Boletín 1: evaluación y selección de modelos
Boletín 2: métodos basados en vecinos más próximos

Para la realización de las prácticas correspondientes a este boletín se utilizará scikit-learn, una librería de aprendizaje estadístico en Python.

En primer lugar, abre mediante *ipython notebook* el fichero **introduction.ipynb**, donde se describen algunas de las operaciones básicas necesarias para trabajar con scikit-learn: aprenderás a cargar los datos, realizar operaciones básicas con matrices, y representaciones gráficas.

A continuación abre el fichero **knn.ipynb**. En este archivo se realiza la experimentación con un algoritmo sobre un conjunto de datos. Concretamente, se ha escogido el método de vecinos más cercanos, y un archivo con un problema muy simple (*toyExample.data*). Los pasos que se realizan son los siguientes:

- Carga de datos y preprocesado básico.
- División del conjunto de datos en entrenamiento y test.
- Generación de los datos sobre los que se harán las representaciones gráficas.
- Búsqueda de los mejores valores para los hiper-parámetros mediante validación cruzada.
- Generación del modelo final, test y representación gráfica.
- Guardar el modelo aprendido.

1. Dado el siguiente conjunto de datos de clasificación con 6 observaciones, 3 variables de entrada y una variable de salida:

Observación	X_1	X_2	X_3	Y
1	0	3	2	1
2	3	0	3	0
3	0	3	-1	0
4	3	0	0	1
5	1	2	1	1
6	2	1	0	0

Suponiendo que se quiere hacer la predicción de la variable de salida para X_1=0, X_2=0, X_3=0 mediante KNN.

a. Computar la distancia entre cada observación y el punto de test.

b. ¿Cuál es la predicción para K=1? ¿Por qué?

c. ¿Cuál es la predicción para K=3? ¿Por qué?

Nota: este ejercicio debe hacerse sin utilizar ninguna función de scikit-learn.

2. Dado el problema de clasificación Blood Transfusion Service Center:

a. Analiza las características del conjunto de datos: número y tipo de variables de entrada y salida, número de instancias, número de clases y distribución de las mismas, valores perdidos, etc.

b. Una de las clases que implementa el algoritmo KNN en scikit-learn es *sklearn.neighbors.KNeighborsClassifier*. Revisa los parámetros y métodos que tiene.

c. Divide los datos en entrenamiento (80%) y test (20%).

d. Realiza la experimentación con KNN (*KNeighborsClassifier*) usando como hiper-parámetro el número de vecinos.

   i. Muestra la gráfica del error de entrenamiento con validación cruzada (5-CV) frente al valor del hiper-parámetro, y justifica la elección del valor más apropiado.

   ii. Muestra la gráfica del error de test frente al valor del hiper-parámetro, y valora si la gráfica del error de entrenamiento con validación cruzada ha hecho una buena estimación del error de test.

3. Repite el ejercicio 2 pero para el problema de regresión Energy Efficiency con la variable de salida *cooling load*. Al ser un problema de regresión deberás utilizar *KNeighborsRegressor*, y como medida de error de entrenamiento y test el MSE.

   Nota: para estimar tanto el error de entrenamiento como el de test es necesario *desestandarizar* los errores calculados.