ACÁMICA

¡Bienvenidos/as a Data Science!

Agenda

¿Cómo anduvieron?

Repaso

Explicación: Validación Cruzada

Hands-On

Break

Explicación: Eligiendo Hiperparámetros

Hands-On

Buenas práctica de un data scientist

Cierre

¿Cómo anduvieron?

Repaso

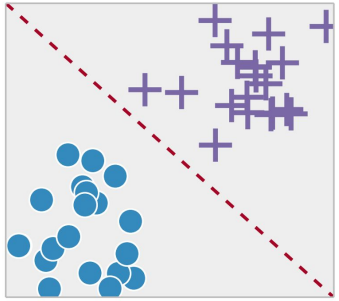
Machine Learning

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

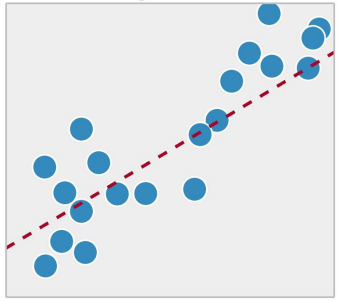
Clasificación



La variable de salida es una categoría:

- Enfermo / Sano
- Gato / Perro / Pájaro
- Spam / no Spam

Regresión



La variable de salida es un valor:

- Precio
- Cantidad

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 ***f*** desconocida, queremos aproximarla mediante un modelo.

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

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.

Aprendizaje supervisado: Clasificación

Modelos

- Árboles de decisión (Hiperparámetros: profundidad, criterio de entrenamiento, etc.)

- KNN (Hiperparámetros: cantidad de vecinos, distancia, etc.)

Métricas de evaluación

- Exactitud
- Precisión/Exhaustividad
- F-Score
- Matriz de Confusión[1]

[1] Bueno, técnicamente no es una métrica

Aprendizaje supervisado: Regresión

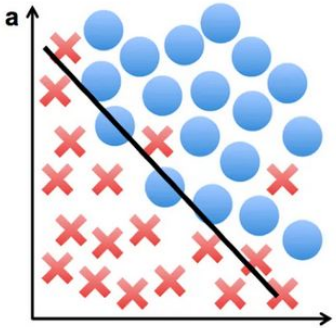
Modelos

- **Regresión lineal** (¿Hiperparámetros?)
- **Árboles de decisión** (Hiperparámetros: profundidad, etc.)
- **KNN** (Hiperparámetros: cantidad de vecinos, distancia, etc.)

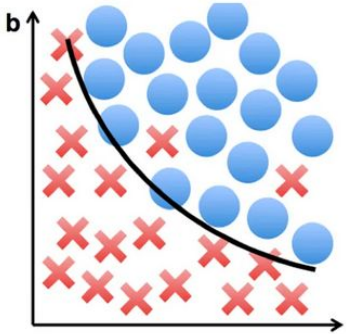
Métricas de evaluación

- MAE
- MSE/RMSE

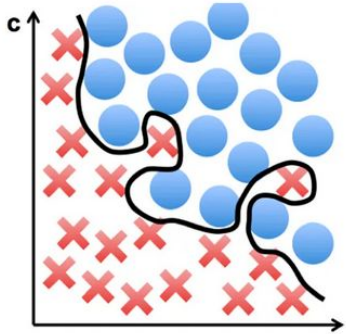
Pero no es tán fácil… Recordemos estas tres situaciones



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

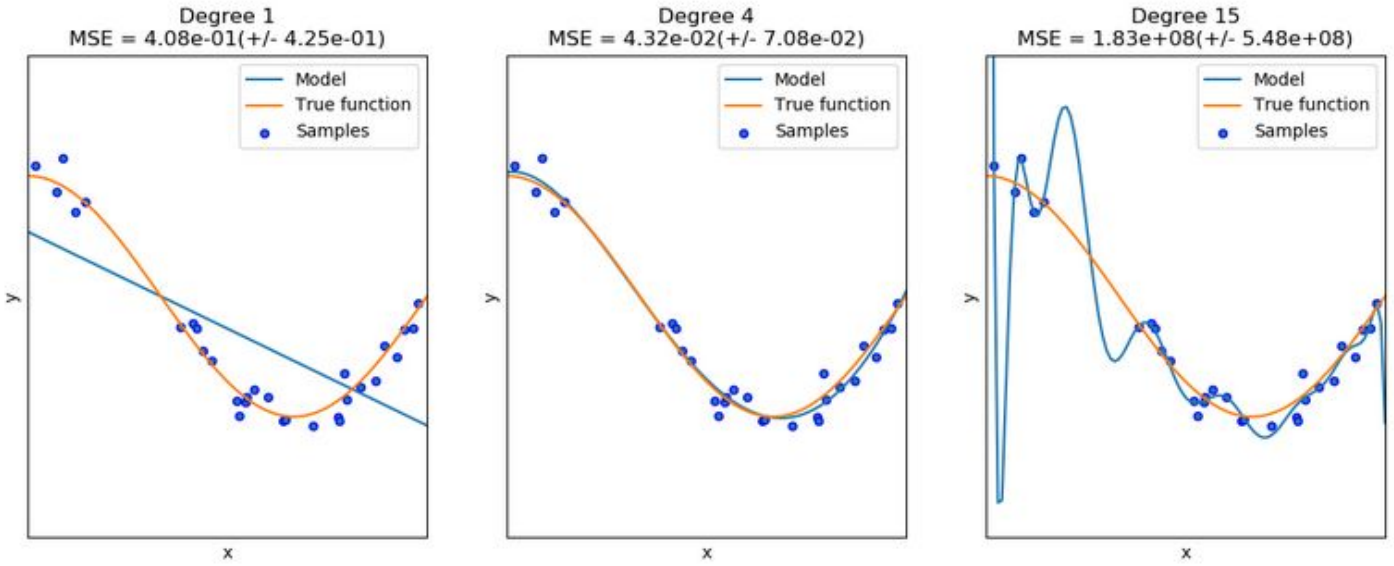


El **modelo b** tiene la complejidad suficiente para encontrar una frontera que parece apropiada para estos datos.



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



Underfitting

Good

Overfitting

Fuente:
https://scikit-learn.org/stable/auto_examples/model_selection/plot_underfitting_overfitting.html#fitt

¿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.**

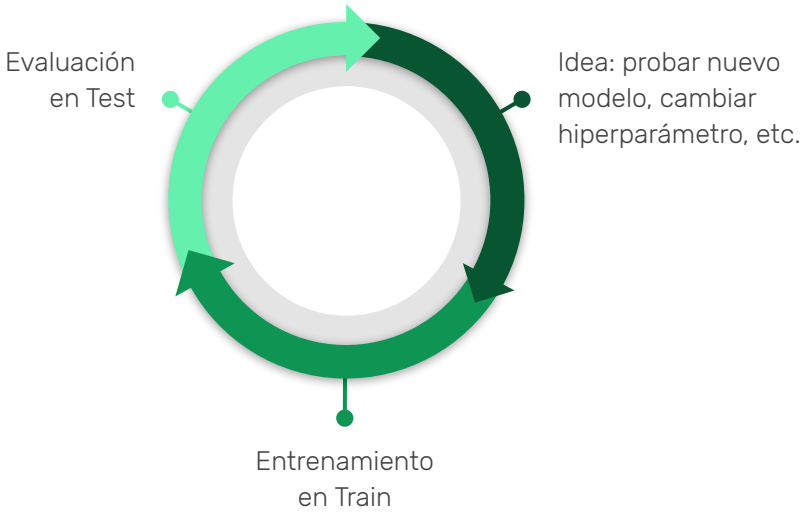
Pero...

Machine Learning involucra un proceso altamente iterativo. En general, entrenamos muchos modelos (ya sea de distinto tipo o variando hiperparámetros).

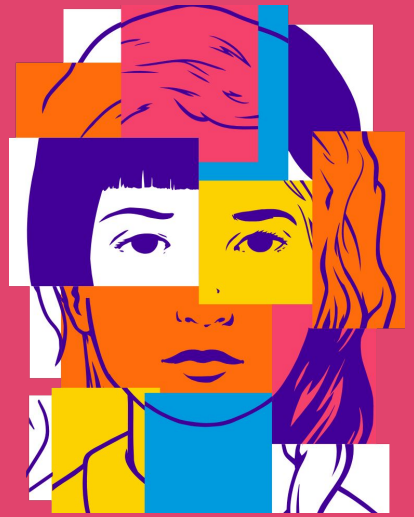
A medida que entrenamos nuevos modelos, puede ocurrir que un modelo tenga una buena performance en el conjunto de Test **por azar.**

Podemos creer que estamos seleccionando el mejor modelo disponible cuando en realidad estamos seleccionando un modelo mediocre.

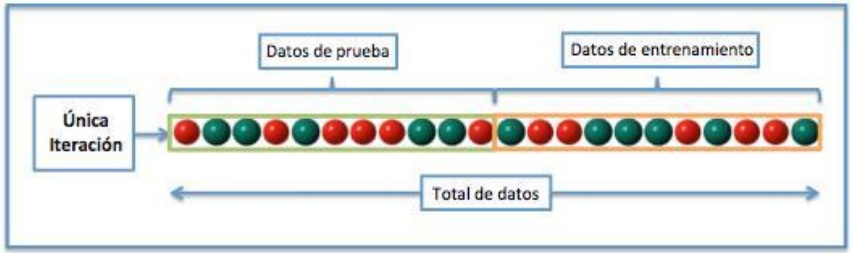
¿Se puede hacer mejor?



Validación Cruzada



Hasta ahora tenemos



Entrenamos modelos con datos de entrenamiento

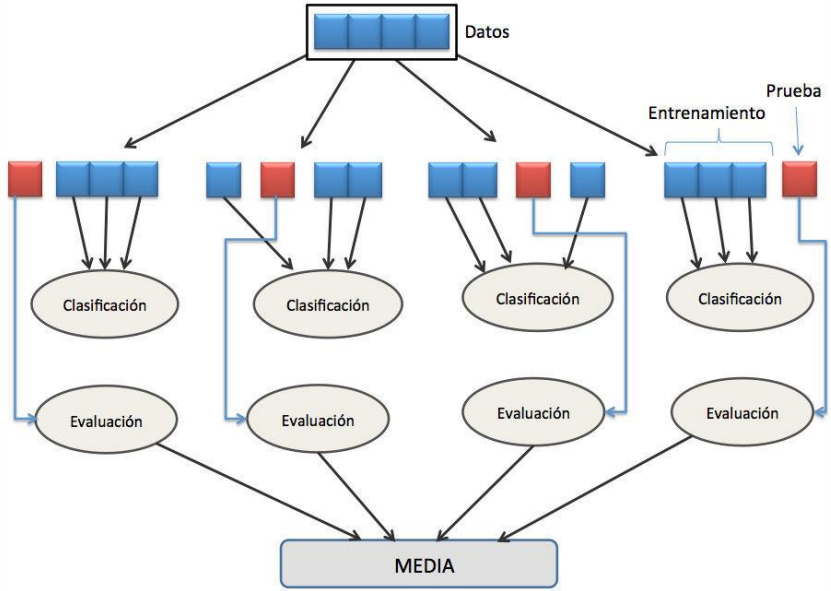
Performance sobre datos de prueba

Fuente: https://es.wikipedia.org/wiki/Validaci%C3%B3n_cruzada

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!

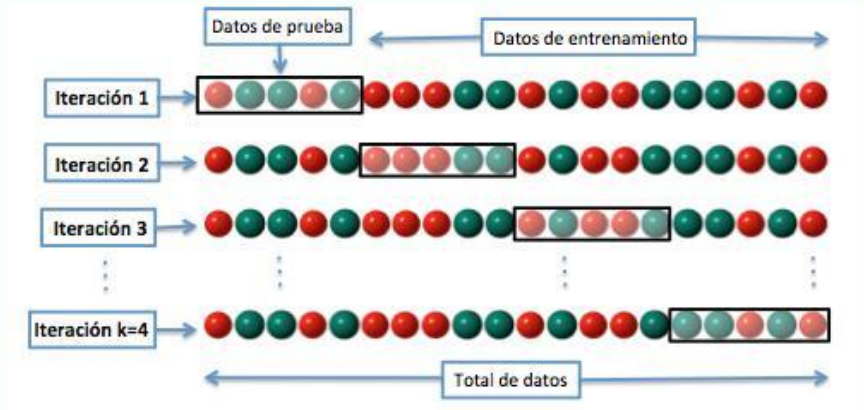


De esta forma, esperamos que la medida de performance sea independiente de la partición de los datos

Fuente: https://es.wikipedia.org/wiki/Validaci%C3%B3n_cruzada

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

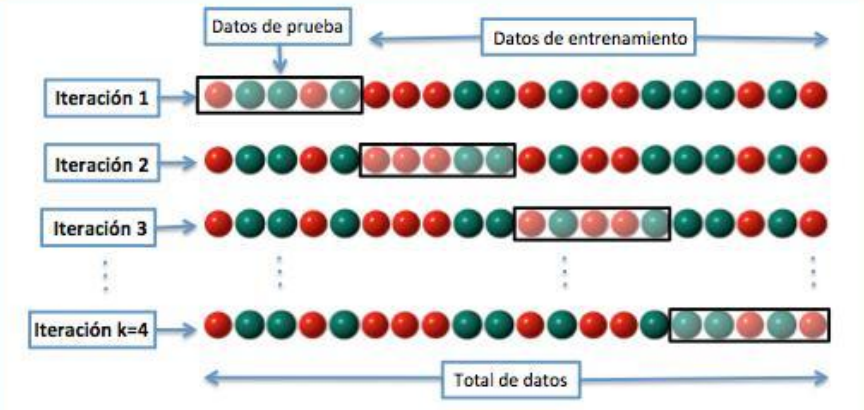
Modelo → Performance k = 4

Promedio

Fuente: https://es.wikipedia.org/wiki/Validaci%C3%B3n_cruzada

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.

Fuente: https://es.wikipedia.org/wiki/Validaci%C3%B3n_cruzada

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

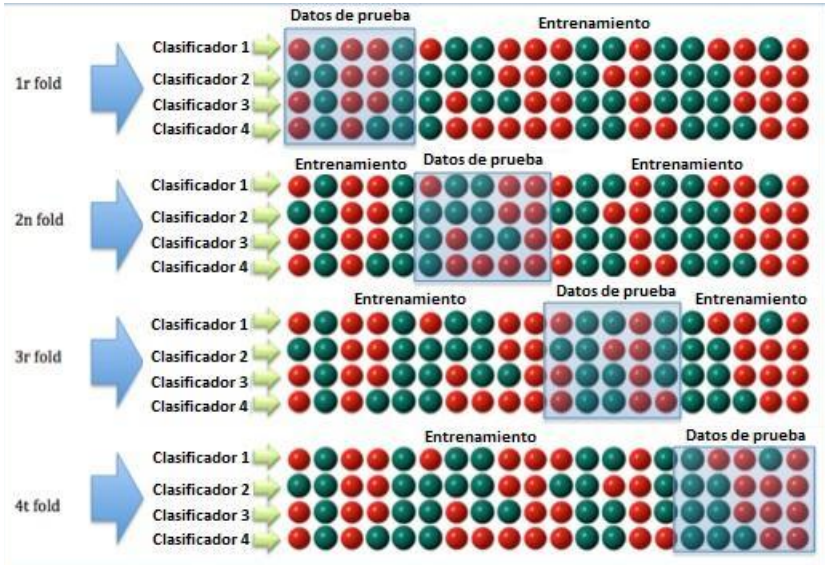
2) Separar en K folds (muestras) del mismo tamaño

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 = número de datos?

Si queremos comparar muchos modelos:



Performance 1 de cada modelo

Performance 2 de cada modelo

Performance 3 de cada modelo

Performance k=4 de cada modelo

Promedio y varianza para cada modelo

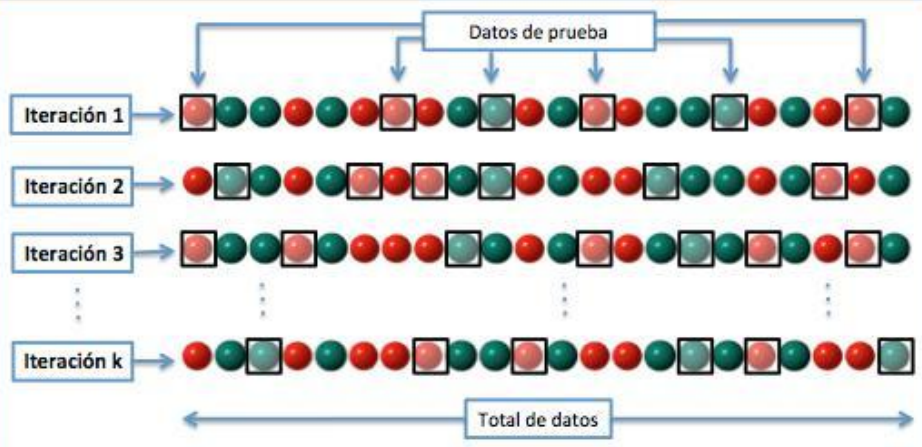
Fuente: https://es.wikipedia.org/wiki/Validaci%C3%B3n_cruzada

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 pocas clases.

Validación Cruzada Aleatoria

Existen otras técnicas de validación cruzada. Otra común es validación cruzada aleatoria.



En este caso, cada dato puede aparecer más de una vez en el conjunto de prueba.

Fuente: https://es.wikipedia.org/wiki/Validaci%C3%B3n_cruzada

¿Cómo quedan entonces los conjuntos de datos?

¿Ya no tenemos que hacer train/test split?

¿Cómo quedan entonces los conjuntos de datos?

¿Ya no tenemos que hacer train/test split?

¡Tenemos que seguir haciéndolo!

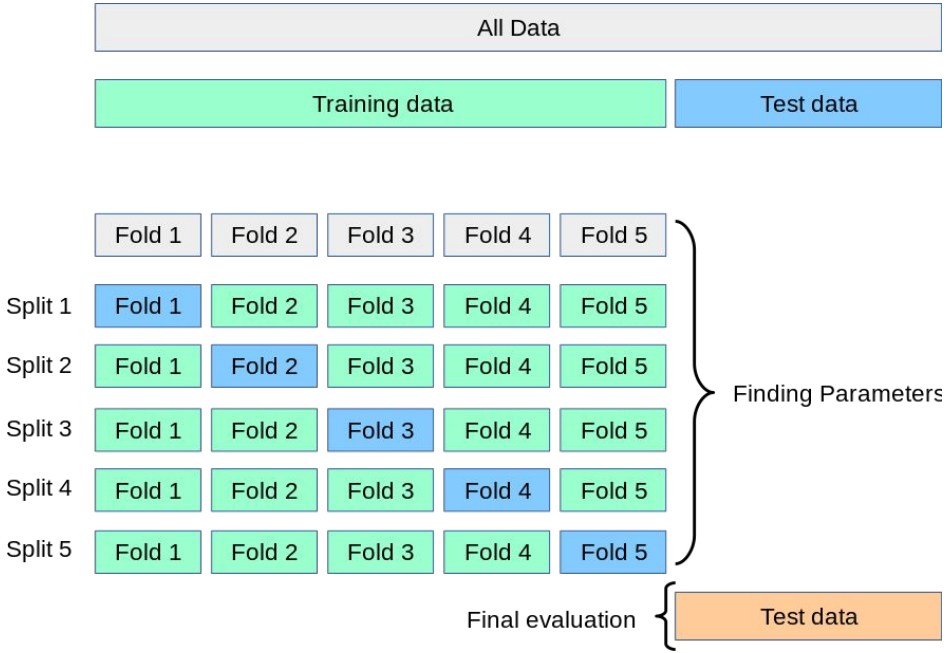
El esquema fundamental para separar nuestros datos es éste:

1. Hacemos un *train/test split* para separar dos conjuntos: uno de *train* y uno de *test*:

Train	**Test**
- Experimentación con atributos, algoritmos e hiperparámetros.	- Estimación realista de performance

1. Dentro del conjunto de train, hacemos **todas las pruebas** que consideremos necesarias y evaluamos los modelos resultantes usando **validación cruzada.** Elegimos nuestro modelo a partir del desempeño en estos datos (y las otras condiciones que consideremos para nuestro problema, como desempeño, Navaja de Ockham, etc.).
2. Evaluamos el **desempeño del modelo** elegido en el conjunto de test. Es el desempeño que vamos a reportar.

Por ejemplo, podría ser así...



Otro comentario acerca de la nomenclatura.

Lamentablemente, la comunidad no se pone de acuerdo en llamar de una única manera a cada conjunto. Algunas combinaciones típicas son train/dev/test, train/test/val, train/val/hold-out, etc.

Fuente: https://scikit-learn.org/stable/modules/cross_validation.html

CV en Scikit-Learn

¡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]

Hands-on training

Hands-on training

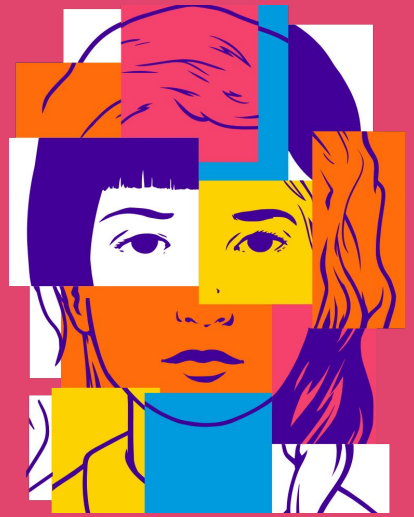
DS_Clase_20_CV.ipynb

Parte 1

¡BREAK!

Ph. Credit: Drew Coffmann

Eligiendo Hiperparámetros



Curvas de validación (o complejidad)

En general, el desempeño de un modelo depende de **muchos hiperparámetros.**
Pero a veces hay uno que es el más importante, el que predomina sobre el resto.

Curvas de validación (o complejidad)

En general, el desempeño de un modelo depende de **muchos hiperparámetros.** Pero a veces hay uno que es el más importante, el que predomina sobre el resto.

Para elegir el valor de ese hiperparámetro - y también caracterizar mejor el desempeño de nuestro modelo -, es útil obtener las **curvas de validación.**

Curvas de validación (o complejidad)

En general, el desempeño de un modelo depende de **muchos hiperparámetros.** Pero a veces hay uno que es el más importante, el que predomina sobre el resto.

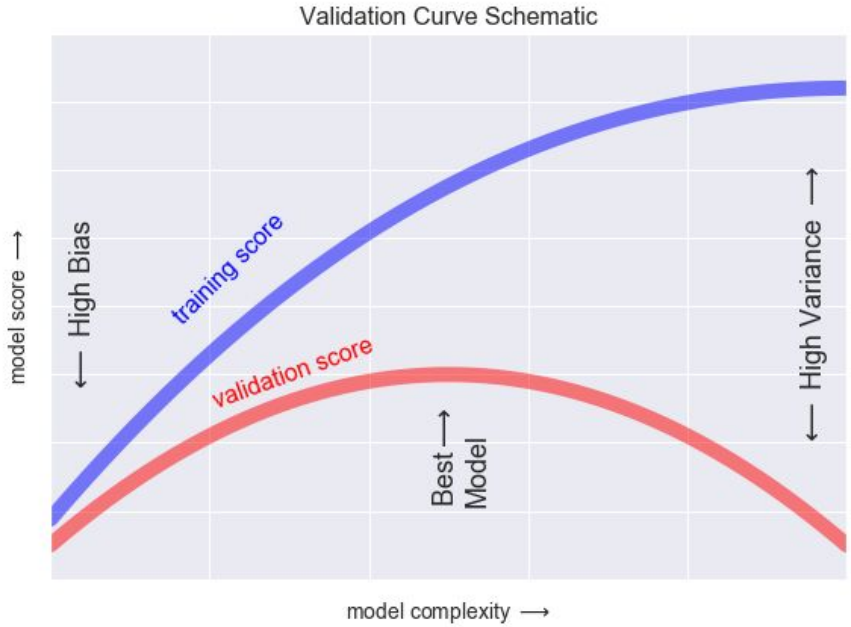
Para elegir el valor de ese hiperparámetro - y también caracterizar mejor el desempeño de nuestro modelo -, es útil obtener las **curvas de validación.**

Una **curva de validación nos muestra el desempeño del modelo,** tanto en el conjunto de entrenamiento como de testeo, en función de un hiperparámetro.

Como en general el hiperparámetro que variemos va a modificar la "complejidad" del modelo (profundidad en árboles, vecinos en KNN, etc.), también nos sirve para diagnosticar overfitting y underfitting.

Curvas de validación (o complejidad)

En general, el desempeño de un modelo depende de **muchos hiperparámetros.** Pero a veces hay uno que es el más importante, el que predomina sobre el resto.



Hands-on training

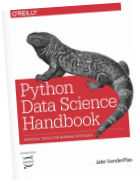
Hands-on training

DS_Clase_20_CV.ipynb

Parte 2

Recursos

Recursos



1. **Por las dudas lo volvemos a mencionar:** Documentación en Scikit-Learn sobre Validación Cruzada.

2. **Capítulo 5, "Machine Learning: Hyperparameters and Model Validation", de Python Data Science Handbook.** Acá van a encontrar una introducción general a ML.

3. **Buenas prácticas en ML:** Nature Article

Para la próxima

1. Trabajar en la Entrega 03
2. Completar el notebook de hoy y/o atrasados.
3. Ver los videos de la plataforma “Ajustes del Modelo”

ACÁMICA