Министерство образования Республики Беларусь

Учреждение образования

БелорусскиЙ государственный университет

информатики и радиоэлектроники

Факультет компьютерных систем и сетей

Кафедра информатики

|  |
| --- |
|  |

**ЛабоРАТОРНАЯ РАБОТА №8**

«Выявление аномалий»

|  |  |  |
| --- | --- | --- |
| Студент |  | К. Д. Зюсько |
| Преподаватель |  | М. В. Стержанов |

Минск 2019

ХОД РАБОТЫ

**Задание.**

Набор данных ex8data1.mat представляет собой файл формата \*.mat (т.е. сохраненного из Matlab). Набор содержит две переменные X1 и X2 - задержка в мс и пропускная способность в мб/c серверов. Среди серверов необходимо выделить те, характеристики которых аномальные. Набор разделен на обучающую выборку (X), которая не содержит меток классов, а также валидационную (Xval, yval), на которой необходимо оценить качество алгоритма выявления аномалий. В метках классов 0 обозначает отсутствие аномалии, а 1, соответственно, ее наличие.

Набор данных ex8data2.mat представляет собой файл формата \*.mat (т.е. сохраненного из Matlab). Набор содержит 11-мерную переменную X - координаты точек, среди которых необходимо выделить аномальные. Набор разделен на обучающую выборку (X), которая не содержит меток классов, а также валидационную (Xval, yval), на которой необходимо оценить качество алгоритма выявления аномалий.

1. Загрузите данные ex8data1.mat из файла.
2. Постройте график загруженных данных в виде диаграммы рассеяния.
3. Представьте данные в виде двух независимых нормально распределенных случайных величин.
4. Оцените параметры распределений случайных величин.
5. Постройте график плотности распределения получившейся случайной величины в виде изолиний, совместив его с графиком из пункта 2.
6. Подберите значение порога для обнаружения аномалий на основе валидационной выборки. В качестве метрики используйте F1-меру.
7. Выделите аномальные наблюдения на графике из пункта 5 с учетом выбранного порогового значения.
8. Загрузите данные ex8data2.mat из файла.
9. Представьте данные в виде 11-мерной нормально распределенной случайной величины.
10. Оцените параметры распределения случайной величины.
11. Подберите значение порога для обнаружения аномалий на основе валидационной выборки. В качестве метрики используйте F1-меру.
12. Выделите аномальные наблюдения в обучающей выборке. Сколько их было обнаружено? Какой был подобран порог?

**Результат выполнения:**

1. Код выгрузки данных из файла представлен ниже:

file\_path = os.path.join(os.path.dirname(\_\_file\_\_), 'data', 'ex8data1.mat')

dataset = sio.loadmat(file\_path)

X = dataset["X"]

Xval = dataset["Xval"]

yval = dataset["yval"]

1. График приведён ниже:

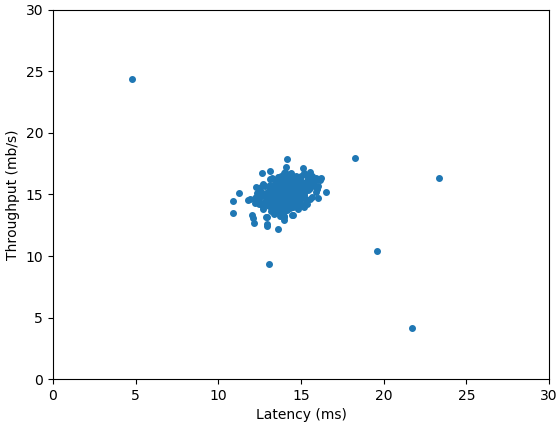


Рисунок 1 – визуализация исходных данных

3. Код реализации:

def estimate\_gaussian(X):

# Useful variables

m, n = X.shape

mu = np.mean(X, axis=0)

# For Sigma2,np.sum requires an axis else it flattens the array and takes the sum which is wrong.

sigma2 = (1/m)\*(np.sum((X-mu)\*\*2, axis=0))

return mu, sigma2

mu, sigma2 = estimate\_gaussian(X)

4. Код реализации:

p = multivariate\_normal(mu, np.diag(sigma2))

print(mu, sigma2)

Результат выполнения:

(array([14.11222578, 14.99771051]), array([1.83263141, 1.70974533]))

5. График приведён ниже:

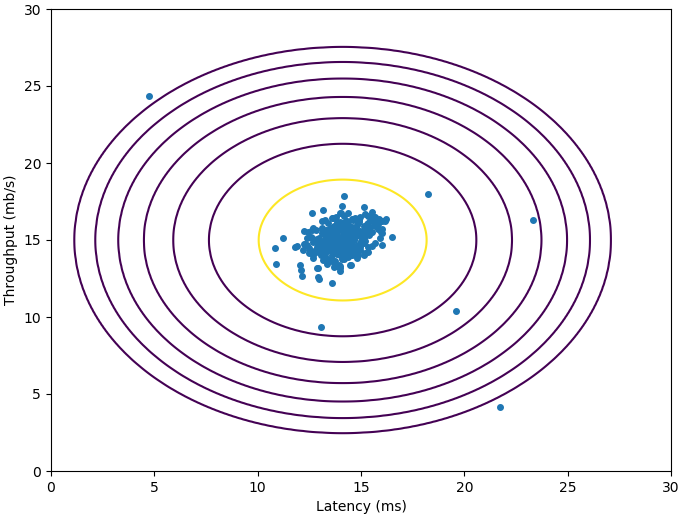


Рисунок 2 – собственные векторы матрицы ковариации

6. Код реализации:

def select\_threshold(y\_val, p\_val):

best\_epsilon, best\_F1 = 0, 0

step\_size = (max(p\_val) - min(p\_val)) / 1000

for epsilon in np.arange(p\_val.min(), p\_val.max(), step\_size):

predictions = (p\_val < epsilon)[:, np.newaxis]

tp = np.sum(predictions[y\_val == 1] == 1)

fp = np.sum(predictions[y\_val == 0] == 1)

fn = np.sum(predictions[y\_val == 1] == 0)

prec = tp / (tp + fp)

rec = tp / (tp + fn)

F1 = 2 \* prec \* rec / (prec + rec)

if F1 > best\_F1:

best\_epsilon = epsilon

best\_F1 = F1

return best\_epsilon, best\_F1

p\_val = p.pdf(Xval)

epsilon, F1 = select\_threshold(yval, p\_val)

print("Best epsilon found using cross-validation:", epsilon)

print("Best F1 on Cross Validation Set:", F1)

Результат выполнения:

('Best epsilon found using cross-validation:', 8.990852779269493e-05)

('Best F1 on Cross Validation Set:', 0.8750000000000001)

7. График приведен ниже:

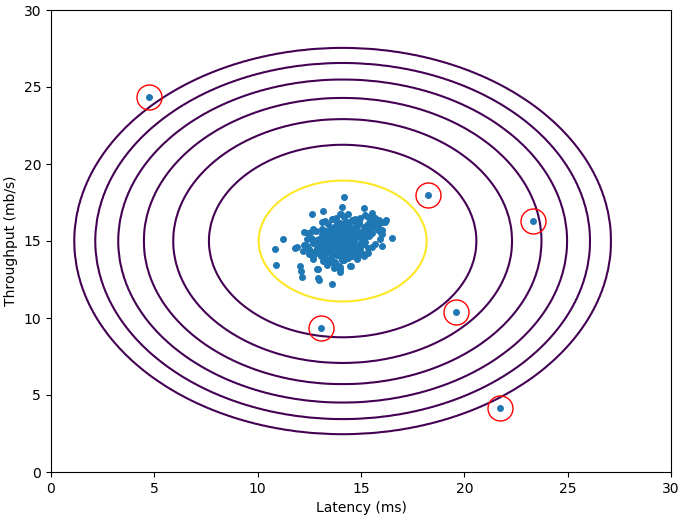


Рисунок 3 – аномальные значения, с учетом выбранного порового значения

8. Код выгрузки данных из файла представлен ниже:

file\_path = os.path.join(os.path.dirname(\_\_file\_\_), 'data', 'ex8data2.mat')

dataset = sio.loadmat(file\_path)

X, Xval, yval = dataset['X'], dataset['Xval'], dataset['yval'][:, 0]

9. Код реализации:

mu, sigma2 = estimate\_gaussian(X)

10. Код реализации:

p = multivariate\_normal(mu, np.diag(sigma2))

11. Код реализации:

p\_val = p.pdf(Xval)

epsilon, F1 = select\_threshold(yval, p\_val)

print('Best epsilon found using cross-validation: %.2e' % epsilon)

print('Best F1 on Cross Validation Set : %f\n' % F1)

print(' (you should see a value epsilon of about 1.38e-18)')

print(' (you should see a Best F1 value of 0.615385)')

Результат выполнения:

Best epsilon found using cross-validation: 1.38e-18

Best F1 on Cross Validation Set : 0.615385

(you should see a value epsilon of about 1.38e-18)

(you should see a Best F1 value of 0.615385)

12. Код реализации:

print('\n# Anomalies found: %d' % np.sum(p.pdf(X) < epsilon))

Результат выполнения:

# Anomalies found: 117

Т. е. при подобранном коэффициенте эпсилон равным 1.38e-18 было найдено 117 аномальных данных.

**Программный код:**

from \_\_future\_\_ import division

from goto import with\_goto

from scipy.stats import multivariate\_normal

import scipy.stats as stats

import scipy.io as sio

import matplotlib.pyplot as plt

import os

import numpy as np

def estimate\_gaussian(X):

# Useful variables

m, n = X.shape

mu = np.mean(X, axis=0)

# For Sigma2,np.sum requires an axis else it flattens the array and takes the sum which is wrong.

sigma2 = (1/m)\*(np.sum((X-mu)\*\*2, axis=0))

return mu, sigma2

def select\_threshold(y\_val, p\_val):

best\_epsilon, best\_F1 = 0, 0

step\_size = (max(p\_val) - min(p\_val)) / 1000

for epsilon in np.arange(p\_val.min(),p\_val.max(),step\_size):

predictions = (p\_val<epsilon)[:,np.newaxis]

tp = np.sum(predictions[y\_val==1]==1)

fp = np.sum(predictions[y\_val==0]==1)

fn = np.sum(predictions[y\_val==1]==0)

prec = tp / (tp + fp)

rec = tp / (tp + fn)

F1 = 2 \* prec \* rec / (prec + rec)

if F1 > best\_F1:

best\_epsilon = epsilon

best\_F1 = F1

return best\_epsilon, best\_F1

@with\_goto

def main():

goto .task

label .task

# 1

file\_path = os.path.join(os.path.dirname(\_\_file\_\_), 'data', 'ex8data1.mat')

dataset = sio.loadmat(file\_path)

X = dataset["X"]

Xval = dataset["Xval"]

yval = dataset["yval"]

# 2

plt.scatter(X[:, 0], X[:, 1], marker="o", s=16)

plt.xlim(0, 30)

plt.ylim(0, 30)

plt.xlabel("Latency (ms)")

plt.ylabel("Throughput (mb/s)")

plt.show()

# 3

mu, sigma2 = estimate\_gaussian(X)

# 4

p = multivariate\_normal(mu, np.diag(sigma2))

print(mu, sigma2)

# 5

xs, ys = np.mgrid[0:30:0.1, 0:30:0.1]

pos = np.empty(xs.shape + (2,))

pos[:, :, 0] = xs

pos[:, :, 1] = ys

plt.figure(figsize=(8, 6))

plt.plot(X.T[0], X.T[1], 'o', ms=4)

plt.contour(xs, ys, p.pdf(pos), 10.\*\*np.arange(-21, -2, 3))

plt.xlabel('Latency (ms)')

plt.ylabel('Throughput (mb/s)')

plt.xlim(0, 30)

plt.ylim(0, 30)

plt.show()

# 6

p\_val = p.pdf(Xval)

epsilon, F1 = select\_threshold(yval, p\_val)

print("Best epsilon found using cross-validation:", epsilon)

print("Best F1 on Cross Validation Set:", F1)

# 7

outliers = X[p.pdf(X) < epsilon]

plt.figure(figsize=(8, 6))

plt.plot(X.T[0], X.T[1], 'o', ms=4)

plt.plot(outliers.T[0], outliers.T[1], 'o', ms=18, mfc='none', mec='r')

plt.contour(xs, ys, p.pdf(pos), 10.\*\*np.arange(-21, -2, 3))

plt.xlabel('Latency (ms)')

plt.ylabel('Throughput (mb/s)')

plt.xlim(0, 30)

plt.ylim(0, 30)

plt.show()

# 8

file\_path = os.path.join(os.path.dirname(\_\_file\_\_), 'data', 'ex8data2.mat')

dataset = sio.loadmat(file\_path)

X, Xval, yval = dataset['X'], dataset['Xval'], dataset['yval'][:, 0]

# 9

mu, sigma2 = estimate\_gaussian(X)

# 10

p = multivariate\_normal(mu, np.diag(sigma2))

# 11

p\_val = p.pdf(Xval)

epsilon, F1 = select\_threshold(yval, p\_val)

# 12

print('Best epsilon found using cross-validation: %.2e' % epsilon)

print('Best F1 on Cross Validation Set : %f\n' % F1)

print(' (you should see a value epsilon of about 1.38e-18)')

print(' (you should see a Best F1 value of 0.615385)')

print('\n# Anomalies found: %d' % np.sum(p.pdf(X) < epsilon))

if \_\_name\_\_ == '\_\_main\_\_':

main()