Desafío - Árboles de Regresión

- Para realizar este desafío debes haber revisado la lectura y videos correspondiente a la unidad.
- Crea una carpeta de trabajo y guarda todos los archivos correspondientes (notebook y csv).
- Una vez terminado el desafío, comprime la carpeta y sube el `.zip` a la sección correspondiente.

Descripción

- Para esta sesión trabajaremos con una base de datos sobre los precios de inmuebles en la ciudad de Ames, Iowa. La base se compone de 2930 registros y contiene un gran número de atributos.
- Nuestro objetivo es generar un modelo que prediga de forma adecuada los precios de inmuebles, medidos con la variable `Sale_Price`.

Ejercicio 1: Preparación del ambiente de trabajo

- Importe las librerías clásicas a utilizar.
- Para este ejercicio implementaremos árboles de regresión, por lo que deberá importar la clase `DecisionTreeRegressor`.
- De manera adicional importe las funciones y clases necesarias para generar un desempeño de métricas en problemas de regresión, división de muestras y búsqueda de grilla con validación cruzada.
- Elimine la columna `'Unnamed: 0'` cuando cargue los datos.

Ejercicio 2: Feature engineering

- Identifique si el `dtype` de cada `pd.Serie` en nuestra base de datos se considera `'object'` o no. Para todas las variables que sean `'object'`, realice lo siguiente:
    1. Genere una recodificación $K-1$ en cada variable. Para efectos prácticos sólo necesitan eliminar una de las categorías, no se concentren en especificar la categoría a eliminar. Pueden utilizar la función `pd.get_dummies` con la opción `drop_first` para ello.
    2. Utilizando el método `pd.concat`, concatene a los atributos creados en la base de datos.
    - *tip*: No se olvide de eliminar los atributos recodificados, de esta forma evitará un aumento artificial del desempeño del modelo.

Ejercicio 3: Primer modelo

- Genere muestras de entrenamiento y validación con `'Sale_Price'` como vector objetivo y los atributos de la base de datos como matriz.
- Recuerde definir el porcentaje de casos en la muestra de validación y una semilla pseudoaleatoria.
- Posteriormente, entrene un árbol de regresión en la muestra de entrenamiento sin modificar los hiperparámetros. Reporte las principales métricas de desempeño.
- Comente sobre el desempeño.

Ejercicio 4: Importancia relativa

- Implemente el método `plot_importance` utilizado en la lectura para reportar la importancia relativa de los atributos.
- Comente sobre cuáles son los principales 10 atributos que afectan la predicción de `Sale_Price`.
- Separe éstos 10 atributos en una nueva base de datos, junto con el vector objetivo.

Ejercicio 5: Refactorización del modelo y *pickling*

- En función de los atributos seleccionados en el ejercicio anterior, vuelva a generar conjuntos de entrenamiento y validación.
- **Dentro de los datos de entrenamiento** genere una búsqueda de grila con `GridSearchCV` utilizando los siguientes hiperparámetros:
  - Máximo de atributos: Evalúe todos los posibles atributos.
  - Máximo de profundidad: entre 1 a 32.
  - Validaciones cruzadas : 5.
- Reporte la mejor combinación de hiperparámetros y su desempeño asociado. Compare el desempeño **en la muestra de validación** con el modelo por defecto.

Pickling

- Ahora generaremos una serialización de nuestro modelo depurado, y nuestros conjuntos de entrenamiento y validación depurados. Para ello importe el módulo `pickle`.
- `pickle` contiene la función `dump`, que permite guardar el modelo desarrollado. La forma canónica para desarrollar el pickling es:

```
pickle.dump(<OBJETO_CON_EL_MODELO>, open('nombre-apellido-actividad07.sav',
'wb'))
```

- Envíe su modelo y conjuntos de entrenamiento a un compañero.