Implementación algoritmo K-Neighbours (Python)

Autor: Pascual Andrés Carrasco Gómez

Corpus

El corpus utilizado en este trabajo es uno de los corpus pioneros en *Machine Learning* (ML) conocido como el problema de clasificación de flores Iris. Las flores Iris se clasifican en tres tipos: “Versicolor”, “Setosa” y “Virginica”. De cada flor se realiza un proceso de extracción de características que consiste en la medición del tamaño de los sépalos y el tamaño de los pétalos. Por lo tanto de cada flor (fila del corpus) tenemos 5 elementos:

Largo sépalo	Ancho sépalo	Largo pétalo	Ancho pétalo	Clase
6.1	2.6	5.6	1.4	virginica

Información acerca del corpus: https://es.wikipedia.org/wiki/Iris_flor_conjunto_de_datos

El corpus utilizado esta formado por 150 muestras etiquetadas que corresponde al fichero “iris.dat”. Las muestras del corpus se han barajado (utilizando una semilla para que las ejecuciones sean deterministas) y posteriormente se ha realizado una partición formada por 120 muestra para entrenamiento (*train*) y 30 muestra para evaluación (*test*).

Algoritmo k-neighbours

La implementación del algoritmo corresponde al fichero python “kneighbours.py”, el cual realiza el preproceso del corpus Iris, clasifica con el algoritmo *kneighbours* las muestras de *test* y por último realiza una evaluación de dichas muestras respecto a su clase real. La evaluación consiste en el calculo de las medidas de evaluación *coverage*, *precision*, *recall* y *f1-score* las cuales están explicadas en mi tesis de máster (pág 25): http://personales.alumno.upv.es/pascargo/pdf/puns.pdf

El fichero python “kneighbours.py” se ejecuta de la siguiente manera:

```
python3 kneighbours.py
```

La salida por pantalla es la siguiente:

```
Uso: python3 kneighbours.py <data.dat> <k>
```

Nos indica que al programa python hemos de indicarle como entrada el corpus y el parametro k del algoritmo *kneighbours*. Si queremos utilizar el vecino más cercano (k=1) la ejecución se realiza de la siguiente manera:

```
python3 kneighbours.py iris.dat 1
```

Y la salida de la ejecución anterior (k=1) es la siguiente:

```
------------------------------------------------
Resultados:
------------------------------------------------
        Aciertos: 28
        Muestras analizadas: 30
        Muestras totales: 30
------------------------------------------------
Medidas de evaluacion:
------------------------------------------------
        Coverage: 1.0
        Precision: 0.9333333333333333
        Recall: 0.9333333333333333
        F1-score: 0.9333333333333333
```

Donde obtenemos una *precision* = 93.33%, *recall* = 93.33% y *f1-score* = 93.33% debido a que tenemos 28 aciertos de 30 muestras analizadas sobre 30 muestras totales (test).

Gráfica

Es interesante estudiar como influye el parámetro k con la partición que hemos realizado sobre el corpus de flores Iris. Por este motivo se ha implementado un programa en *shell* (bash) "grafica.sh" que realiza un barrido del parámetro k = [1...30] y dibuja un gráfico mostrando la precisión respecto al parámetro k.

Para ejecutar este *script* es necesario tener instalado *gnuplot* en Linux que es la herramienta encargada de plotear el gráfico. Mi sistema operativo es Linux Mint 18.2 e instalando el paquete "gnuplot" es suficiente, pero en sistemas operativos más antiguos es necesario instalar el paquete "gnuplot-x11" ya que no instala por defecto el paquete gráfico para dibujar las gráficas.

El *script* "grafica.sh" se ejecuta de la siguiente manera:

```
./grafica.sh
```

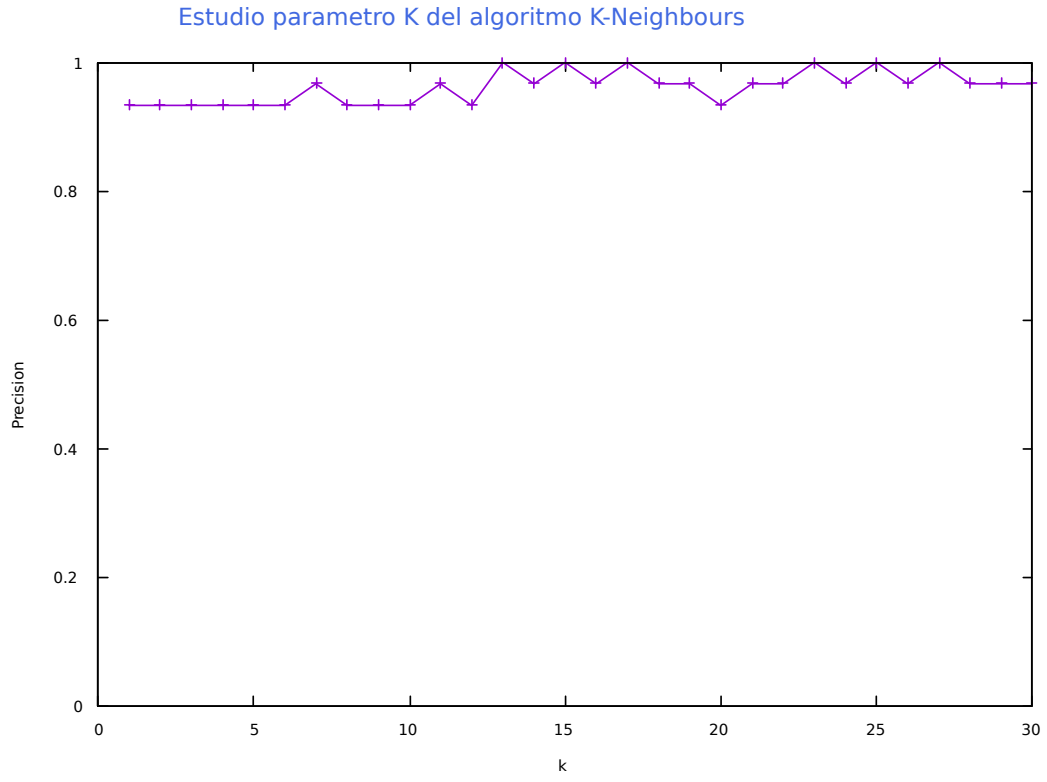
La salida por pantalla es la siguiente:

```
Uso: ./grafica.sh <data.dat> <n_k>
```

Nos indica que al *script* hemos de indicarle como entrada el corpus y el rango de valores para el parámetro k, del algoritmo *kneighbours*, que queremos que nos muestre la gráfica. Si queremos que la gráfica muestre un barrido para k = [1...30]:

```
./grafica.sh iris.dat 30
```

El resultado de la ejecución anterior es la gráfica que se muestra a continuación:



Nota: Para poder ejecutar el script es necesario proporcionarle permisos de ejecución.

```
chmod +x grafica.sh
```

Observamos en la gráfica que con k=13 se obtiene una *precision = 100*% debido a que tenemos 30 aciertos de 30 muestras analizadas sobre 30 muestras totales (test).

El estudio realizado con las muestras de evaluación (*test*) y el barrido de parámetros k = [1...30] utilizando la partición de entrenamiento (*train*) para construir el modelo *kneighbours* nos indica que un buen parámetro k para clasificar nuevas muestras es k=13.