Exercicio 2: SVM

Diego Fernandez Merjildo

October 5, 2016

1 Descrição do exercicio

- Leia os dados do arquivo data1.csv A classe de cada dado é o valor da última coluna (0 ou 1).
- Treine um SVM com kernel RBF nos dados do arquivos.
- A validação externa deve ser 5-fold estratificado. Para cada conjunto de treino da validação externa faça um 3-fold para escolher os melhores hiperparametros para C (cost) e gamma.
- Faça um grid search de para o C nos valores 2**-5, 2**-2, 2**0, 2**2, e 2**5 e gamma nos valores 2**-15, 2**-10, 2**-5, 2**0, e 2**5

1. Qual a accuracia media (na validação de fora).
2. Quais os valores de C e gamma a serem usados no classificador final (fazer o 3-fold no conjunto todo).

2 Resultados

Os resultados foram obtidos rodando o script apresentado na seção 3. No script usamos um k-fold estratificado de 5 externo e um outro k-fold estratificado de 3 interno.
Rodando o script em python obtemos os seguintes valores:

```
1
2 Valor Hiperparametros (C=32, Gamma=0.0009765625)
3 Valor Hiperparametros (C=4, Gamma=0.03125)
4 Valor Hiperparametros (C=1, Gamma=0.03125)
5 Valor Hiperparametros (C=1, Gamma=0.03125)
6 Valor Hiperparametros (C=4, Gamma=0.03125)
7
8 Acuracia media:0.907474804031
9 Valor final hiperparametros (C=4, Gamma=0.03125)
```

3 Codigo fonte em python

Listing 1: Codigo em Python

```
#!/usr/bin/python

import sys,os,csv
import pandas
import numpy as np
from sklearn import cross_validation
from sklearn.cross_validation import StratifiedKFold
from sklearn import svm as SVM

datFileName="data1.csv"
dirPath=os.path.dirname(os.path.realpath(__file__))
classList=[]
data=[]

## Load CSV
def loadCsvData(fileName):
    raw_data = open(fileName, 'rb')
    rawData = pandas.read_csv(raw_data, delimiter=",", skiprows=1)
    return rawData.values

def getData(rawData):
    #print "\n---- Getting data from File ----"
    lineNum = rawData.shape[0]
    colNum = rawData.shape[1]
    data = np.array(rawData[0:lineNum, 0:colNum-1])
    for i in range(lineNum):
        classList.append(rawData[i][colNum - 1])
    return [data, np.array(classList) ]

# In order to get hyperparameter
def internFolds(data_train, data_test, labelsTrain, labelsTest):
    acxmax = 0
    c_max=0
    gamma_max=0
    for c in [2**-5, 2**-2, 1, 2**2, 2**5]:
        for gamm in [2**-15, 2**-10, 2**-5, 1, 2**5]:
            svm = SVM.SVC(C = c, gamma = gamm)
            svm.fit(data_train, labelsTrain)
            accuracy = svm.score(data_test, labelsTest)
            if accuracy > acxmax:
                acxmax = accuracy
                c_max = c
                gamma_max = gamm
```

```
44      return [acxmax, c_max, gamma_max]
45
46  def main(argv=None):
47      if argv is None:
48          arv = sys.argv
49      rawdata = loadCsvData(dirPath + "/" + datFileName)
50      [data, labels] = getData(rawdata)
51      final_accuracy = 0
52
53      skf = cross_validation.StratifiedKFold(labels, n_folds=5)
54      for train_index, test_index in skf:
55          new_data_train = data[train_index]
56          new_data_test = data[test_index]
57          new_labels_train = labels[train_index]
58          new_labels_test = labels[test_index]
59
60          acx = 0
61          skf_intern = cross_validation.StratifiedKFold(↩
                new_labels_train, n_folds=3)
62          for intern_train_index, intern_test_index in skf_intern:
63              intern_data_train = new_data_train[intern_train_index]
64              intern_data_test = new_data_train[intern_test_index]
65              intern_labels_train = new_labels_train[↩
                    intern_train_index]
66              intern_labels_test = new_labels_train[intern_test_index↩
                    ]
67              [accuracy, c, gamma] = internFolds(intern_data_train, ↩
                    intern_data_test, intern_labels_train, ↩
                    intern_labels_test)
68              if accuracy > acx:
69                  acx = accuracy
70                  c_final = c
71                  gamma_final = gamma
72          #print("acx", acx)
73          print("Valor Hiperparametros (C=%s, Gamma=%s)" % (c_final, ↩
                gamma_final) )
74          svm_model = SVM.SVC(C = c_final, gamma = gamma_final)
75          svm_model.fit(new_data_train, new_labels_train)
76          acc_5_fold = svm_model.score(new_data_test, new_labels_test↩
                )
77          final_accuracy = final_accuracy + acc_5_fold
78
79      final_accuracy = final_accuracy / 5
80      print("Acuracia media:%s" % final_accuracy)
81      print("Valor final hiperparametros (C=%s, Gamma=%s)" % (c_final↩
            , gamma_final) )
82
83  if __name__ == "__main__":
```

```
84      sys.exit(main())
```