Fundamentos de Aprendizaje Automático 2020/2021

PRÁCTICA DE INTRODUCCIÓN

1. Objetivos

El objetivo de esta práctica es avanzar en la implementación de la clase *Datos*, una de las que se empleará durante el curso para gestionar los diferentes datasets con los que probar los algoritmos de clasificación a estudiar.

2. Preliminares

Las prácticas se realizan en Python y se puede emplear la distribución que se considere oportuna, teniendo en cuenta que necesitaremos al menos los paquetes Numpy y Scikit-learn. Por otro lado, las entregas deben incluir un Jupyter Notebook, que presente de forma conjunta las implementaciones de código realizadas y la discusión de los resultados obtenidos. De esta forma, no es necesaria una memoria en otro formato, salvo para casos puntuales.

Las diapositivas disponibles en *Moodle* proporcionan una introducción a Python (variables y tipos de datos, funciones, secuencias de control, ficheros, clases, módulos, paquete NumPy, …). Si no se tiene experiencia previa con el lenguaje, se recomienda leer esta introducción con detalle y hacer los ejercicios propuestos para adquirir los conocimientos básicos necesarios para implementar la primera de las clases pedidas. Son especialmente importantes los ejercicios que se proponen con el paquete NumPy para comprender las particularidades del trabajo con matrices en estas prácticas.

3. Clase Datos

Los algoritmos de aprendizaje automático permiten construir modelos a partir de un conjunto de datos (datasets). En las prácticas utilizaremos algunos datasets del repositorio de la Universidad de California Irvine (UCI)[1]. Estos datasets se proporcionarán ya adaptados para su uso en las prácticas en Moodle. A grandes rasgos, son ficheros en formato csv, donde la primera fila se destina al nombre de los atributos.

Como ejemplo, en esta práctica se proporcionan en *Moodle* los ficheros correspondientes a los conjuntos de datos ***tic-tac-toe*** (http://archive.ics.uci.edu/ml/datasets/Tic-Tac-Toe+Endgame) y ***statlog*** (http://archive.ics.uci.edu/ml/datasets/Statlog+%28German+Credit+Data%29).

En esta práctica nos ocuparemos de implementar el tratamiento preliminar de estos dataset mediante la clase *Datos*, que leerá los datos del fichero de entrada y almacenará la información necesaria para su posterior uso por los algoritmos de aprendizaje automático y métodos de particionado. Una posible estructura de implementación se comenta en el siguiente apartado.

[1] http://archive.ics.uci.edu/ml/

3.1 Diseño

Con el objetivo de desarrollar una aplicación lo más flexible y general posible se plantea la siguiente estructura para la clase *Datos*:

```
import pandas as pd
import numpy as np

class Datos:

  # TODO: procesar el fichero para asignar correctamente las variables
nominalAtributos, datos y diccionario
  def __init__(self, nombreFichero):

  # TODO: implementar en la practica 1
  def extraeDatos(self, idx):
    pass
```

En esta práctica de introducción se deberá implementar únicamente el constructor de la clase *(__init__)* que recibe como parámetro el nombre del fichero de datos. La función *extraeDatos* se implementará en la práctica 1, por lo que no hay que desarrollarla ahora (la sentencia *pass* permite diferir la implementación).

Se pueden definir en la clase Datos los atributos que se precisen, pero no debe modificarse la signatura de los métodos. Se recomiendan al menos los siguientes atributos:

Los atributos de la clase *Datos* deberán guardar la siguiente información:

- **`nominalAtributos`**: Lista de valores booleanos con la misma longitud que el número de atributos del problema (incluyendo la clase) que contendrá *True* en caso de que el atributo sea nominal y *False* en caso contrario. Esta estructura será útil posteriormente también para el uso del paquete Scikit-learn. En los datasets proporcionados, trabajaremos generalmente con tres tipos de datos: nominales, enteros o números reales. En función de este tipo asignaremos True si es nominal y False si es entero o real. Si algún dato no corresponde a uno de estos tres tipos, se deberá informar del error. Por ejemplo, se puede lanzar una excepción del tipo `ValueError`
- **`datos`**: Array bidimensional que se utilizará para almacenar los datos. Se recomienda que esta estructura sea un array Numpy, bien directamente, bien a través del uso de Data Frames de la librería Pandas (https://pandas.pydata.org/pandas-docs/stable/user_guide/dsintro.html). El uso de NumPy facilitará el indexado de los datos, así como el uso del paquete Scikit-learn.
- **`diccionario`**: Actúa como un conversor de datos nominales a datos numéricos, ya que la mayoría de los algoritmos de clasificación trabajan mejor con este tipo de dato. El diccionario contendrá para cada uno de los atributos del dataset, un conjunto de pares *clave-valor*, donde *clave* es el dato nominal original del dataset y *valor* es el valor numérico que asociamos a ese dato nominal. Si el atributo en el dataset original ya es un valor continuo, su conjunto de pares clave-valor estará vacío. En el caso de las variables nominales, el diccionario establecerá la relación entre los valores nominales o categóricos (claves) y un entero (valor). Para garantizar que este mapeo es consistente para diferentes ficheros y permutaciones de los datos, los enteros deben asignarse en orden lexicográfico de las claves. Así, por ejemplo, en el caso del conjunto de datos tic-

tac-toe, donde el primer atributo puede tomar los valores 'x', 'b', 'o', el diccionario correspondiente deberá ser: `{'b': 0, 'o': 1, 'x': 2}`. Una vez obtenidos los diccionarios de cada atributo, las variables categóricas deberán ser codificadas a los valores enteros asignados en el diccionario para su posterior almacenamiento en el array `datos`. Como veremos a lo largo de las prácticas, el uso de estas matrices, con valores numéricos, facilita mucho la programación y acelera la ejecución de los algoritmos. En este sentido, la combinación entre la matriz `datos` y el `diccionario` nos permite saber cuáles son los valores categóricos originales asociados a cada valor numérico y usar ese valor numérico para los cálculos matemáticos.

La clase se definirá en el módulo *Datos* (fichero **Datos.py**). De esta forma, se pueden instanciar elementos de la clase como sigue, dentro de un módulo Main (fichero **MainDatos.py**) que prueba esta clase inicial y cualquiera de las posteriores.

```
from Datos import Datos
dataset=Datos('<datasets-home>/tic-tac-toe.data')
```

Tanto la plantilla de la clase Datos, como el módulo Main se encuentran disponibles en Moodle.

4. Fecha de entrega y entregables

Semana del 28 de Septiembre al 4 de Octubre de 2020. La entrega debe realizarse antes del comienzo de la clase de prácticas correspondiente. Se deberá entregar un fichero comprimido .zip con nombre **`FAAINTRO_<grupo>_<pareja>.zip`** (ejemplo `FAAINTRO_1461_1.zip`) y el siguiente contenido:

1. Código Python (**Datos.py**) con la implementación de la clase Datos
2. Jupyther Notebook para probar la clase

Esta práctica se evalúa como APTO/NO APTO,