# Introducción a Machine Learning

Juliana Glavina

Curso Quimioinformática en Python
*(basada en la clase de Marina Villacampa)*

10 de Junio de 2025

# ¿Qué es Machine Learning?

**Objetivo.** Usar el conocimiento pasado para aprender cómo realizar una tarea que permita generalizar situaciones futuras del mismo tipo.

**Aprender lo más posible con poco o ninguna intervención humana.**

Es un subcampo de ciencias de la computación y una rama de la inteligencia artificial que estudia algoritmos que pueden hacer cosas interesantes.

El aprendizaje (**learning**) está basado en datos observados. Estos datos pueden ser datos en una hoja de cálculo, un sensor o una lista de instrucciones humanas.

# ¿Qué es Data Mining?

Es el descubrimiento de patrones o relaciones en los datos, y su traducción a una estructura útil.

Los datos en general son considerados muy grandes, posiblemente cambian en el tiempo y muy complicados para que un humano los pueda entender.

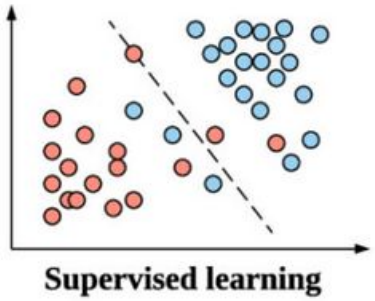
# Tipos de Aprendizaje

**Supervisado**

Se **generaliza** directamente de la experiencia **pasada**.

Se **sabe** que **tarea** hay que resolver como también si se resolvió correctamente o no.

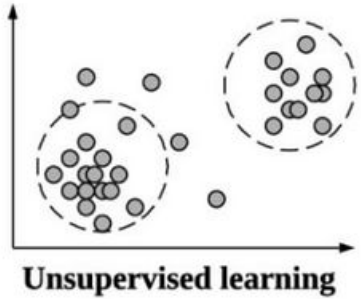
Hay datos observados y valores target observados




**No Supervisado**

Se trata de resolver problemas para los cuales **no existe una etiqueta de información** con la verdad fundamental.

Se trata de **descubrir esa etiqueta** desde los datos directamente.



# Tipos de Aprendizaje

**Aprendizaje semi-supervisado**

Es un tipo de aprendizaje intermedio entre el supervisado y no supervisado.

El set de entrenamiento tiene datos etiquetados y no etiquetados.

Es útil cuando resulta complicado extraer información de grandes volúmenes de datos, por ejemplo en imágenes médicas.

# Tipos de Aprendizaje

**Aprendizaje de Refuerzo**

Es un modelo de aprendizaje conductual, donde el algoritmo recibe retroalimentación del análisis de datos, llevando el mejor resultado.

El sistema **no** está entrenado con un conjunto de datos inicial de entrada, sino que el sistema **aprende** a través de la prueba y el error.

Decisiones exitosas conducen al fortalecimiento del proceso, porque es el que resuelve el problema de manera más efectiva.

# Machine Learning

¿Por donde Empezamos?

1. **Entender el TIPO de problema:** Supervisado? O NO supervisado? Regresión o clasificación?
2. **Conocer cómo son nuestros datos (Data Mining).** Si existen problemas con nuestros datos debemos conocerlos!
3. **Seleccionar el Método adecuado (modificarlo) y aplicarlo**
4. **Evaluación.** Dado un modelo, es necesario **EVALUARLO** (puede generalizar?) usando datos que no hayan sido utilizados para entrenarlo.
5. **¿En qué ejemplos falla? ¿Porqué? ¿Cómo podemos mejorarlo?**

# Machine Learning

¿Por donde Empezamos?

1. **Entender el TIPO de problema:** Supervisado? O NO supervisado? Regresión o clasificación?
2. **Conocer cómo son nuestros datos (Data Mining).** Si existen problemas con nuestros datos debemos conocerlos!
3. **Seleccionar el Método adecuado (modificarlo) y aplicarlo**
4. **Evaluación.** Dado un modelo, es necesario **EVALUARLO** (puede generalizar?) usando datos que no hayan sido utilizados para entrenarlo.
5. **¿En qué ejemplos falla? ¿Porqué? ¿Cómo podemos mejorarlo?**

# Pregunta

En su proyecto de machine learning, si tuvieran que priorizar…

A. Mejorar el algoritmo para entrenar el modelo
B. Mejorar la calidad del conjunto de datos

¿Cuál tendría el mayor impacto?

# Pregunta

En su proyecto de machine learning, si tuvieran que priorizar…

A. Mejorar el algoritmo para entrenar el modelo
**B. Mejorar la calidad del conjunto de datos**

¿Cuál tendría el mayor impacto?

Data trumps all.

La calidad y el tamaño del conjunto de datos es mucho más importante que cualquier algoritmo utilizado para construir el modelo.

# Pregunta

En un proyecto real de machine learning, ¿Cuánto tiempo del proyecto se dedica a la preparación de los datos?

A. Más de la mitad del tiempo que dura el proyecto
B. Menos de la mitad del tiempo que dura el proyecto

# Pregunta

En un proyecto real de machine learning, ¿Cuánto tiempo del proyecto se dedica a la preparación de los datos?

**A. Más de la mitad del tiempo que dura el proyecto**
B. Menos de la mitad del tiempo que dura el proyecto

# Datos Datos Datos

**Cantidad de Datos** → Idealmente 1 o 2 órdenes de magnitud mayor a los parámetros a entrenar.

**Calidad y Confiabilidad**

- Omisión de valores
- Datos duplicados
- Valores mal medidos
- Valores mal etiquetados
- Conjuntos de datos que para una característica a veces son confiables y otras veces no.

# Estandarización de Datos

La estandarización de **variables numéricas** se utiliza para que los valores de distintas características numéricas en el conjunto de datos se encuentren dentro de una escala común, sin distorsionar las diferencias en los rangos de valores ni perder información.

Esto es especialmente útil cuando:

- Los datos tienen variables que varían en escalas,
- El algoritmo a usar asume que todos los datos están centrados alrededor de 0 y tienen una varianza en la misma escala.

Por ej. El rango de **A** es entre -0.5 y 0.5 y el rango de **B** es entre -5 y 5.

La expansión de **B** es 10 veces más que la de **A**, por lo tanto:

1. El modelo asume que **B** es 10 veces más importante que **A**
2. El entrenamiento lleva más tiempo
3. El modelo podría llegar a ser subóptimo

# Estandarización de Datos

Existen distintas formas para estandarizar

**Linear Scaling** Posee una distribución uniforme en un rango fijo

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

**Z-Score** Cuando no hay “outliers” extremos

$$x_{scaled} = \frac{x - \mu}{\sigma}$$

**Log** Particularmente útil cuando los datos siguen una distribución de potencia.

(bajos valores de X tienen valores altos de Y y a medida que aumenta X, Y decae rápidamente)

$$x_{scaled} = ln(x)$$

**Clipping** Cuando hay “outliers” extremos

$$if\ x > max,\ x_{scaled} = max$$

$$if\ x < min,\ x_{scaled} = min$$

# Evaluación

Queremos modelos que aprendan de los datos y que se puedan usar en el futuro.

¿Pero cuán bueno son nuestros modelos frente a datos “nuevos?

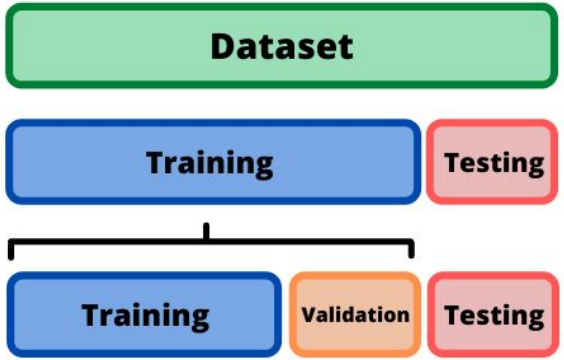
En general, se implementan múltiples modelos y se selecciona el “mejor” ¿Pero cómo elegimos el mejor?

**Partición de los datos**

**Training Set.** Se usa para entrenar al modelo. Durante el entrenamiento se usa la variable de partida y la variable de salida

**Validation Set.** Se usa para seleccionar los mejores modelos (si es que se entrenan distintos modelos) y optimización de parámetros.

**Test Set.** Se usa para evaluar los modelos resultantes.



# Evaluación

Supongamos que tenemos un modelo y se entrenó con el **set de entrenamiento** y se evaluó en el **set de evaluación** durante múltiples rondas. En cada ronda, se usan los resultados de la evaluación para actualizar los parámetros del modelo.

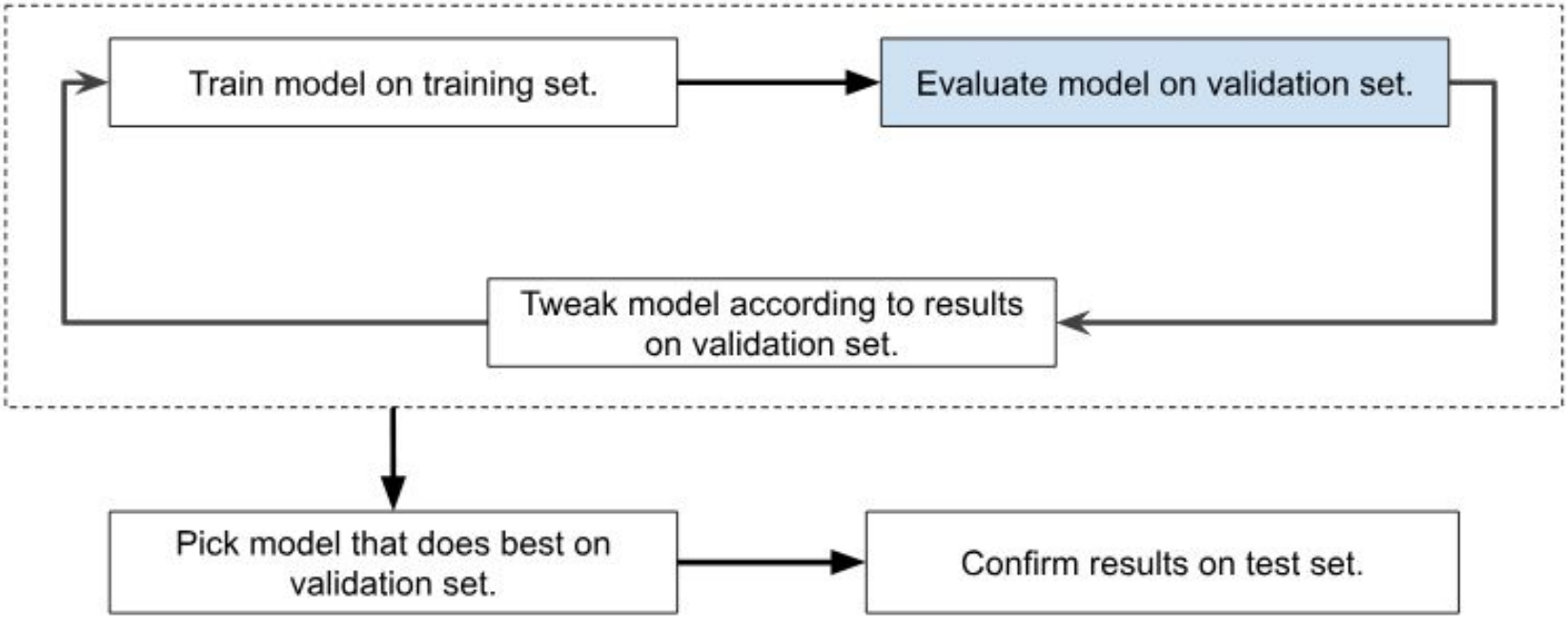
¿Esto es correcto?

# Evaluación

Supongamos que tenemos un modelo y se entrenó con el **set de entrenamiento** y se evaluó en el **set de evaluación** durante múltiples rondas. En cada ronda, se usan los resultados de la evaluación para actualizar los parámetros del modelo.

¿Esto es correcto?

No! Lo ideal es ajustar los parámetros del modelo con el **set de validación** y luego evaluar con el **set de evaluación**. Lo que termina ocurriendo si no es que estamos **sobre-ajustando** los datos (*overfitting*).

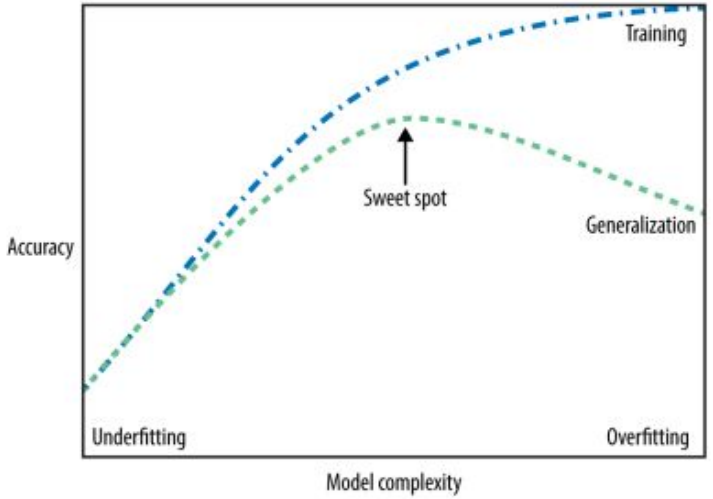

Train model on training set.
Evaluate model on validation set.
Tweak model according to results on validation set.
Pick model that does best on validation set.
Confirm results on test set.

# Overfitting

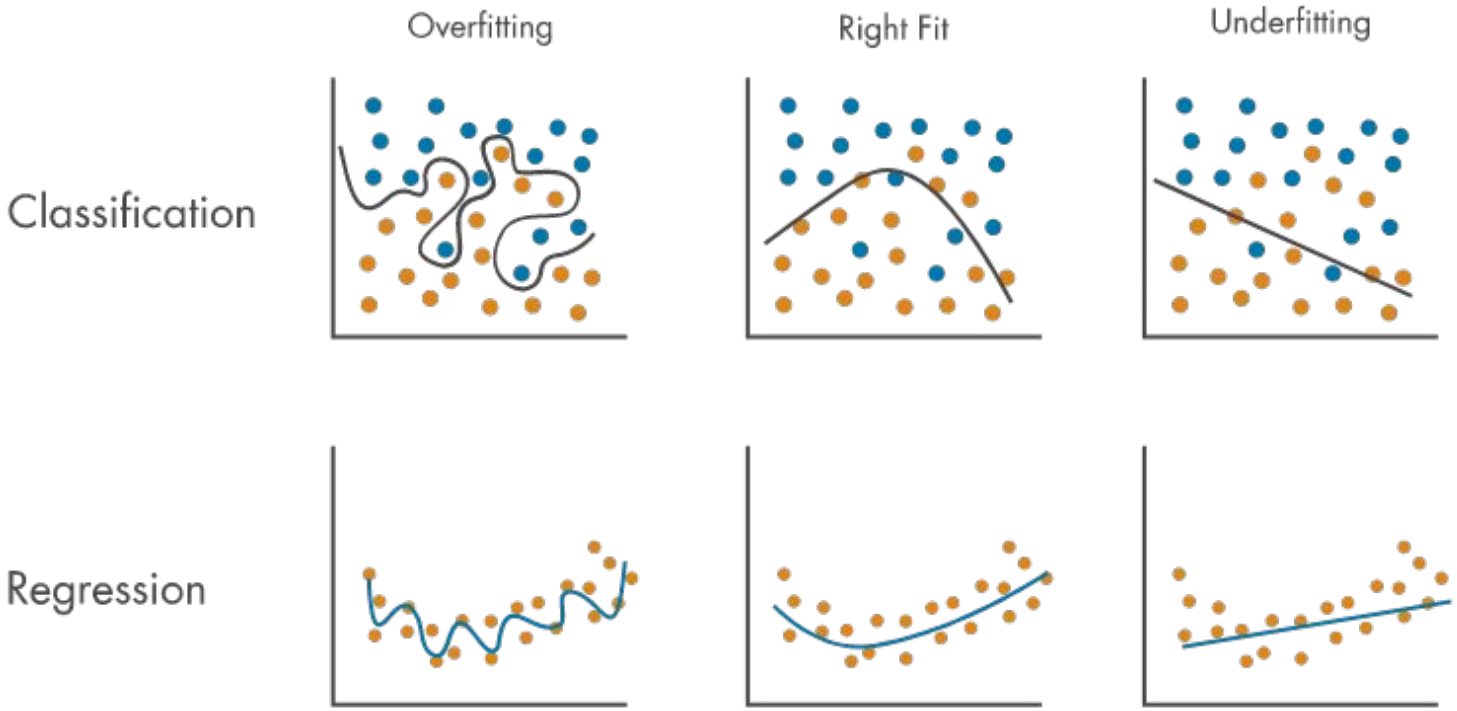
Significa crear un modelo que predice el set de entrenamiento tan bien que falla en hacer predicciones correctas en datos nuevos, es decir, que el modelo *memorizó* los datos de entrenamiento.

**Underfitting**
El modelo ni siquiera predice correctamente el set de entrenamiento

**Generalization**
Un modelo que generaliza puede hacer buenas predicciones en los nuevos datos.



# Overfitting



# Overfitting

**Complejidad del modelo**

Se puede medir como la cantidad, de pesos, parámetros etc, que incluye el modelo.

En función del tipo de datos que estemos usando para entrenar el modelo, será necesario un modelo más o menos complejo

Datos muy sencillos con relaciones simples y/o pocas variables, un modelo más sencillo podrá generalizar mejor que un modelo extremadamente complejo

Datos más complejos con relaciones no lineales, será necesario un modelo algo más complejo

# Overfitting

**Complejidad del modelo**

La única forma de medir si un algoritmo funcionará de forma correcta en datos nuevos es a partir de la evaluación del **set de evaluación**.

Generalmente, siempre se intenta encontrar el **modelo más simple** posible. Pero… si el modelo es demasiado simple, no será capaz de capturar la variabilidad general de los datos, y por tanto el modelo funcionará de forma insuficiente incluso en el training dataset (**underfitting**).

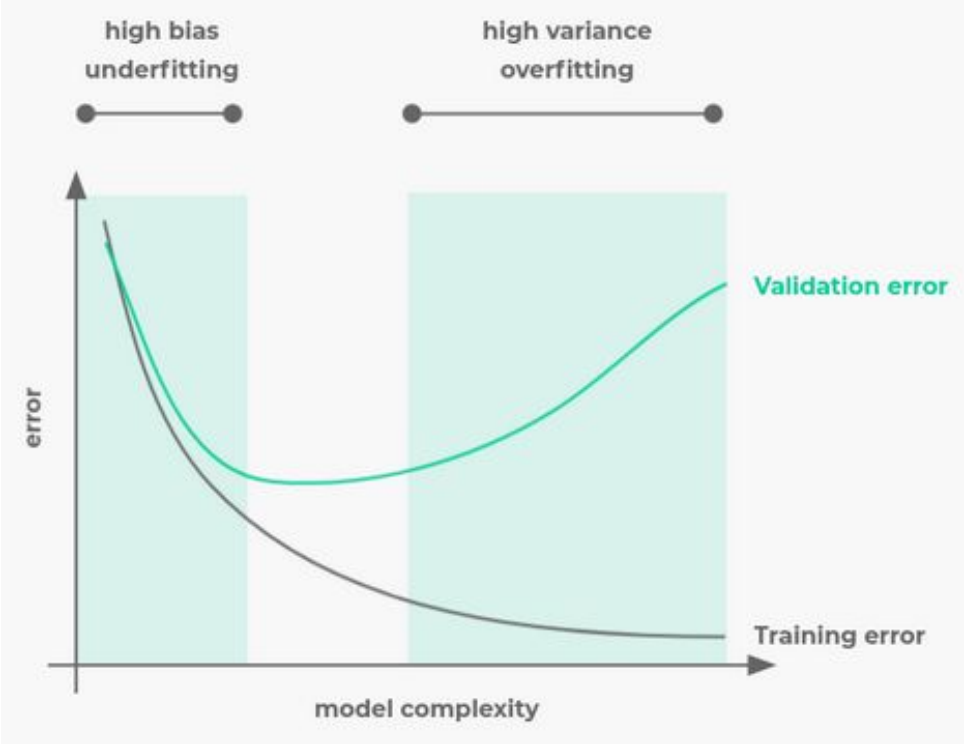
Cuanto más complejo sea el modelo, hará mejor predicciones en datos complejos.

Si el modelo resulta demasiado complejo, perderá la imagen global de los datos y la tendencia real, ajustándose al 'ruido' de los datos, y no será capaz de generalizar a nuevos datos (**overfitting**).

# Overfitting

**Underfitting** puede darse porque el modelo es poco complejo o bien, porque no hay suficientes datos para entrenarlo.

**Overfitting** por el contrario puede darse porque el modelo es demasiado complejo o los datos de entrenamiento son demasiados

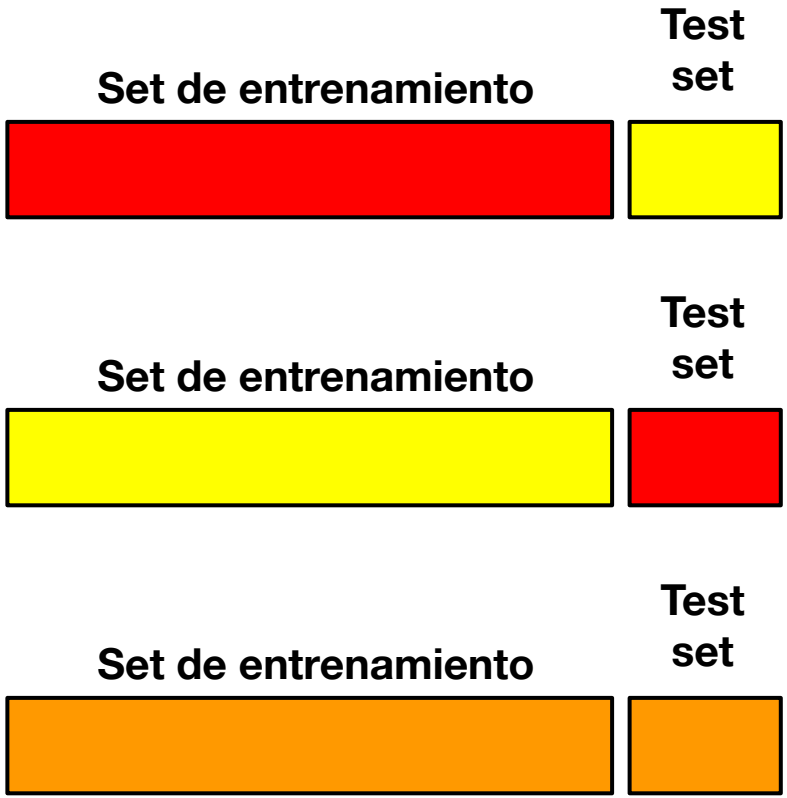


# Cross-Validation

Queda claro que los datos se evalúan en el **set de evaluación** o en el de **validación**, pero no en el de entrenamiento.

Pero… entonces cuán bueno es nuestro modelo va a depender de esa única división. Podría suceder que nuestro modelo mejore si usamos otra división.

Además, esta división es más compleja cuando se tiene una cantidad limitada de datos, ya que se quiere usar lo más posible para entrenar el modelo.

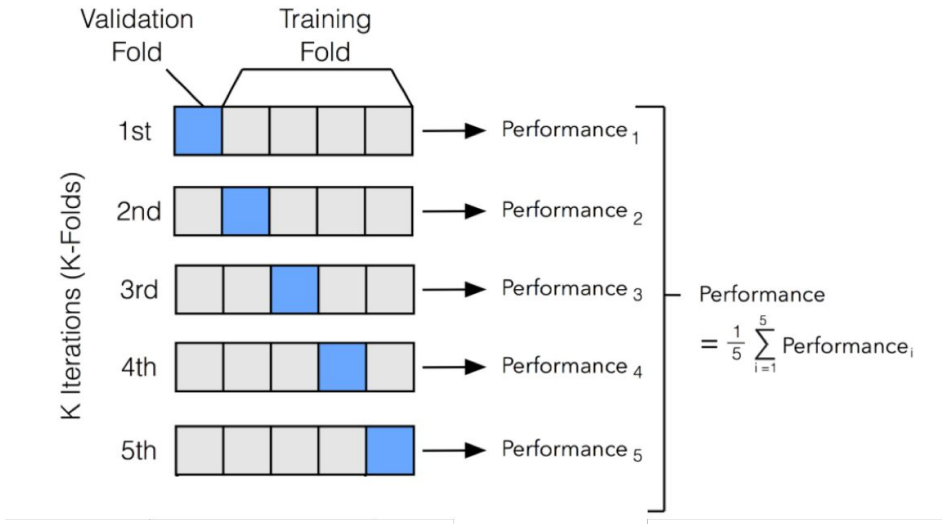


# Cross-Validation

Se puede usar entonces **cross-validation**.

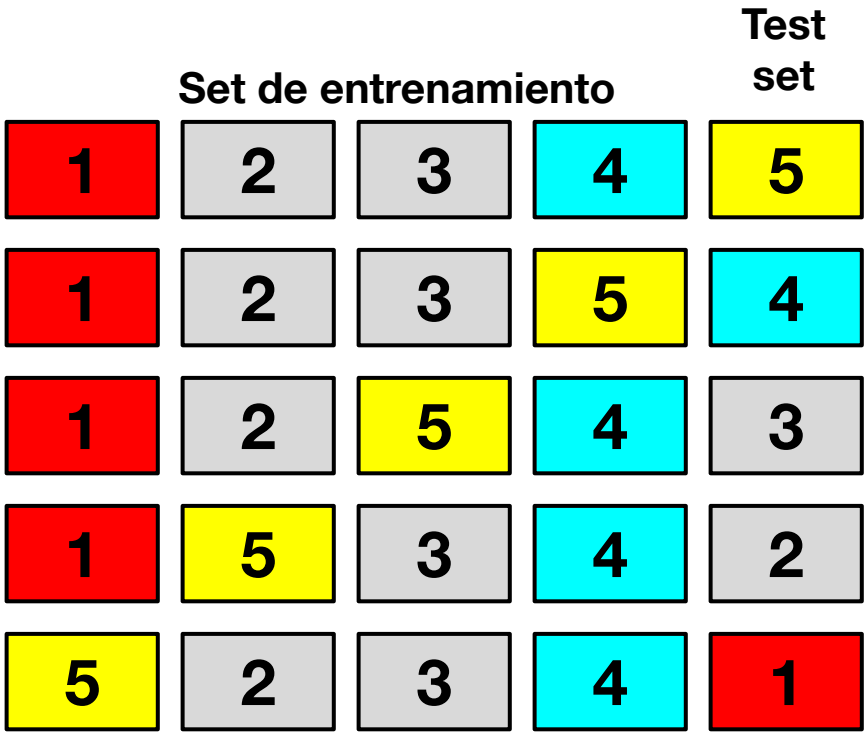
Implica dividir los datos x cantidad de veces en **set de entrenamiento** y **set de validación**.

Para cada partición se mide la *performance* y se toma el promedio para estimar la *performance* global.



# Cross-Validation

Esto nos permite que, si alguna partición está enriquecida en algún tipo de datos, las predicciones de nuestro modelo no dependan de eso.



# Aprendizaje Supervisado

# Aprendizaje Supervisado

El algoritmo produce una **función** que establece una correspondencia entre las **entradas** y las **salidas** deseadas del sistema.

**Regresión**

Dados **valores observados X** se predice una respuesta **continua Y**

Ejemplo.

**Clasificación**

Dados **valores observados X** se predice una respuesta **discreta Y**

Ejemplo.

# Aprendizaje Supervisado

¿Qué tipos de algoritmos existen?

- **Linear Models** → Regresión
- **Logistic Regression** → Clasificación
- **Support Vector Machines (SVM)** → Regresión y **Clasificación**
- **k-Nearest Neighbors (k-NN)** → Regresión y **Clasificación**
- **Decision Trees** → Regresión y **Clasificación**
- Ensembles of Decision Trees
  - **Random forests** → Regresión y **Clasificación**
  - Gradient boosting machines}
- Redes Neuronales

# Evaluación

¿Qué métricas se utilizan para evaluar?

**Regresión**

Mean Absolute Error - MAE

Mean Squared Error - MSE

R cuadrado ($R^2$)

R cuadrado ajustado ($R^2_{Adj}$)

**Clasificación**

Accuracy

Precision

Recall

Área bajo la curva (Area under curve - AUC)

# Aprendizaje No Supervisado

# Aprendizaje No Supervisado

¿Qué tipos de algoritmos existen?

- **Clustering**
- Estimación de Densidad
- Detección de Anomalías
- Association Mining o Rule-Induction
- **Reducción de Dimensionalidad (PCA)**

Ahora al colab !