

Лабораторная работа №8 “Выявление аномалий”

Долматович Алина, 858641

```
In [1]: 1 from scipy.io import loadmat
        2 import matplotlib.pyplot as pyplot
        3 import numpy as np
        4 from scipy.stats import norm
        5 import math
        6 import seaborn as sns
```

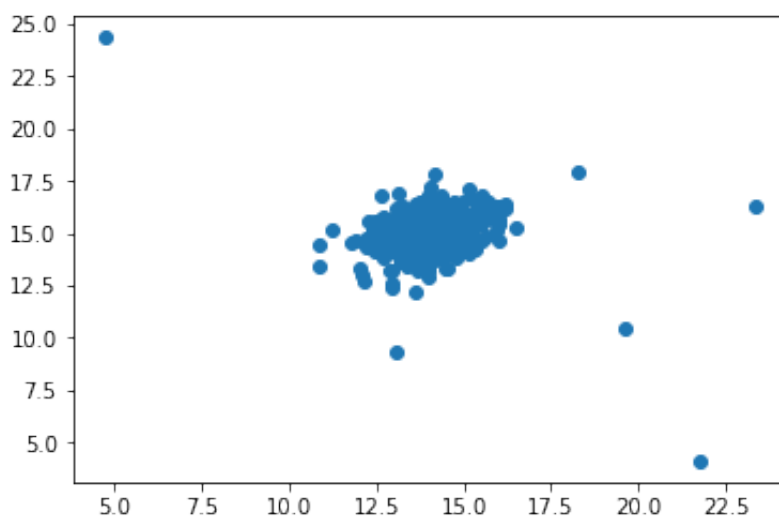
Загрузите данные ex8data1.mat из файла.

```
In [2]: 1 data = loadmat('ex8data1.mat')
        2
        3 x = data['X']
        4
        5 xVal = data['Xval']
        6 yVal = data['yval']
        7
        8 print(x.shape, xVal.shape, yVal.shape)

((307, 2), (307, 2), (307, 1))
```

Постройте график загруженных данных в виде диаграммы рассеяния.

```
In [3]: 1 pyplot.scatter(x[:, 0], x[:, 1])
        2 pyplot.show()
```



Представьте данные в виде двух независимых нормально распределенных случайных величин.

```
In [4]: 1 def normalTransform(x):
2         normal = []
3         for i in range(x.shape[1]):
4             mu, sigma = norm.fit(x[:, i])
5             value = 1 / math.sqrt(2 * math.pi * sigma ** 2) * math.e *
6             normal.append(value)
7         return np.array(normal).T
8
9 normalX = normalTransform(x)
10 print(normalX.shape)

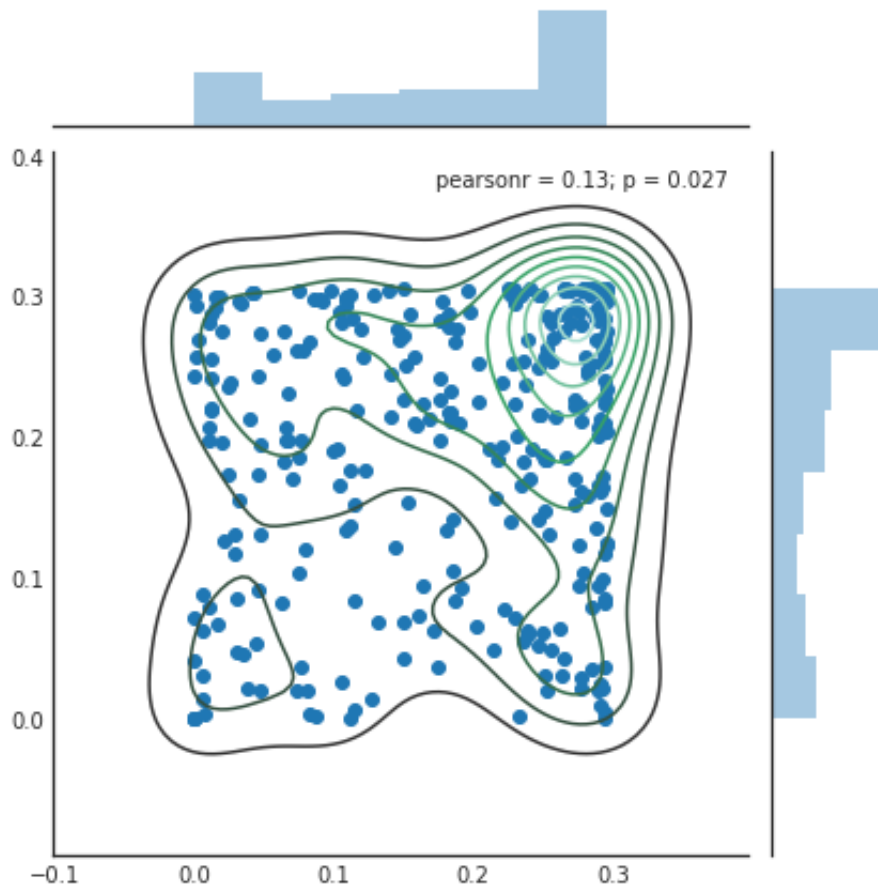
(307, 2)
```

Оцените параметры распределений случайных величин.

```
In [5]: 1 def checkParameters(x):
2         for i in range(x.shape[1]):
3             mu, std = norm.fit(x[:, i])
4             print(mu, std)
5
6         # checkParameters(x)
7         # checkParameters(normalX)
```

Постройте график плотности распределения получившейся случайной величины в виде изолиний, совместив его с графиком из пункта 2.

```
In [6]: 1 def densityPlot(x):  
2     with sns.axes_style('white'):  
3         sns.jointplot(x[:,0],x[:,1]).plot_joint(sns.kdeplot)  
4  
5     pyplot.show()  
6  
7 densityPlot(normalX)
```



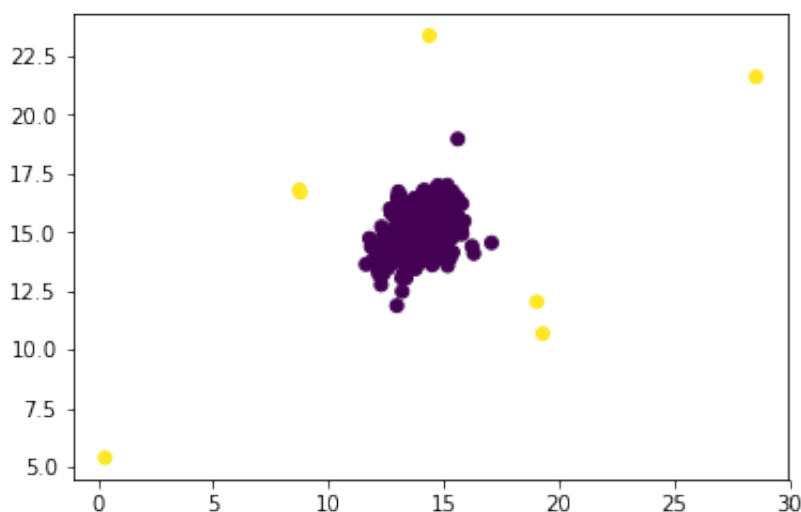
Подберите значение порога для обнаружения аномалий на основе валидационной выборки. В качестве метрики используйте F1-меру.

```
In [7]: 1 def f1Score(ypred, yact):
2         tp, tn, fp, fn = 0., 0., 0., 0.
3         for yp, ya in zip(ypred, yact):
4             if ya==1 and yp==1:
5                 tp+=1
6             if ya==0 and yp==0:
7                 tn+=1
8             if yp==0 and ya==1:
9                 fn+=1
10            if yp==1 and ya==0:
11                fp+=1
12            precision = tp/(tp+fp)
13            recall = tp/(tp+fn)
14            f1 = 2 * (precision * recall) / (precision + recall)
15            return f1
```

Выделите аномальные наблюдения на графике из пункта 5 с учетом выбранного порогового значения.

```
In [8]: 1 def zEvaluation(x, threshold=3.075):
2         prediction = []
3         rows, columns = x.shape
4
5         for row in range(rows):
6             for column in range(columns):
7                 mean = x[:, column].mean()
8                 std = x[:, column].std()
9                 if abs(x[row, column] - mean) / std > threshold:
10                    prediction.append(1)
11                    break
12             if column == columns - 1:
13                 prediction.append(0)
14         return np.array(prediction)
15
16 prediction = zEvaluation(xVal)
17 print(prediction.shape)
18 f1Score(prediction, yVal)
19
20 pyplot.scatter(xVal[:,0], xVal[:,1], c=prediction)
21 pyplot.show()
```

(307,)



Загрузите данные ex8data2.mat из файла.

```
In [9]: 1 data2 = loadmat('ex8data2.mat')
2
3 x = data2['X']
4
5 xVal = data2['Xval']
6 yVal = data2['yval']
7
8 print(x.shape, xVal.shape, yVal.shape)
```

((1000, 11), (100, 11), (100, 1))

Представьте данные в виде 11-мерной нормально распределенной случайной величины.

```
In [10]: 1 normalX = normalTransform(x)
          2 print(normalX.shape)

(1000, 11)
```

Оцените параметры распределения случайной величины.

```
In [11]: 1 # checkParameters(x)
          2 # checkParameters(normalX)
```

Подберите значение порога для обнаружения аномалий на основе валидационной выборки. В качестве метрики используйте F1-меру.

```
In [12]: 1 threshold=3
          2 prediction = zEvaluation(xVal, threshold)
          3 f1Score(prediction, yVal)
```

Out[12]: 0.625

Выделите аномальные наблюдения в обучающей выборке. Сколько их было обнаружено? Какой был подобран порог?

```
In [13]: 1 pyplot.scatter(xVal[:,0], xVal[:,1], c=prediction)
          2 pyplot.show()
```

