Exercicio 7: Anomalias

Diego Fernandez Merjildo

November 23, 2016

1 Descrição do exercicio

Este exercicio é sobre anomalias e series temporais, e é aberto. Eu vou propor um problema para voces mas eu nao sei a solucao, eu gostaria que voces explorassem suas ideias e intuicoes.

Nao há resposta certa ou errada, so ha alternativas que dao e nao dao mais ou menos certo e nao dar certo NAO é um problema - é a realidade de uma pesquisa/exploracao.

Queremos detectar regioes numa serie temporal que sao anomalas, obviamente sem uma definicao do que é anomalo!. As series estao abaixo.

2 Resultados

```
closest point
(array([[  2702.08593075,   3001.97673369,   3301.76147744, ...,
           6300.93789736,   6600.91165352,   6900.86520547],
        [  3001.4715636 ,   3301.41787772,   3601.27074803, ...,
           6600.70596367,   6900.68919575,   7200.65467613],
        [  3301.53476919,   3601.48543107,   3901.34256298, ...,
           6900.77196451,   7200.75406427,   7500.71794847],
        ...,
        [ 33900.26594343,  34200.27460177,  34500.26791599, ...,
          37500.24966641,  37800.25128245,  38100.24775216],
        [ 34200.29987319,  34500.30887906,  34800.30167643, ...,
          37800.28109929,  38100.2826874 ,  38400.27883934],
        [ 34500.2229436 ,  34800.23100718,  35100.22512551, ...,
          38100.21029714,  38400.21193117,  38700.20887445]]), array↩
             ([[2979, 2978, 2977, ..., 2967, 2966, 2965],
        [2979, 2978, 2977, ..., 2967, 2966, 2965],
        [2979, 2978, 2977, ..., 2967, 2966, 2965],
        ...,
        [2979, 2978, 2977, ..., 2967, 2966, 2965],
        [2979, 2978, 2977, ..., 2967, 2966, 2965],
        [2979, 2978, 2977, ..., 2967, 2966, 2965]]))
(array([[  35100.0076569 ,   35400.00951511,   35700.00866194, ...,
```

```
           38700.0085628 ,   39000.00916313,   39300.00880329],
       [  35400.00955769,   35700.01160387,   36000.01065505, ...,
          39000.01046684,   39300.01111933,   39600.0107167 ],
       [  35700.00887613,   36000.01084639,   36300.00993676, ...,
          39300.00978615,   39600.01041787,   39900.01003258],
       ...,
       [ 315000.00000215,  315300.0000041 ,  315600.00000043, ...,
         318600.00000306,  318900.00000894,  319200.000006  ],
       [ 315300.00001915,  315600.00000079,  315900.00000508, ...,
         318900.00000131,  319200.00000001,  319500.0000002 ],
       [ 315600.00000003,  315900.00001335,  316200.00000523, ...,
         319200.00001136,  319500.00002124,  319800.00001656]]), ↩
            array([[2979, 2978, 2977, ..., 2967, 2966, 2965],
       [2979, 2978, 2977, ..., 2967, 2966, 2965],
       [2979, 2978, 2977, ..., 2967, 2966, 2965],
       ...,
       [2979, 2978, 2977, ..., 2967, 2966, 2965],
       [2979, 2978, 2977, ..., 2967, 2966, 2965],
       [2979, 2978, 2977, ..., 2967, 2966, 2965]]))
Radius
distance outliers: []
distance test: []
sd: 32.43247255
mean: 44.4942541557
```

3 Codigo fonte em python

Listing 1: Codigo em Python

```
1
2  #!/usr/bin/python
3
4  import os
5  import sys
6  import pandas
7  import time
8  import numpy as np
9  import matplotlib.pyplot as plt
10
11 from sklearn.mixture import GaussianMixture
12 from sklearn.neighbors import KernelDensity
13 from sklearn.ensemble import IsolationForest
14 from sklearn.preprocessing import StandardScaler
15 from sklearn.preprocessing import scale
16 from sklearn.neighbors import NearestNeighbors
```

```
17
18 pattern = '%Y-%m-%d %H:%M:%S'
19
20 def load_data(file_name):
21     raw_data = open(file_name, 'rb')
22     rawData = pandas.read_csv(raw_data, delimiter=",", header=None)
23     data = np.array(rawData)[1:]
24     for line in data:
25         line[0] = int(time.mktime(time.strptime(line[0], pattern)))
26         data[:,1] = [float(i) for i in data[:,1]]
27     return data
28
29 def main():
30     dir_path = os.path.dirname(os.path.realpath(__file__))
31     serie1 = load_data( dir_path + "/serie1.csv")
32     serie2 = load_data( dir_path + "/serie2.csv")
33     serie3 = load_data( dir_path + "/serie3.csv")
34     serie4 = load_data( dir_path + "/serie4.csv")
35
36     print "len 1:", len(serie1)
37     print "shape:", serie1.shape
38     print serie1[:,0]
39     print serie1[:,1]
40
41     data = serie1
42     data = scale(data[:,1], with_std=True)
43
44     n_samples, n_features = np.shape(data)
45     n_samples_train = n_samples // 2
46     n_samples_test = n_samples - n_samples_train
47
48     x_train = data[:2980, :]
49     x_outlier = data[2988:3095, :]
50     x_test = data[3096:n_samples, :]
51
52     print "closest point "
53     neigh = NearestNeighbors(n_neighbors=15)
54     neigh.fit(x_train)
55
56     print(neigh.kneighbors(x_outlier))
57
58     print(neigh.kneighbors(x_test))
59
60     print "Radius"
61     neigh = NearestNeighbors(radius=1.6)
62     neigh.fit(x_train)
63
64     out_rng = neigh.radius_neighbors(x_outlier)
```

```
65      print "distance outliers:", np.asarray(out_rng[0][0])
66
67      test_rng = neigh.radius_neighbors(x_test)
68      print "distance test:", np.asarray(test_rng[0][0])
69
70
71      #X = scale(data[:,1], with_std=True)
72      X = data[:,1]
73      sd = np.std(X, ddof=1)
74      print "sd:", sd
75
76      print "mean:", np.mean(X)
77
78  #      gmm = GaussianMixture(n_components=1).fit(serie1[:,1])
79      # gmm_log_dens = gmm.score_samples(serie1[:,0])
80
81      # kde = KernelDensity(kernel='gaussian', bandwidth=0.75).fit(↩
            serie1)
82      #kde_log_dens = kde.score_samples(x_plot)
83
84      # plt.figure(figsize=(25, 17))
85      # plt.fill(x_plot[:, 0], np.exp(gmm_log_dens), fc='#ffaf00', ↩
            alpha=0.7)
86
87  if __name__ == "__main__":
88      sys.exit(main())
```