Relatório do Segundo Exercício

Délio Gomes Soares - 188947

05/10/2016

1 Atividade Proposta

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 dos 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 hiperparâmetros 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 acurácia média (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).

NÃO use funções prontas que já fazer o *grid search*como *GridSearchCV* do *sklearn* ou o *tune* do pacote e1070 do R. Neste exercício eu quero que vocês façam os *loops* explicitamente.

2 Solução

2.1 Respostas

1. A acurácia média da validação de fora foi de 91,19270%.

2. Os valores de C e *gamma* para o classificador final foram de 4,0 para o C e 0,03125 para o *gamma*.

2.2 Saída do código

A saída produzida pelo código em python com a acurácia média e os valores de C e *gamma* para o classificador final.

```
-------------------------------------------------------
                Questao 1 - Acuracia Media

A Acuracia media final eh: 0.911927025009
-------------------------------------------------------
        Questao 2 - C e gamma do classificado final

Valor de c final:        4.0
Valor de gamma final:    0.03125
-------------------------------------------------------
```

2.3 Código fonte em *python*

```
# -*- coding: utf-8 -*-
"""
Exercicio 2
Autor: Delio Gomes Soares
RA: 188947
Data: 05/10/2016
"""

import csv
import numpy as np
from sklearn.svm import SVC
from sklearn.cross_validation import StratifiedKFold

#--------------------------------------------------------------#
def fileRead(file): # Leitura do conjunto de dados
    with open(file, 'r') as f:
```

```
        try:
            reader = csv.reader(f)
            data =  list(reader)
        finally:
            f.close()
    return data;
#-------------------------------------------------------------------#
Cost = np.array([2**-5, 2**-2, 2**0, 2**2, 2**5])
Gamma = np.array([2**-15, 2**-10, 2**-5, 2**0, 2**5])

def svm(X, y):
    Hmax = np.array([float,float])
    melhorAcuracia = 0;
    acInterna = 0
    contItera = 0
    for c in Cost:
        for g in Gamma:
            skf3 = StratifiedKFold(y, n_folds=3)
            for train_indexInt, test_indexInt in skf3:
                X_TrInter = X[train_indexInt]
                y_TrInter = y[train_indexInt]
                X_TeInter  = X[test_indexInt]
                y_TeInter  = y[test_indexInt]
                clf = SVC(kernel='rbf', C = c, gamma = g)
                clf.fit(X_TrInter, y_TrInter)
                result = clf.score(X_TeInter, y_TeInter)
                acInterna += result
                contItera += 1
            mediaACInterno = acInterna/contItera
            contItera = 0
            acInterna = 0
            if(melhorAcuracia < mediaACInterno):
                melhorAcuracia = mediaACInterno
                Hmax[0] = c
                Hmax[1] = g
    return Hmax
#-------------------------------------------------------------------#

entrada = fileRead('data1.csv');
dados = entrada[1:]
classe = []
for row in dados:
    classe.append(row[-1])
    del row[-1]

```

```
dados = np.asarray(dados)
dados = dados.astype(float)
classe = np.array(classe);
classe = classe.astype(float)

contItera = 0
acFinal = 0
cMax = 0
gammaMax = 0
skf5 = StratifiedKFold(classe, n_folds=5)
Hmax = []

for train_index, test_index in skf5:

    X_Treino = dados[train_index]
    y_Treino = classe[train_index]

    Hmax = svm(X_Treino, y_Treino)

    cMax = Hmax[0]
    gammaMax = Hmax[1]

    X_Teste  = dados[test_index]
    y_Teste  = classe[test_index]

    modelo = SVC(kernel='rbf', C = cMax, gamma = gammaMax)
    modelo.fit(X_Treino, y_Treino)
    result = modelo.score(X_Teste, y_Teste)

    acFinal += result
    contItera += 1

mediaACFinal = acFinal/contItera

print("-------------------------------------------------------------------")
print("\t\tQuestao 1 - Acuracia Media\n")
print("A Acuracia media final eh:", mediaACFinal)

Hmax = svm(dados, classe)

print("-------------------------------------------------------------------")
print("\tQuestao 2 - C e gamma do classificado final \n")
print("Valor de c final:\t", Hmax[0])
print("Valor de gamma final:\t", Hmax[1])
print("-------------------------------------------------------------------")
```