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

Para la realización de las prácticas de esta segunda parte de la materia se utilizará scikit-learn, una librería de aprendizaje estadístico en Python, a través de **Jupyter Notebooks**. La ejecución se realizará mediante un contenedor Docker. Para ello debes seguir los siguientes pasos:

1. Ejecuta Docker. El contenedor que lanzaremos utilizará como base la siguiente imagen: jupyter/scipy-notebook:6d42503c684f. Podéis descargarla ejecutando el comando "docker pull jupyter/scipy-notebook:6d42503c684f".

2. Lanza el contenedor. Para ello, desde una terminal ejecuta el siguiente comando:

    a. Linux: docker run -it --rm -v "${PWD}":/home/jovyan -p 8888:8888 jupyter/scipy-notebook:6d42503c684f start-notebook.sh --NotebookApp.token=''

    b. Windows: docker run -it --rm -v "Dirección de la carpeta":/home/jovyan/work -p 8888:8888 jupyter/scipy-notebook:6d42503c684f start-notebook.sh --NotebookApp.token=''

        i. Dirección de la carpeta representa la ruta donde tenéis los Jupyter Notebooks. Por ejemplo: C:\Users\Manuel Mucientes\Desktop\Aprendizaje_Estadistico

    Con este comando le indicamos al contenedor que mapee el puerto del servidor de Jupyter Notebook (puerto `8888`) a un puerto de nuestro ordenador (el "8888", por ejemplo). También le indicaremos que monte nuestro directorio actual (`${PWD}`) en vez del directorio raíz del contenedor (`/home/jovyan`); de esta manera el servidor podrá ver nuestros archivos y editarlos. Adicionalmente, haremos que no sea necesario autenticarnos a la hora de conectarnos al servidor (--NotebookApp.token=''), para evitar posibles problemas con tokens y sesiones. Por último, queremos ejecutar el contenedor interactivamente (`-it`) y que éste se elimine una vez cerrado (`--rm`), para minimizar conflictos.

3. Entramos al servidor de Jupyter Notebook a través de un navegador, utilizando la siguiente dirección: http://127.0.0.1:8888

4. Finalmente, para cerrar el contenedor —recuerda guardar los cambios en los Notebooks— pulsaremos "Control-C" en la terminal en la que ejecutamos el comando.

introduction.ipynb

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.

knn.ipynb

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.

Instrucciones para la experimentación en TODOS los boletines de prácticas

En los diferentes ejercicios que se realizarán durante el curso, existen una serie de operaciones con una componente aleatoria: la división en entrenamiento y test, el aprendizaje de un modelo o incluso, en algunos casos, el test del modelo. Como norma general de experimentación es interesante asegurar la repetibilidad de los experimentos, eliminando la aleatoriedad, puesto que nos permite depurar errores, comparar modelos, etc. Además, para la evaluación de los boletines también es imprescindible eliminar esa aleatoriedad.

Para ello vamos a fijar la semilla del generador de números aleatorios, de tal manera que su secuencia sea siempre la misma. La semilla se establece mediante el comando **np.random.seed(SEED_VALUE)**, y en este boletín utilizaremos un **SEED_VALUE=1**. Será necesario utilizar este comando inmediatamente antes de cualquier operación con un componente aleatorio. Esto incluye: train_test_split(), fit(), predict(), etc. En aquellas funciones que lo admitan, sustituiremos el comando np.random.seed(SEED_VALUE) por el argumento **random_state=SEED_VALUE**.

Boletín

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. No es necesario estandarizar las variables.

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, correlación entre las variables, 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.

      Muestra la gráfica del error de entrenamiento con validación cruzada (5-CV) frente al valor del hiper-parámetro. ¿Cuál es el menor error de validación cruzada, su desviación estándar y el valor del hiper-parámetro para el que se consigue? ¿Cuál es el valor del hiper-parámetro si se aplicase la regla de una desviación estándar?

      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. ¿Cuál es el menor error de test y el valor del hiper-parámetro para el que se consigue? ¿Cuál es el error de test para el valor del hiper-parámetro seleccionado por la validación cruzada? ¿Cuál es el error de test para el valor del hiper-parámetro seleccionado por la validación cruzada mediante la regla de una desviación estándar?

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: al ser un problema de regresión, para estimar tanto el error de entrenamiento como el de test es necesario *desestandarizar* los errores calculados.

Entregable

Se debe entregar un único fichero comprimido con el nombre *PrimerApellido_SegundoApellido.zip* (también son válidos los formatos .rar y .7z), que contenga dos archivos:

- El primer archivo debe ser de tipo pdf, y contendrá exclusivamente las respuestas a los ejercicios (incluyendo las gráficas necesarias para justificar dichas respuestas). No se incluirá en este archivo ningún otro tipo de texto.

- El segundo archivo será de tipo ipynb, y permitirá reproducir toda la experimentación realizada en el boletín.