PEC 2: Extracción y selección de atributos

Presentación

El objetivo de esta prueba de evaluación es el estudio de un conjunto de datos de mamografías para el cribaje de cáncer de mama.

Competencias

En este enunciado se trabajan en un determinado grado las siguientes competencias general de máster:

- Capacidad para proyectar, calcular y diseñar productos, procesos e instalaciones en todos los ámbitos de la ingeniería en informática.
- Capacidad para el modelado matemático, cálculo y simulación en centros tecnológicos y de ingeniería de empresa, particularmente en tareas de investigación, desarrollo e innovación en todos los ámbitos relacionados con la ingeniería en informática.
- Capacidad para la aplicación de los conocimientos adquiridos y para solucionar problemas en entornos nuevos o poco conocidos dentro de contextos más amplios y multidisciplinares, siendo capaces de integrar estos conocimientos.
- Poseer habilidades para el aprendizaje continuado, autodirigido y autónomo.
- Capacidad para modelar, diseñar, definir la arquitectura, implantar, gestionar, operar, administrar y mantener aplicaciones, redes, sistemas, servicios y contenidos informáticos.
- Capacidad para asegurar, gestionar, auditar y certificar la calidad de los desarrollos, procesos, sistemas, servicios, aplicaciones y productos informáticos.

Las competencias específicas de esta asignatura que es trabajan son:

- Entender que es el aprendizaje automático en el contexto de la Inteligencia Artificial.
- Distinguir entre los diferentes tipos y métodos de aprendizaje.
- Aplicar las técnicas estudiadas en un caso concreto.

Objetivos

En este PEC se aplicarán a un caso concreto los conceptos del temario sobre extracción y selección de atributos (Tema 3).

Descripción de la PEC a realizar

Datos

El conjunto de datos consta de 516 casos en los que la lesión resultó ser benigna y 445 casos en los que la lesión resultó ser maligna. Cada entrada del registro incluído en el fichero de datos *mamografias.data* contiene los siguientes atributos:

1. Evaluación clínica: del 1 al 5 (1= claramente benigno, 5 = altamente sospechoso de ser maligno).

2. Edad: edad del paciente en años.

3. Forma de la lesión (1 = redondo, 2 = ovalado, 3 = lobular, 4 = irregular).

4. Contorno de la lesión (1 = circunscrito, 2 = microlobulado, 3 = oscurecido, 4 = mal definido, 5 = espiculado).

5. Densidad de la lesión (1 = alta densidad, 2 = iso, 3 = baja densidad, 4 = contiene grasa).

6. Gravedad real de la lesión (0 = benigno, 1 = maligno). Este dato indica la clase real de pertenencia de la lesión.

Un valor de interrogante '?' Corresponde a un valor ausente. El archivo adjunto *mamografias_info.txt* describe la información de dichos atributos. Los archivos pertenecen a la base de datos 'Mammographic Mass Data Set' del Machine Learning Repository de la Universidad de California, Irvine:

http://archive.ics.uci.edu/ml/datasets/Mammographic+Mass

Ejercicio 1

Efectuar, si es necesario, el tratamiento previo de los datos. Es conveniente mostrar los histogramas con la distribucion de valores de cada uno de los atributos. Justifique la necesidad de cada operación en base a las características del problema.

Los datos proporcionados constan de seis columnas, las cinco primeras corresponden a medidas indirectas del tumor de diferente naturaleza (atributos), mientras que la sexta columna es una etiqueta que indica si la lesión es benigna o maligna (esta útlima la utilizaremos como etiqueta para validar los resultados). Por tanto las técnicas de análisis las aplicaremos a las columnas de la 1 a la 5. Todos los atributos son numéricos, y como el rango de valores es diferente conviene estandarizarlos para pode comparar. Los valores ausentes los substituiremos por la moda del atributo (valor más probable). El código listado a continuación describe los pasos a realizar, y finalmente guarda los datos tratados en la variable *atrsN*:

```
from collections import Counter

# declaración de funciones

def valorAbsent (l):
   cl = Counter([l[i] for i in range(len(l)) if (l[i] != '?')])
   moda = float(cl.most_common()[0][0])
   return list(map(lambda x: moda if x=='?' else float(x), l))

# Cargar el archivo:
ll = list(map(lambda l: (l.strip()).split(','),
            open('mamografias.data', 'r').readlines()))

# unzipear la lista para obtener las clases:
llt = list(zip(*ll))
# classes
classes = llt[5]

ind_cl0 = [i for i in range(len(classes)) if classes[i]=='0']
ind_cl1 = [i for i in range(len(classes)) if classes[i]=='1']

# atributos

llt = llt[0:5]

# valores ausentes:
atrs = list(map(valorAbsent, llt))

# normalización:

import numpy

prom = list(map(numpy.average, atrs))
stds = list(map(numpy.std, atrs))

atrsN = [list(map(lambda x: (x - prom[i]) / stds[i], atrs[i]))
        for i in range(len(atrs))]
```

Ejercicio 2

Aplique un análisis PCA a los datos resultantes del ejercicio anterior y estudie los vectores propios y las varianzas resultantes. Debe tomar como modelo el código 3.5 de los materiales. ¿Cuántas componentes necesarios para obtener un 95% de la varianza?.

Para aplicar PCA a los datos anteriores, utilizaremos las funciones de diagonalización de matrices incluidas en la librería numpy. El código siguiente calcula los valores y vectores propios de la matriz de covarianza de los datos y calcula la varianza explicada por cada uno de los componentes:

```
from numpy import *

# Obtener matriz de covarianza:
d = array(atrsN).transpose()
d1 = d - d.mean(0)
matcov = dot(d1.transpose(), d1)
# Valores y vectores propios:
valp1,vecp1 = linalg.eig(matcov)
# Ordenar valores propios:
ind_cre = argsort(valp1)
ind_decre = ind_cre[::-1]
val_decre = valp1[ind_decre]
vec_decre = vecp1[:,ind_decre]
# calcular componentes 95 % varianza:
n = val_decre / val_decre.sum()
```

Tras ejecutar el código, podemos comprobar que hacen falta los 5 componentes para tener un 99.7% de la varianza, puesto que con 4 componentes se obtiene tan solo un 94.1%.

Ejercicio 3

Con los resultados del ejercicio anterior, proyectar con tres componentes y obtiene una representación gráfica en 3D de los datos. Debe tomar como modelo el código 3.5 de los materiales. Al hacer la gráfica, dibuje los puntos de cada una de las clases de un color diferente. ¿Qué conclusiones se pueden sacar?. Realice comprobaciones similares tomando dos componentes.

Para aplicar las proyecciones y representarlas gráficamente utilizaremos las librerías pylab y matplotlib como se indica en el código siguiente:

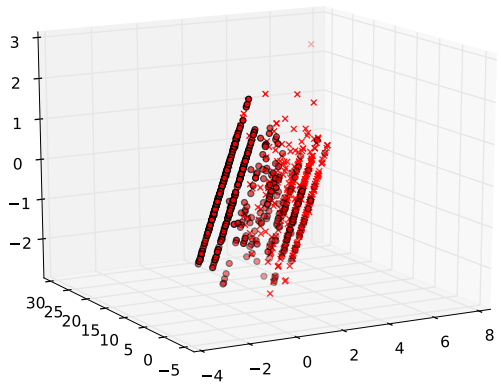
```
import pylab
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plot
set_printoptions(precision = 3)

# Proyección de los datos (espacio PCA 3D)
d_PCA = zeros((d.shape[0],d.shape[1]))
for i in range(d.shape[0]):
    for j in range(d.shape[1]):
        d_PCA[i,j] = dot(d1[i,:],vecp1[:,j])

# representación gráfica de los datos PCA en 3D:
fig2 = plot.figure()
sp2 = fig2.gca(projection = '3d')
sp2.scatter(d_PCA[ind_cl1,0],d_PCA[ind_cl1,1],d_PCA[ind_cl1,2],c='r',marker='x')
sp2.scatter(d_PCA[ind_cl0,0],d_PCA[ind_cl0,1],d_PCA[ind_cl0,2],c='r',marker='o')
plot.show()
```

La figura siguiente representa la proyección de los datos en el espacio PCA 3D:

Figura 1: Proyección datos espacio PCA 3D



Ejercicio 4

Aplique la técnica Multidimensional Scaling (MDS) para obtener una representación gráfica bidimensional de los datos del ejercicio 1. Debe tomar como modelo los códigos del 3.18 al 3.21 de los materiales. Represente gráficamente los puntos de tal forma que se distingan las clases, utilizando métricas euclídea y Pearson. ¿Qué conclusiones se pueden sacar?. ¿Y en comparación a la representación del ejercicio anterior?.

Para aplicar el MDS con diferentes métricas (Pearson y Eucídea) y graficar los resultados utilizaremos el código scaledown los materiales (código 3.18) como se indica en el siguiente código:

```
from scaledown import *
from pearson import *
from euclidean import *
import pylab as py

metrica = pearson
#metrica = euclidean

# Tecnica MDS
mds = scaledown(list(zip(*atrsN)), metrica)
dades = numpy.array(mds)
# Representación de los datos en el espacio 2D MDS:
fig1 = py.figure()
py.scatter(dades[ind_cl0,0],dades[ind_cl0,1],marker='o',c='r')
py.scatter(dades[ind_cl1,0],dades[ind_cl1,1],marker='x',c='b')
py.show()
```

Las figuras siguientes indican la distribución de los puntos una vez proyectados en un espacio 2D utilizando la técnica MDS con métrica Euclídea y Pearson:

Ilustración 1: MDS con distancia Euclídea

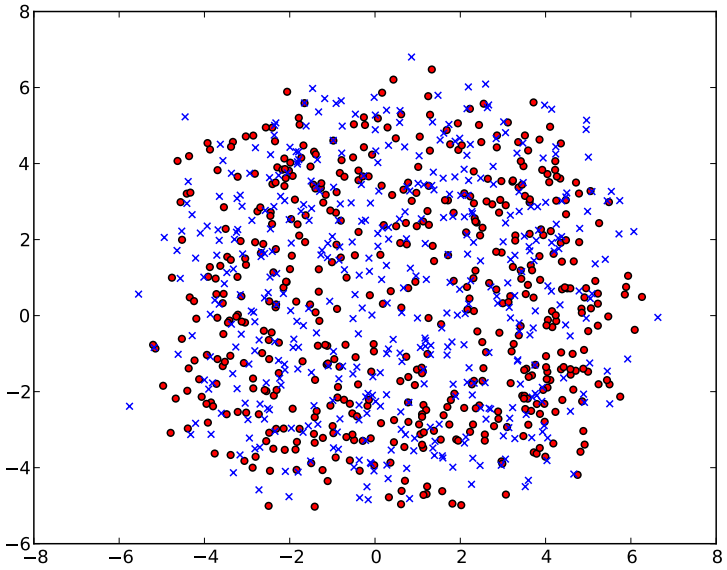
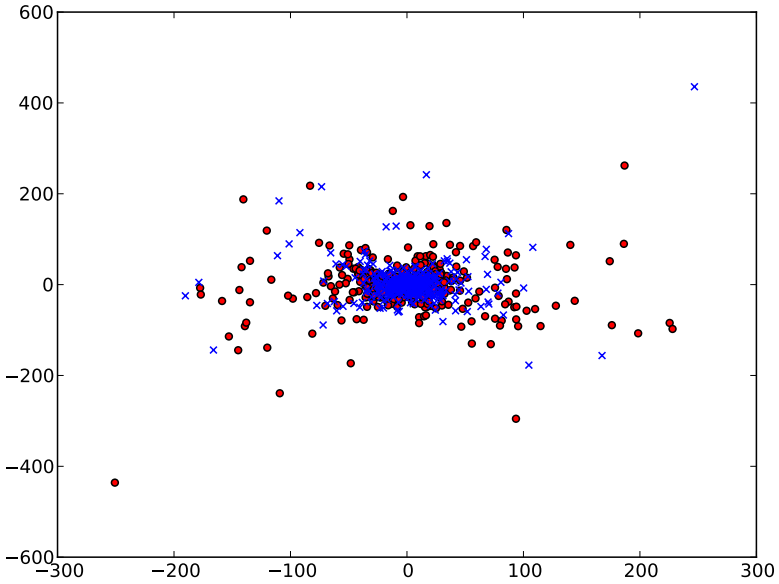


Ilustración 2: MDS con distancia Pearson



Ejercicio 5

Aplique LDA a los datos del ejercicio 1 y las resultantes del ejercicio 2. Debe tomar como modelo el código 3.17 de los materiales. Interprete los resultados obtenidos y compárelos con las tècnicas aplicadas en los ejercicios anteriores.

Para aplicar el LDA utilizaremos la librería sklearn del scikits siguiendo el código 3.17 de los materiales de la asignatura. El código siguiente indica los pasos a seguir:

```
# Aplicación de LDA a los datos del ejercicio 1:
import numpy as np
import pylab
from sklearn.lda import LDA

XT = np.array(atrsN).transpose()
labelT = np.array(classes)

# Fase de entrenamiento:
clf = LDA()
clf.fit(XT, labelT)
LDA(priors=None)

# Predicciones LDA:
prediccions = clf.predict(XT)
# Etiquetas classes de pertenencia:
classes=list(map(lambda x: int(x),classes))

# Fraccion de aciertos:
print float(sum(labelT == prediccions)) / float(len(classes))

# Aplicación a los datos proyectados PCA

# Análisis PCA:
# Matriz Covarianza:
d1 = XT - XT.mean(0)
matcov = dot(d1.transpose(), d1)
valp1,vecp1 = linalg.eig(matcov)

# Ordenar valores propios:
ind_cre = argsort(valp1)
ind_decre = ind_cre[::-1]
val_decre = valp1[ind_decre]
vec_decre = vecp1[:,ind_decre]

# proyeccion PCA:
XT_PCA = numpy.array([[dot(d1[i,:],vecp1[:,j])
                       for i in range(XT.shape[0])]
                      for j in range(XT.shape[1]-1)]).transpose()

# Entrenamiento de LDA con datos PCA:
clf = LDA()
clf.fit(XT_PCA, labelT)
LDA(priors=None)
# predicciones sobre el conjunto:
prediccions_PCA = clf.predict(XT_PCA)
# Fraccion de aciertos datos PCA:
print float(sum(labelT == prediccions_PCA)) / float(len(labelT))
```

Tras ejecutar el código comprobamos que la fracción de aciertos cuando se utilizan los datos estandarizados es de 80.02%, mientras que el resultado en el caso de aplicar LDA a los datos PCA es ligeramente mejor 80.74%.

Recursos

Este PEC requiere de los siguientes recursos:

Básicos: Ficheros de datos adjuntos al enunciado.

Complementarios: Manual de teoría de la asignatura, y listados de código del capítulo 3 (Extracción y Selección de atributos).

Criterios de valoración

Los ejercicios tendrán la siguiente valoración asociada:

Ejercicio 1: 1 punto

Ejercicio 2: 2 puntos

Ejercicio 3: 2 puntos

Ejercicio 4: 2.5 puntos

Ejercicio 5: 2.5 puntos

Es necesario razonar las respuestas en todos los ejercicios. Las respuestas sin justificación no recibirán puntuación.

Formato y fecha de entrega

La PEC debe entregarse antes del **próximo 25 de Abril** (antes de las 24h).

La solución a entregar consiste en un informe en formato PDF (formato libre) más los archivos de código (*. Py) que usó para resolver la prueba. Estos archivos deben comprimir en un archivo ZIP.

Adjuntar el fichero a un mensaje en el apartado de **Entrega y Registro de AC (RAC)**. El nombre del archivo debe ser ApellidosNombre_IA_PEC2 con la extensión. zip.

Para dudas y aclaraciones sobre el enunciado, diríjase al consultor responsable de su aula.

Nota: **Propiedad intelectual**

A menudo es inevitable, al producir una obra multimedia, hacer uso de recursos creados por terceras personas. Es por tanto comprensible hacerlo en el marco de una práctica de los estudios del Máster de Informática, siempre que esto se documente claramente y no suponga plagio en la práctica.

Por lo tanto, al presentar una práctica que haga uso de recursos ajenos, se presentará junto con ella un documento en el que se detallen todos ellos, especificando el nombre de cada recurso, su autor, el lugar donde se obtuvo y el su estatus legal: si la obra está protegida por copyright o se acoge a alguna otra licencia de uso (Creative Commons, licencia GNU, GPL ...). El estudiante deberá asegurarse de que la licencia que sea no impide específicamente su uso en el marco de la práctica. En caso de no encontrar la información correspondiente deberá asumir que la obra está protegida por copyright.

Deberán, además, adjuntar los archivos originales cuando las obras utilizadas sean digitales, y su código fuente si corresponde.