Práctica 2: Selección de características con sklearn

La selección de características peude usarse para reducir el volumen de un conjunto de datos y por tanto aumentar la velocidad con la que los datos pueden procesarse, o bien para mejorar la precisión de un algoritmo de aprendizaje, eliminando las variables que causen errores en el modelo.

Se emplearán los conjuntos de datos siguientes:

IRIS: Los datos están en el fichero siguiente: `datasets-uci-iris.csv`

LETTER: Los datos están en el fichero siguiente: `datasets-uci-letter.csv`

SINTETICO: El dataset se genera artificialmente con la siguiente llamada:

```
from sklearn.datasets import make_classification
X, y = make_classification(n_samples=1000,
                           n_features=10,
                           n_informative=3,
                           n_redundant=0,
                           n_repeated=0,
                           n_classes=2,
                           random_state=0,
                           shuffle=False)
```

Los datasets que estén contenidos en ficheros `.csv` se cargarán en memoria en un dataframe *pandas*, de forma análoga al siguiente ejemplo (*nota:* si el fichero csv incluye cabecera, no son necesarios los argumentos "header" ni "names"):

```
import pandas as pd
iris_filename = 'datasets-uci-iris.csv'
iris = pd.read_csv(iris_filename, sep=',', decimal='.',
header=None,names= ['sepal_length', 'sepal_width', 'petal_length',
'petal_width','target'])
```

Parte 1. Eliminación de variables con poca varianza

Las variables cuyo valor sea constante o casi constante no aportan información al modelo y por tanto pueden ser eliminadas. Calcula la varianza de cada una de las variables para todos los datasets mencionados con la orden `np.var()` y anótala en el documento de prácticas.

Elimina de cada dataset las variables cuya varianza sea menor que el 10% de la varianza de la variable más dispersa, eligiendo el valor de XX en un script como el que sigue. Anota en el documento el código que hayas escrito para realizar esta tarea.

```
from sklearn.feature_selection import VarianceThreshold
sel = VarianceThreshold(XX)
# .iloc sirve para acceder al dataframe con indices numéricos
iris_reducido = sel.fit_transform(iris.iloc[:,0:4])
```

Parte 2. Eliminación de variables basada en estadísticos univariantes

Hay diferentes estadísticos que miden la calidad de una variable. Scikit-learn tiene implementados, entre otros, los siguientes métodos:

- SelectKBest: elimina todas las variables menos las k mejor valoradas
- SelectPercentile: elimina todas las variables menos el porcentaje indicado

Estos métodos hacen uso de una de las siguientes funciones para determinar si una variable es relevante:

- Para regresión: `f_regression`
- Para clasificación: `chi2` o `f_classif`

El siguiente script realiza una selección de características sobre el problema Iris mediante el método SelectKBest con la función chi2. Aplica este método y SelectPercentile a todos los datasets de la práctica y anota las diferencias entre los mismos. Copia en el documento el código que hayas utilizado.

```
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
sel = SelectKBest(chi2,k=2)
iris_reducido1 = sel.fit_transform(iris.iloc[:,0:4],iris.iloc[:,4])
```

Parte 3. Eliminación recursiva de variables

La eliminación recursiva de variables consiste en entrenar un modelo con todas las variables y determinar la importancia de cada una en el resultado. Se eliminan las variables con importancia menor y se repite el proceso. Mediante validación cruzada, se valora cada uno de los modelos que se vayan obteniendo y se determina el punto óptimo como aquél en el que el error del modelo sea menor.

Es necesario que el clasificador/modelo de regresión utilizado proporcione una medida de la importancia de cada variable para poder emplear este método

```
from sklearn.feature_selection import RFECV
from sklearn.svm import SVC
estimator = SVC(kernel="linear")
sel = RFECV(estimator, step=1, cv=5)
iris_reducido2 = sel.fit(iris.iloc[:,0:4],iris.iloc[:,4])
print sel.ranking_
print sel.support_
```

Parte 4. Eliminación de variables usando SelectFromModel

Los árboles de decisión no necesariamente emplean todas las variables. La fracción de veces que una variable es elegida tras lanzar repetidamente un árbol de decisión con semilla aleatoria es una medida de la importancia de la variable. Por medio del siguiente script, ordena la importancia de las variables en todos los problemas de esta práctica. Anota los resultados y las gráficas obtenidas.

```
import numpy as np
import matplotlib.pyplot as plt

from sklearn.ensemble import ExtraTreesClassifier


# Clasificador basado en arboles de decision
forest = ExtraTreesClassifier(n_estimators=250,
                              random_state=0)

# X, y son las variables de entrada y de salida del dataset
forest.fit(X, y)
importances = forest.feature_importances_
std = np.std([tree.feature_importances_ for tree in
forest.estimators_],
             axis=0)
indices = np.argsort(importances)[::-1]

# Variables ordenadas por importancia
print("Variables ordenadas:")

for f in range(X.shape[1]):
    print("%d. variable %d (%f)" % (f + 1, indices[f],
importances[indices[f]]))

# Grafico con las importancias de las variables
plt.figure()
plt.title("Importancia de las variables")
plt.bar(range(X.shape[1]), importances[indices],
       color="r", yerr=std[indices], align="center")
plt.xticks(range(X.shape[1]), indices)
plt.xlim([-1, X.shape[1]])
plt.show()
```