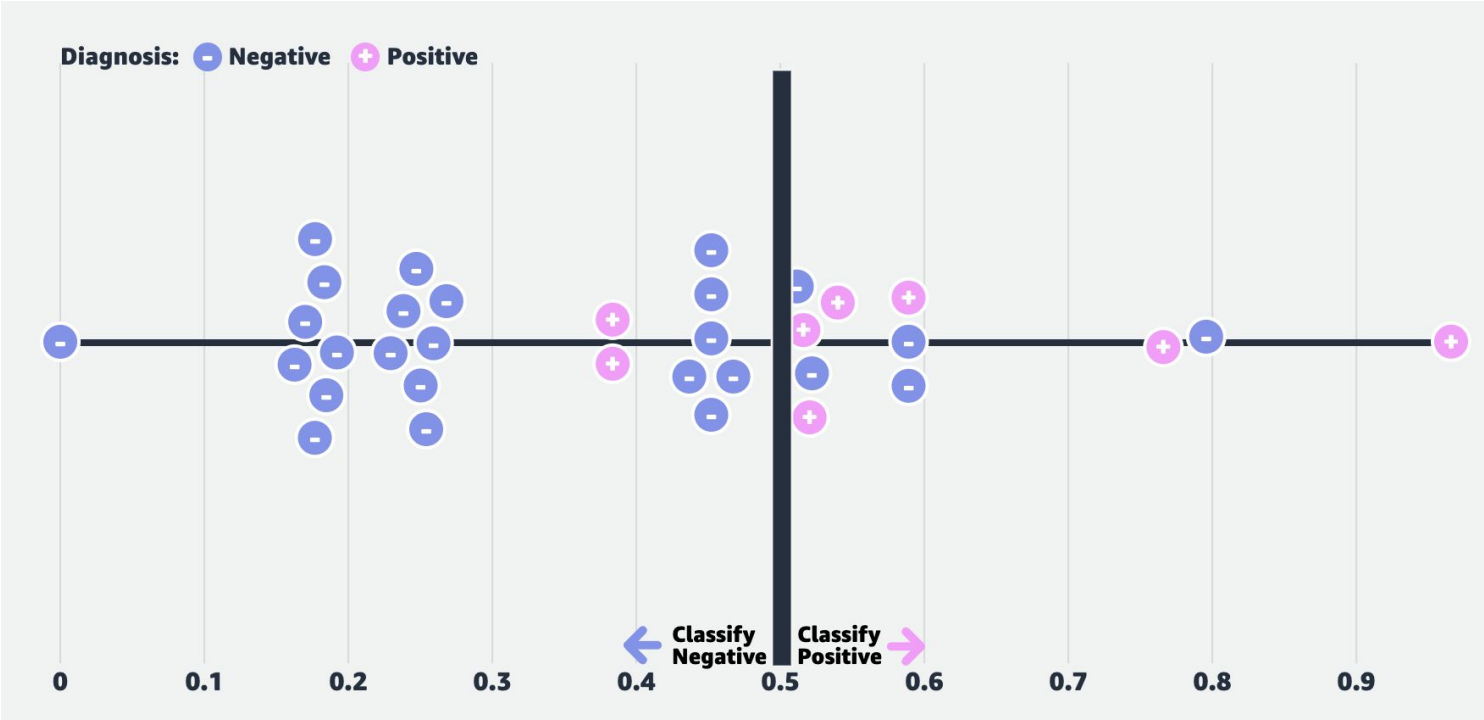
Podemos pensar que esta tabla (y consecuentemente la **matriz de confusión**) se generó de la siguiente manera:

a) Utilizamos el modelo M que prediga cada instancia.
b) M devolvió scores (supongamos [0,1])
c) Decidimos etiquetar según el **umbral 0.5.**
$$\textit{Predicción}(x) = \textit{Sí} \text{ si } P(x) > 0.5$$

¿Qué ocurre si cambiamos el umbral para ser?

$$\textit{Predicción}(x) = \textit{Sí} \text{ si } P(x) > 0.3$$

Umbral de decisión



Fuente: https://mlu-explain.github.io/precision-recall/

Umbral de decisión

Volviendo al ejemplo de COVID -> {Sí, No}

Instancia	1	2	3	4	5	6
Clase real	Sí	Sí	No	Sí	Sí	No
Predicha	Sí	No	Sí	Sí	No	No
Resultado	OK	Err	Err	OK	Err	OK
Resultado Detallado	TP	FN	FP	TP	FN	TN

Podemos pensar que esta tabla (y consecuentemente la **matriz de confusión**) se generó de la siguiente manera:

a) Utilizamos el modelo M que prediga cada instancia.
b) M devolvió scores (supongamos [0,1])
c) Decidimos etiquetar según el **umbral 0.5.**
$$Predicción(x) = Sí \text{ si } P(x) > 0.5$$

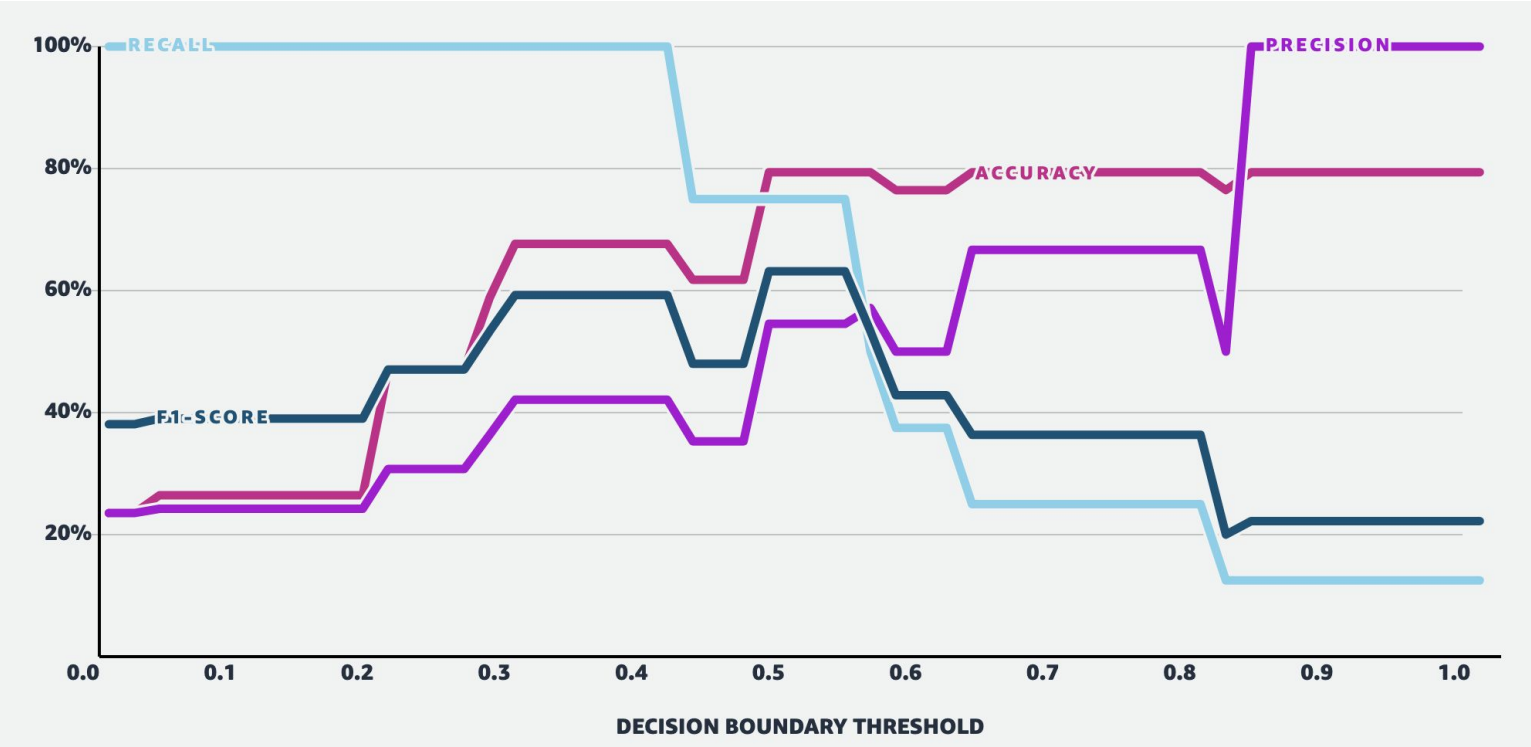
¿Qué ocurre si cambiamos el umbral para ser?

$$Predicción(x) = Sí \text{ si } P(x) > 0.3$$

Instancia	1	2	3	4	5	6
Clase real	Sí	Sí	No	Sí	Sí	No
Predicha	Sí	**Sí**	Sí	Sí	No	**Sí**
Resultado	OK	OK	Err	OK	Err	Err
Resultado Detallado	TP	TP	FP	TP	FN	FP

¡Tenemos que actualizar la matriz de confusión!

Umbral de decisión



Fuente: https://mlu-explain.github.io/precision-recall/

Medidas de Performance

Caso binario: curvas Recall-Precision

Al comparar algoritmos, generalmente no interesa la asignación final de etiquetas. Pero sí, ver qué tan bien un modelo **ORDENA** a las instancias. Luego, elegido el modelo, seteamos el umbral dependiendo de los requerimientos del problema**.**

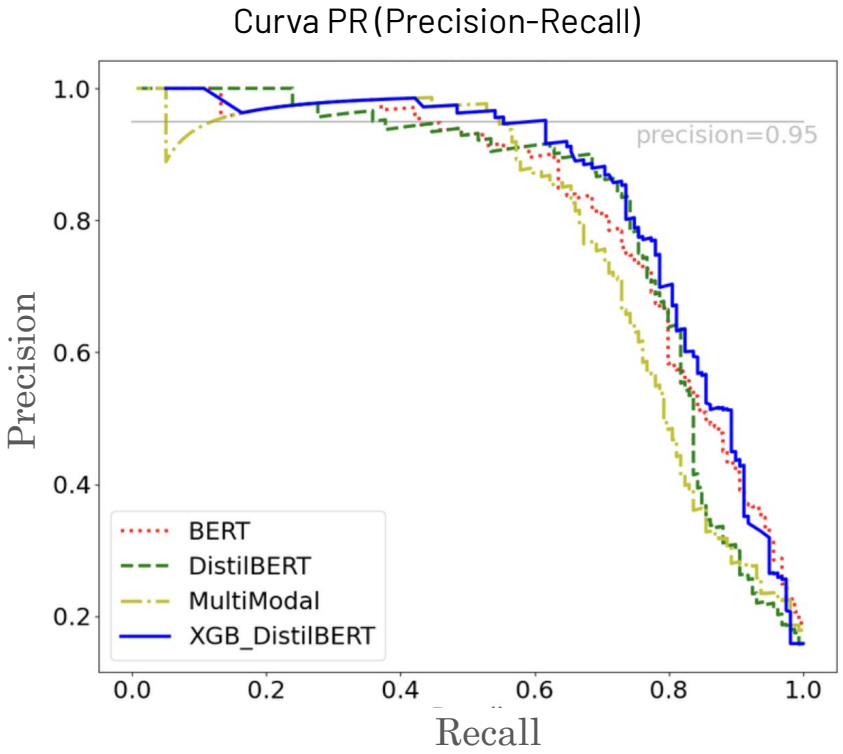
IDEA del algoritmo. Cómo funciona mi modelo si digo:

- $Predicción(x) = Sí$ si $P(x) > 0.01$
- $Predicción(x) = Sí$ si $P(x) > 0.02$
- ...
- $Predicción(x) = Sí$ si $P(x) > 0.98$
- $Predicción(x) = Sí$ si $P(x) > 0.99$

Métricas que se deducen de estas curvas:

- **AUC-PR:** área bajo la curva Precision-Recall.
- **Recall@X:** cuál es el recall si fijamos la precisión en X.

Tarea, ver qué es la curva ROC (mlu-explain.github.io/roc-auc/) y qué pasa cuando hay imbalances grandes (youtube.com/watch?v=6tJXX3vmYI0)



Medidas de Performance

Caso multiclase

Matriz de Confusión		**Clase Predicha**		
		Perro	Gato	Loro
Clase Real	Perro	$TP_{(C=perro)}$	X_2	
	Gato	X_1	$TP_{(C=gato)}$	
	Loro			$TP_{(C=pato)}$

X_1 = **TN** para Perro, **FN** para Gato, **TN** para Loro.
X_2 = **FP** para Perro, **FN** para Gato, **TN** para Loro.

Calculamos las métricas por clase:

$$\text{Precision}_{(C=c)} = \frac{\text{TP}_{(C=c)}}{\text{TP}_{(C=c)} + \text{FP}_{(C=c)}}$$

$$\text{Recall}_{(C=c)} = \frac{\text{TP}_{(C=c)}}{\text{TP}_{(C=c)} + \text{FN}_{(C=c)}}$$

$$\text{F-1 Score}_{(C=c)} = \frac{2 \times \text{Precision}_{(C=c)} \times \text{Recall}_{(C=c)}}{\text{Precision}_{(C=c)} + \text{Recall}_{(C=c)}}$$

¿Puede obtener un sólo número para la métrica *M*?

a) Macro average:
*⅓ * (M(p) + M(g) + M(l))*

b) "Weighted average":
freq(p).M(p)+ freq(g).M(g) + freq(l).M(l)

c) "Micro average" (juntamos antes de calcular nada)
TP = **TP**(p)+**TP**(g) + **TP**(l)
FP = **FP**(p)+**FP**(g) + **FP**(l)
FN = **FN**(p)+**FN**(g) + **FN**(l)
TN = **TN**(p) + **TN**(g) + **TN**(l)

Notas extras:

Hay una costumbre muy grande de dejarse llevar por las métricas que "usa la mayoría" (o que vienen usando en los papers de tal rama).

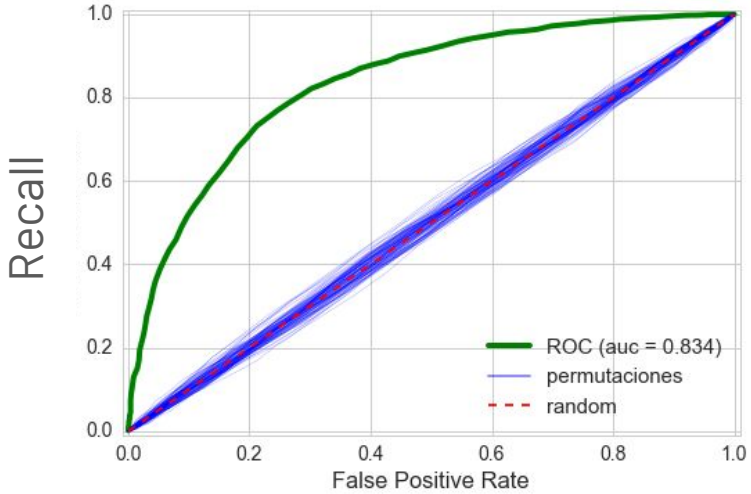
En ciertas áreas de machine learning **(ej: análisis de emociones)** las métricas que se utilizan no tienen ningún sentido (o al menos favorecen sistemas que no aprendieron tan bien como otros).

Es importante **ser críticos** en este tema y tomarse una gran parte del tiempo del desarrollo del sistema para elegir una métrica correcta.

¡**Hay métricas muy buenas** muy poco utilizadas!

Para más sobre este tema: charla / curso de Luciana Ferrer.

Totalmente opcional, pero muy recomendable:
Raschka, S. (2018). **Model evaluation, model selection, and algorithm selection in machine learning.** arXiv preprint arXiv:1811.12808.
https://arxiv.org/abs/1811.12808



Curva AUC-ROC + test de permutaciones
(métricas + estadísticas es posible!)

Resumen

Evaluación, validación y selección de modelos:

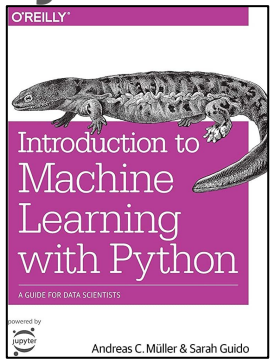
- ¡Cuidado con la sobreestimación de resultados!
- Validación cruzada (datos de entrenamiento vs. validación vs. control).
- k-fold cross validation.
- Selección de modelos: grid search y random search.
- Conjuntos de datos: desarrollo, held-out.

Métricas:

- Accuracy (eficacia), matriz de confusión, precision, recall.
- Umbral de decisión
- Curva Precision-Recall y área bajo la curva. AUC, Recall@X. (y les quedó entender ROC)
- Métricas multiclase y heurísticas para promediar.
- Faltó (entre otras cosas):
  - Costos en los errores.
  - Teoría de la decisión óptima.
  - Problemas de balanceo
  - Etc, etc, etc.

Tarea

- Leer el **capítulo 5 “Model Evaluation and Improvement”** de Müller, A. C., & Guido, S. (2016). **Introduction to machine learning with Python: a guide for data scientists.**



- Completar las notebooks
- Completar el quiz