# Evaluación Práctica 1

## Entrega

- **Fecha y hora**: Domingo 15 de junio de 2025, 22:00.
- **Lugar**: Buzón de esta evaluación en la plataforma del curso.

## Enunciado

Para este taller, deberá realizar un informe dentro de un Jupyter Notebook. Para esto, deberá entregar un solo archivo en formato Jupyter Notebook (.ipynb) con la solución pedida de cada pregunta, tanto texto como código según corresponda. Para facilitar la corrección, no borre las salidas de las celdas y **siempre que sea posible use *random seed* o *random state***. Sin embargo, **el notebook deberá poder ejecutarse desde el inicio hasta el final sin errores**. Considere que esto debe incluir tanto la descarga de datos como la ejecución del código asociado a cada pregunta. **Este taller puede ser realizado tanto en parejas como de forma individual.**

## Preguntas

### Fase 1: Preparación y Exploración de Datos (3 pts.)

#### Pregunta 1: Selección y descarga de dataset [1 pts.]

Busque en Kaggle o algún otro sitio similar un conjunto de datos de cualquier tema para aplicar un análisis exploratorio de datos, tal como fue visto en clases. Este mismo conjunto de datos deberá ser utilizado en preguntas posteriores, por lo que **deberá tener tanto columnas numéricas como categóricas**, e incluir una columna que pueda ser usada como valor a predecir.

**El total de columnas del dataset (incluyendo la columna a predecir) debe ser al menos 10, mientras que el número de filas deberá ser al menos 100**.

Describa cómo obtener el conjunto de datos (incluya la referencia de dónde encontró el dataset), explique qué información contiene el dataset y cuál es la tarea de predicción que realizará. Adicionalmente, deberá descargar los datos de manera programática

(usando el comando gdown, por ejemplo) y cargarlos utilizando Python (con una librería apropiada).

**Algunos datasets de ejemplo**:

- Kaggle | Countries Life Expectancy using linear Regression
- Kaggle | Coffee Quality Data (CQI May-2023)
- Kaggle | Spaceship Titanic
- Kaggle | House Prices - Advanced Regression Techniques

**Lugares recomendados para buscar dataset**:

- Kaggle | Competencias
- Kaggle | Datasets
- Kaggle | Competenciones de la categoría "Getting Started"

### Pregunta 2: Análisis de columnas [1.5 pts.]

Liste cada una de las columnas incluidas en el conjunto de datos, describa qué representa y el tipo de dato asociado a cada una de ellas. Además, para cada columna, describa la naturaleza de su contenido, ya sea por medio de un conteo de valores (columnas categóricas) o graficando la forma en que sus valores se distribuyen (columnas numéricas). Deberá incluir al menos 3 gráficos diferentes que permitan obtener conclusiones interesantes sobre:

- Distribuciones de variables individuales
- Relaciones entre variables
- Patrones o anomalías en los datos

Deberá escribir conclusiones de cada gráfico. Utilice Python y la librería que prefiera para apoyar este análisis.

### Pregunta 3: Tratamiento de valores nulos [0.5 pts.]

Muestre si existe presencia de nulos en el conjunto de datos. En caso de que existan nulos, indique una estrategia para tratarlos, ya sea reemplazandolos con algún valor o eliminando la fila o columna. Justifique su razonamiento para cada tratamiento utilizado (ya sea eliminar filas, alguna columna o completar valores).

**Importante**: Si el conjunto de datos utilizado no tiene valores nulos, esta pregunta no tendrá puntaje asignado (no perjudicará su nota máxima posible).

## Fase 2: Ingeniería de Features (1.5 pts.)

### Pregunta 4: Selección de columnas [0.5 pts]

A partir del análisis anterior, indique claramente cuáles son las columnas que utilizará para la tarea de predicción a realizar en las próximas preguntas. Además, elimine las columnas no significativas y justifique su decisión.

### Pregunta 5: Análisis por visualización de datos [1.0 pts.]

Una vez que ha decidido el conjunto de columnas a utilizar, además de tener claro cuál es la columna correspondiente a la etiqueta a predecir, grafique el conjunto de datos utilizando PCA, t-SNE o UMAP, justificando la elección de la técnica utilizada. Puede utilizar gráficos en 2 o 3 dimensiones. Además de incluir el código apropiado en Python para graficar la reducción de dimensionalidad, indique qué conclusiones puede sacar a partir de este gráfico (por ejemplo, dificultad de la tarea a realizar, clases que se pueden separar más fácilmente, naturaleza de los datos, etc.).

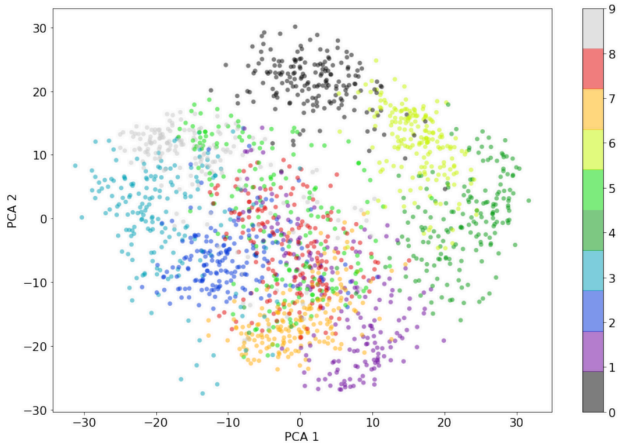


**Figura**. Ejemplo de gráfico esperado al aplicar PCA

**Importante**: Si utiliza PCA, muestre la varianza explicada por componente. Si usa t-SNE/UMAP, justifique los hiperparámetros elegidos (`perplexity`, `n_neighbors`, etc.).

## Fase 3: Modelado y Evaluación (4 pts.)

### Pregunta 6: Partición de datos [0.5 pts.]

Realice un *split* de los datos para entrenar y evaluar los modelos a construir. Para esto, decida según su criterio el porcentaje y la estrategia utilizada para dividir los datos. Justifique ambas decisiones.

### Pregunta 7: Predicción con modelos de ML [2 pts.]

Escoja **dos modelos de Machine Learning de los vistos en clases** (KNN, SVM, Árboles de Decisión, Random Forest, Gradient Boosting, o MLP) y entrenelos utilizando su partición de los datos. Para evaluar los resultados, **utilice el mismo conjunto de datos**

**no vistos para ambos modelos** y calcule métricas que sean relevantes indicando claramente cuáles son las métricas utilizadas. Puede incluir gráficos si le parece conveniente.

**Importante**: No se evaluará el haber logrado el mejor rendimiento posible, aunque sí deberá lograr un rendimiento notablemente mejor que el de un clasificador *dummy* que clasifique al azar o utilizando la etiqueta más frecuente. Puede implementar dicho modelo por su cuenta a través de la clase `DummyClassifier` de scikit-learn.

## Pregunta 8: Sobreentrenamiento [1.5 pts.]

Entre los modelos utilizados (los elegidos en la pregunta anterior), elija el modelo que mejor permita visualizar *overfitting*, y muestre cómo se produce sobreentrenamiento a medida que aumenta su complejidad. Elija y justifique cuál parámetro varía para esto, pero solo debe ser uno. Grafique los rendimientos en los sets de entrenamiento y test a medida que aumenta la complejidad del modelo, usando la métrica de rendimiento *balanced accuracy* para evitar problemas en caso de desbalance de clases.

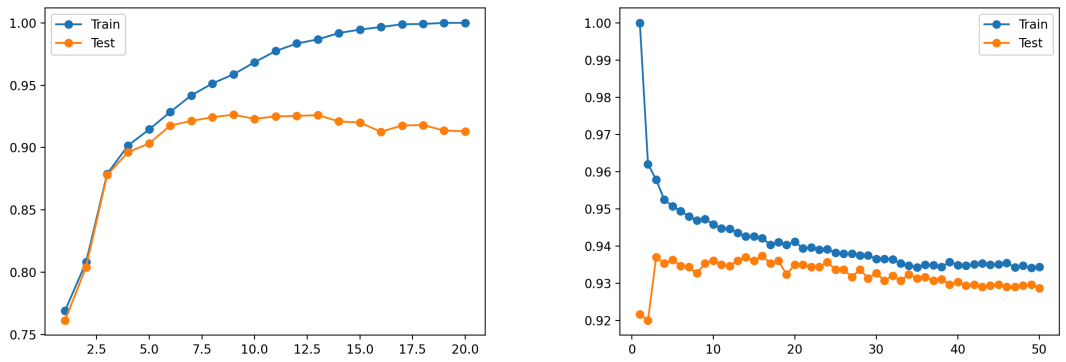


**Figura**. Ejemplo de gráficos esperados para evidenciar sobreentrenamiento

**Parámetros sugeridos para cada modelo**:

- Random Forest: `n_estimators` o `max_depth`
- SVM: C (regularización)
- KNN: `n_neighbors`
- Árboles: `max_depth`

**Importante**: Si la métrica sugerida no permite visualizar correctamente este efecto, puede utilizar una métrica distinta que muestre el efecto esperado. Por ejemplo, para clasificación puede usar *accuracy*, *precision*, *recall* o F1, mientras que en el caso de regresión puede usar MAE, MSE o $R^2$.

# Descuentos posibles

Las siguientes situaciones podrían provocar que se descuente hasta 1 punto de la nota obtenida:

- Código complejo (no intuitivo) mal documentado o sin comentarios explicativos
- Gráficos (no intuitivos) sin títulos, etiquetas o interpretación
- No incluye instalación de todas las dependencias
- Archivo no se ejecuta de forma continua (sin errores)
- No usar *random seed* o *random state* para reproducibilidad
- Otras razones que dificulten (o imposibiliten) la corrección