MO444 – Aprendizado de Máquina

Edgar Rodolfo Quispe Condori - RA 192617

June 8, 2017

Questão 1. *Há varias possivies definições de um outlier, entre elas:*

- *dados cujo vizinho(s) mais próximo(s) esta a uma distancia muito maior que a distancia dos vizinhos mais proximos dos dados normais (mas cuidado, o vizinho(s) mais proximo(s) de um outlier pode ser outro outlier!!)*
- *dados que estao fora do hiperplano onde os dados normais se encontram*
- *dados que se incluidos em clusters com dados normais tornam estes clusters muito menos compactos.*
- *dados que estao numa região de baixa densidade de pontos/dados*

Os dados em data8.csv contem 900 dados normais e 7 outliers.
descubra os outliers e descreva o que voce fez

Para descobrir os outliers nos consideramos que ele estão a uma distancía longa do resto de dados, então foi usado o algoritmo de clusterizacão de Nearest Neightbours com $n\_neigbours = 9$, isto permite recuperar ate o 8vo vizinho mais perto. Plotando as distâncias dá para ver que existem un conjunto de pontos que têm seu 8vo vizinho mais perto muito longe (> 1). Considera-se o 8to vizinho mais perto para evitar que se os outlier estão pertos entre eles se tenha falsos positivos.

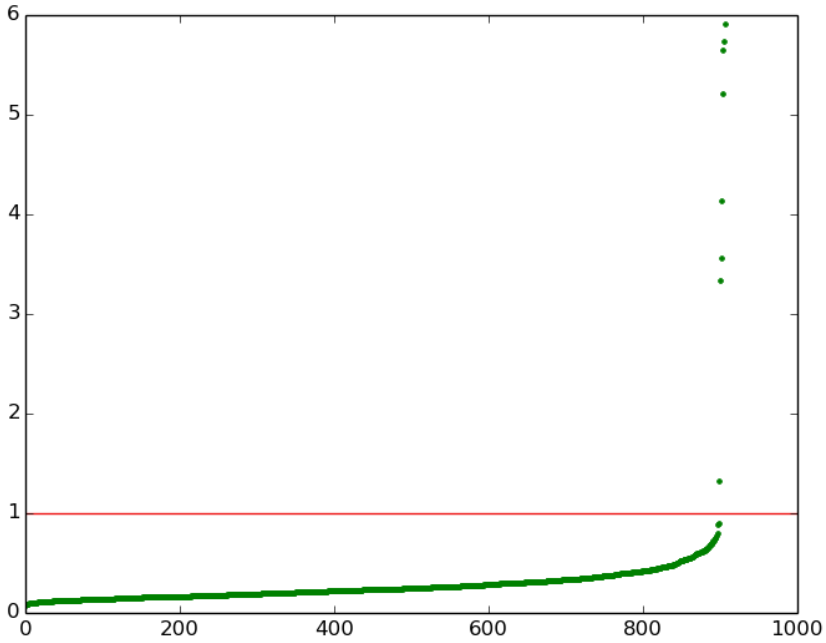

Figure 1: Distância dos 8vo vizinho mais perto

Indice do Dado	Ind. do 8vo vizinho mais perto	Dist. ao 8vo vizinho mais perto
210	**749**	**3.54911264966**
318	**756**	**5.21026558056**
464	**523**	**5.61933873423**
682	**384**	**3.32305029453**
724	**264**	**4.1122445258**
799	**749**	**5.7171244564**
821	747	1.29908044401
827	**264**	**5.91413684657**

Table 1: Dados com 8vo vizinho mais perto a uma distância maior que 1.

Esses pontos são mostrados na tabela 1, mas baseados no enunciado do problema considera-se os 8 mais longos como outliers (eles estão em negrita).

Esses resultados podem-se reafirmar ao usar o *Isolation Forest* que com os hiperparametros $'n\_estimators' : 300, 'contamination' : 0.0077177$. Ele retorna os dados com indices **210, 318, 464, 724, 799, 827** como ouliers.

O codigo:

```
import numpy as np
import pandas as pd
from sklearn.neighbors import NearestNeighbors
from sklearn.ensemble import IsolationForest
import pylab as plt

def read_data():
  #read cvs file
  data = pd.read_csv('data8.csv')
  data = np.array(data).astype(np.float)
  return data

def test_algorithm(data, alg_name, params):
  print 'Testing', alg_name, 'with', params
  alg = {'svm': OneClassSVM, 'isolation_forest': IsolationForest}
  clf = alg[alg_name](**params).fit(data)
  pred = clf.predict(data)
  for i in range(len(pred)):
    if(pred[i] == -1):
      print "Outlier:", i

def evaluate_IF(data):
  for n_estimators in [100, 300, 500, 700]:
    params = {'n_estimators' : n_estimators, 'contamination': 0.0077177}
    test_algorithm(data, 'isolation_forest', params)

def plotDistances(distances, save_file_name = ""):
  distances.sort()
  plt.figure()
  plt.plot(range(len(distances)), distances, 'g.')
  plt.axhline (y=1 , color='#EE1212' )
  plt.savefig(save_file_name)

def nneighbors(data):
  nn = NearestNeighbors(n_neighbors = 9).fit(data)
  distances, indices = nn.kneighbors(data)

  for i in range(distances.shape[0]):
    distance = distances[i]
    indice = indices[i]
    if distance[8] >= 1:
      print "OUTLIER", i, indice[8], distance[8]

  plotDistances(distances[:, 8].ravel(), "dis.png")

if __name__ == '__main__':
    data = read_data()
```

```
nneighbors(data)
evaluate_IF(data)
```