Desafío - Regresión desde el aprendizaje de máquinas

- Para realizar este desafío debes haber estudiado previamente todo el material disponibilizado correspondiente a la unidad.
- Una vez terminado el desafío, comprime la carpeta que contiene el desarrollo de los requerimientos solicitados y sube el `.zip` en el LMS.
- Desarrollo desafío:
    - El desafío se debe desarrollar de manera Individual.
    - Para la realización del desafío necesitarás apoyarte del archivo *Apoyo Desafío - Regresión desde el aprendizaje de máquinas*.

Contexto

En esta sesión trabajaremos una base de datos sobre los precios de las viviendas en Boston, utilizada en el paper Harrison Jr, D., & Rubinfeld, D. L. (1978). Hedonic housing prices and the demand for clean air. Journal of environmental economics and management, 5(1), 81-102.

Nuestro objetivo es desarrollar un modelo predictivo para el valor mediano de las casas mediante el entrenamiento de un modelo de regresión lineal.

- crim : Tasa de criminalidad por sector de Boston.
- `zn` proporción de terreno residencial asignado para terrenos baldíos.
- `indus` proporción de negocios no asociados al comercio por sector.
- `chas` Dummy. 1 si el sector colinda con el río Charles, 0 de lo contrario.
- `nox` Concentración de dióxido de carbono.
- `rm` cantidad promedio de habitaciones por casa.
- `age` proporción de casas construidas antes de 1940.
- `dis` distancia promedio a cinco centros de empleos.
- `rad` índice de accesibilidad a autopistas.
- `tax` nivel de impuestos asociados a viviendas.
- `ptratio` razón alumno:profesor por sector de Boston.
- `black` proporción de afroamericanos por sector de Boston.
- `lstat` porcentaje de población de estratos bajos.
- `medv` valor mediano de las casas

Desafío 1: Prepare el ambiente de trabajo

- Importe las librerías básicas para el análisis de datos.
- Importe el módulo `linear_model` , y las funciones `mean_squared_error` , `r2_score` y `train_test_split` .
- Importe la base de datos `boston.csv` y elimine la columna `Unnamed: 0` .
- Obtenga las medidas descriptivas de la base de datos con `.describe()` .

Desafío 2: División de la muestra

- Genere conjuntos de entrenamiento y pruebas con `train_test_split` .
- Reserve un 33% de la muestra para el conjunto de pruebas.
- Incluya una semilla pseudoaleatoria a su elección, esto lo puede hacer con el argumento `random_state` dentro del método `train_test_plit`.

Desafío 3: Generación de modelos

- Ahora implementaremos dos versiones del modelo lineal:
    - Con intercepto.
    - Sin intercepto.
- Cada versión debe generarse en un nuevo objeto inicializado.
- Posteriormente se deben entrenar los modelos especificando la matriz y vector de entrenamiento.
- Con los modelos entrenados, genere una predicción de la matriz de pruebas con el método `.predict()`.

Desafío 4: Obtención de métricas

- Ahora generaremos una función llamada `report_scores` que ingrese como argumentos el vector de datos predichos y el vector de datos por validar.
- La función debe imprimir las métricas del Error Cuadrático Promedio y R2.
- Reporte las métricas para ambos modelos. En base a ello, seleccione el mejor modelo.

Desafío 5: Refactorización del modelo

- Genere una función llamada `fetch_features` que ingrese como argumentos la base de datos y el nombre del vector objetivo. El nombre del vector debe ser `medv` por defecto.
- La función debe retornar una lista con las correlaciones entre cada atributo y el vector objetivo y su nombre.
- Reporte brevemente cuales son los 6 atributos con una mayor correlación absoluta con `medv` (de mayor a menor correlación).

Desafío 6: Refactorización del modelo predictivo

- Genere otros conjuntos de entrenamiento y validación en base a una matriz con los 6 atributos identificados en el ejercicio anterior, y el vector objetivo.
- Entrene un modelo en base al mejor desempeño.
- Reporte las métricas para el nuevo modelo.

Desafío 7: Predicción de casos

- A continuación se generaron dos arrays que representan el peor escenario posible (`worst_neighbor`) y el mejor escenario posible (`best_neighbor`). Las variables representan, para cada caso, los valores de los siguientes atributos (en el mismo orden entregado): `'lstat', 'rm', 'ptratio', 'indus', 'tax', 'nox'`.

```
worst_neighbor = np.array([37.9, 12.6, 3.5, 27.7, 187, 0.87]).reshape(1,
-1)
best_neighbor = np.array([1.73, 22, 8.7, 0.46, 711, 0.38]).reshape(1,
-1)
```

- Ingrese los arrays en el modelo entrenado en el ejercicio anterior, y reporte la predicción entregada por el modelo.