МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ

ИНСТИТУТ ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Лабораторные работы по дисциплине

«Введение в машинное обучение»

Выполнил:

Студент гр.09-305

Шарафеев Марат

Преподаватель:

Шустова Е.П.

Казань-2024

**Задание 6. Наивный** **Байесовский** **классификатор.**

1. Подберите свои данные для решения задачи классификации. Нужно воспользоваться известными репозиториями данных. Приведите к раткое описание данных. Сформируйте обучающую и тестовую выборки.
2. Решите задачу классификации с помощью наивного Байесовского классификатора; если число признаков = 2, то визуализируйте данные.
3. Постройте кросс-валидационную таблицу, сделайте вывод о точности решения задачи классификации.
4. Задайте несколько новых данных, покажите соответствующие точки на графике (выделите их другим цветом).
5. Определите класс для новых данных.
6. Приведите полный код для решения задачи классификации с помощью наивного Байесовского классификатора.

**Ход работы:**

**Описание данных**

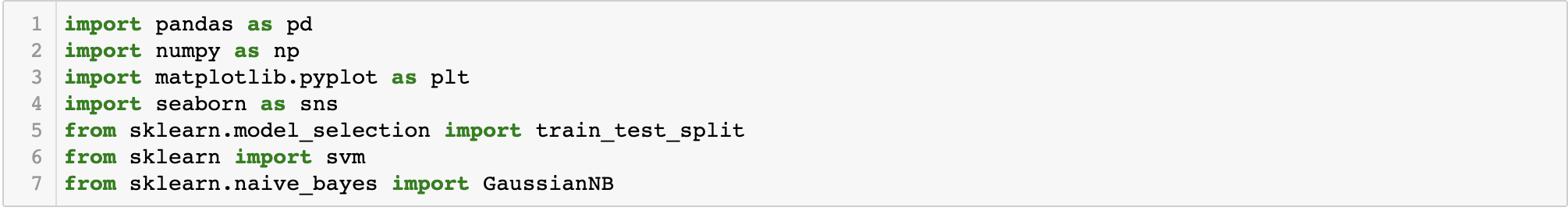
Для задачи классификации были выбраны данные о пожарах в двух регионах Алжира, а именно региона Беджая и Сиди-Бель-Аббес. Набор состоит из 244 строк.

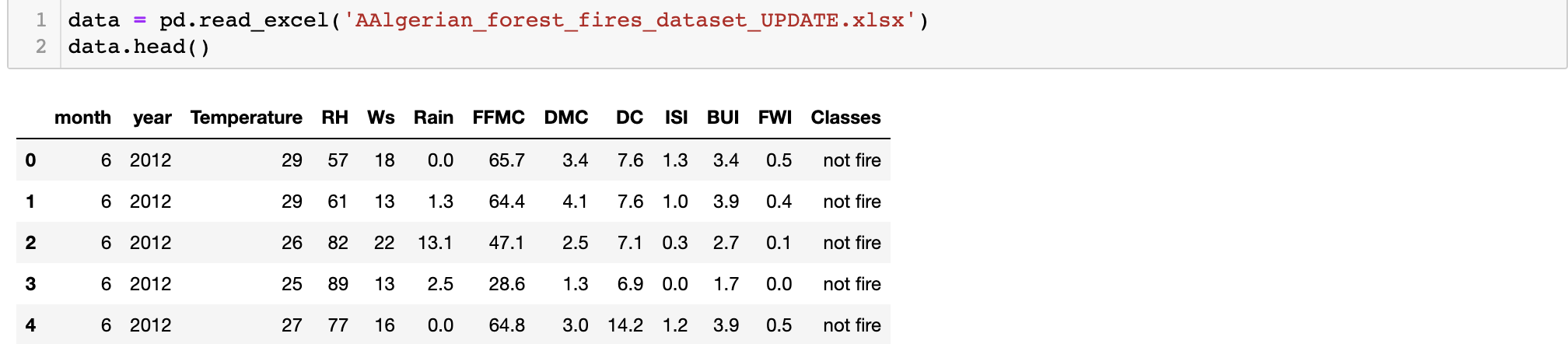
Данные были получены с [Algerian Forest Fires Dataset Data Set](https://archive.ics.uci.edu/dataset/547/algerian+forest+fires+dataset).

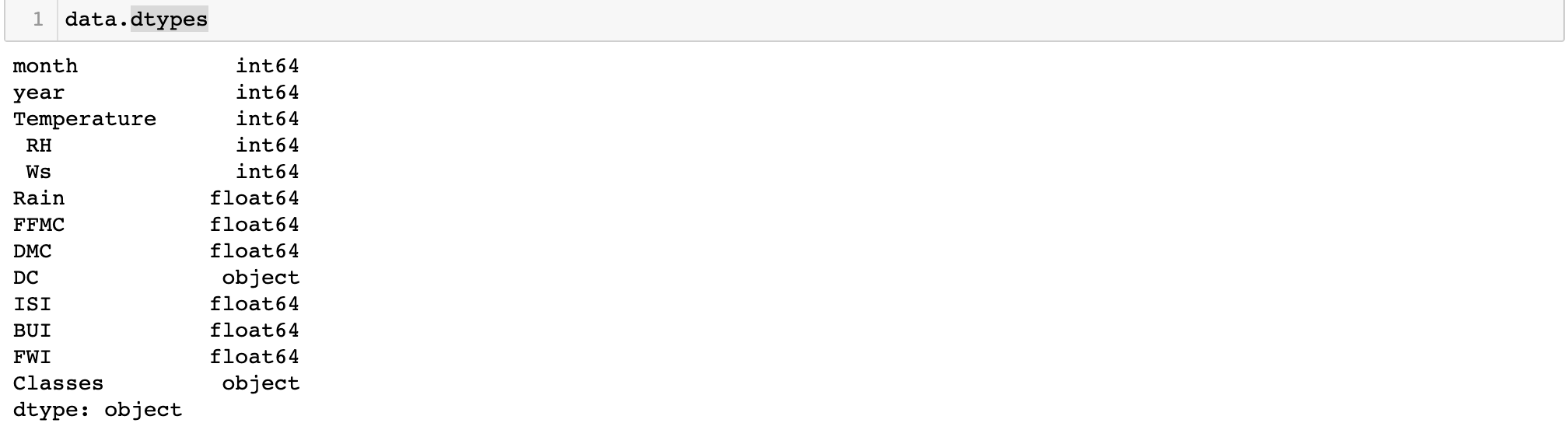
Датасет содержит следующие атрибуты:

* Month - месяц
* Year - год
* Temperature - температура в полдень (максимальная температура) в градусах Цельсия: от 22 до 42
* RH - Относительная влажность в %: от 21 до 90
* Ws - Скорость ветра в км/ч
* Rain - total day in mm: 0 to 16.8
* FFMC - Индекс точного определения влажности топлива (FFMC) по системе FWI: от 28,6 до 92,5. 7.
* DMC - Индекс влажности Даффа (DMC) по системе FWI: от 1,1 до 65,9
* DC - Индекс кода засухи (DC) по системе FWI: от 7 до 220,4
* ISI - Индекс начального спреда (ISI) из системы FWI: от 0 до 18,5
* BUI - Индекс Buildup Index (BUI) по системе FWI: от 1,1 до 68
* FWI - Индекс пожарной погоды (FWI): от 0 до 31,1. 12.
* Classes - Классы: два класса, а именно «Огонь» и «Не огонь».

Подключим необходимые библиотеки и считаем данные в переменную data:





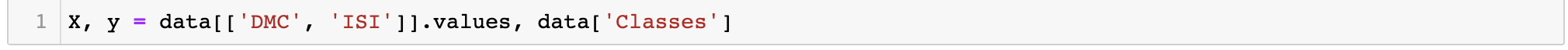


Посмотрим на корреляцию данных:

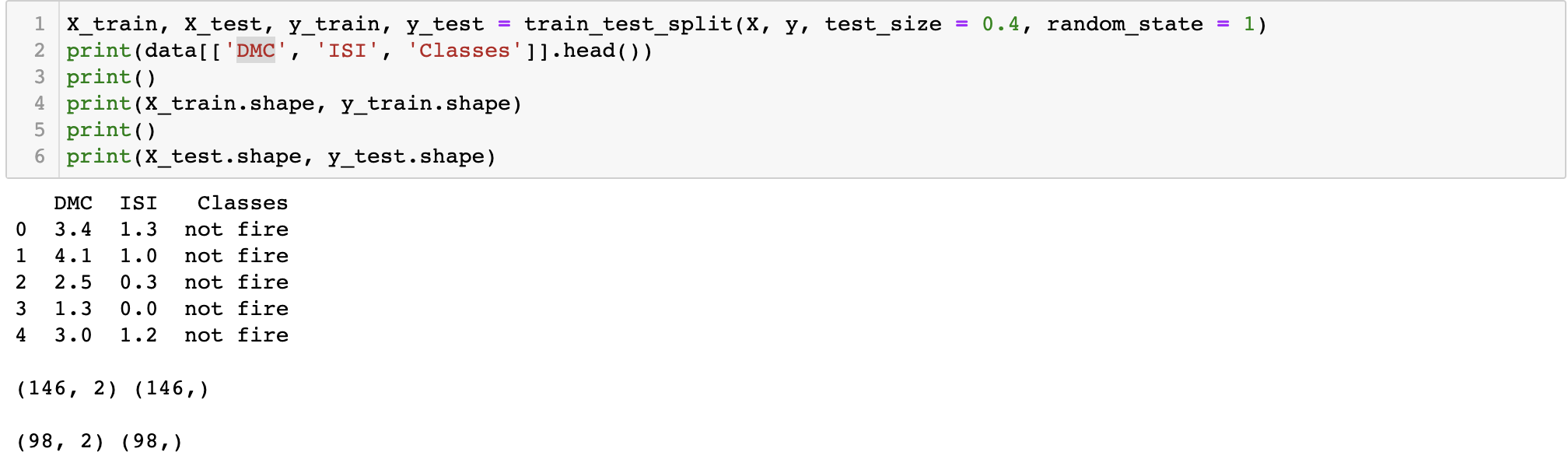


Для удобства будем использовать два параметра:

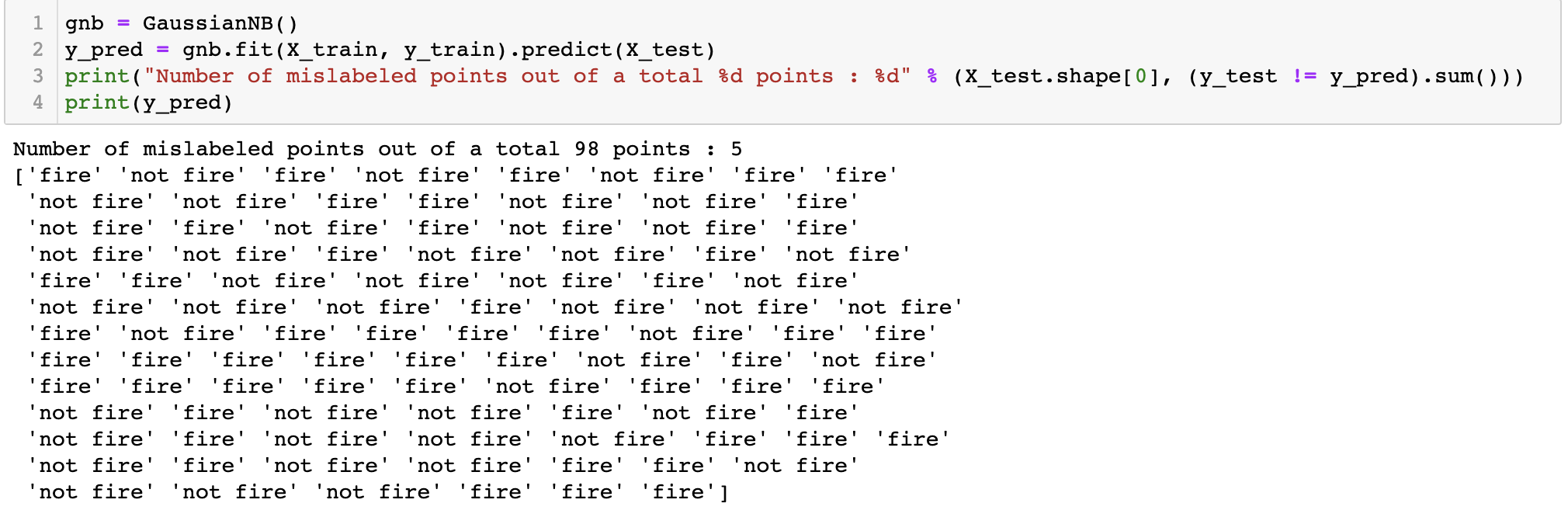
* DMC - Индекс влажности Даффа (DMC) по системе FWI: от 1,1 до 65,9
* ISI - Индекс начального спреда (ISI) из системы FWI: от 0 до 18,5
* Classes - Классы: два класса, а именно «Огонь» и «Не огонь».



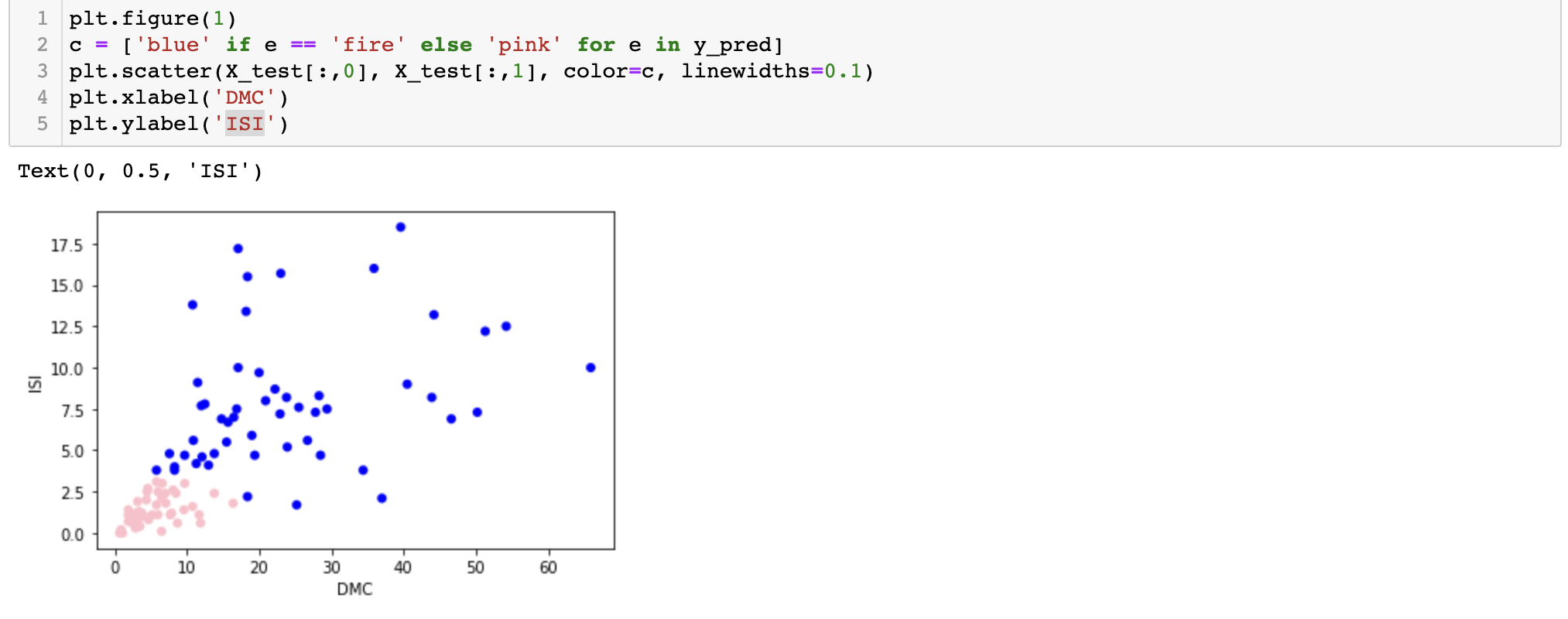
Сформируем обучающие и тестовые выборки:



Решим задачу классификации с помощью наивного Байесовского классификатора:



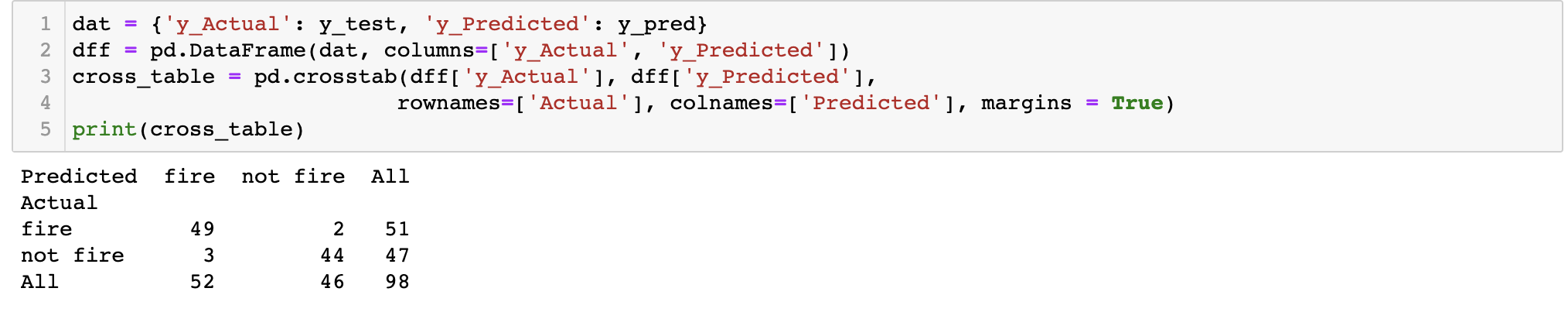
Визуализируем данные:



В обучающую выборку классы попали в таком соотношении:

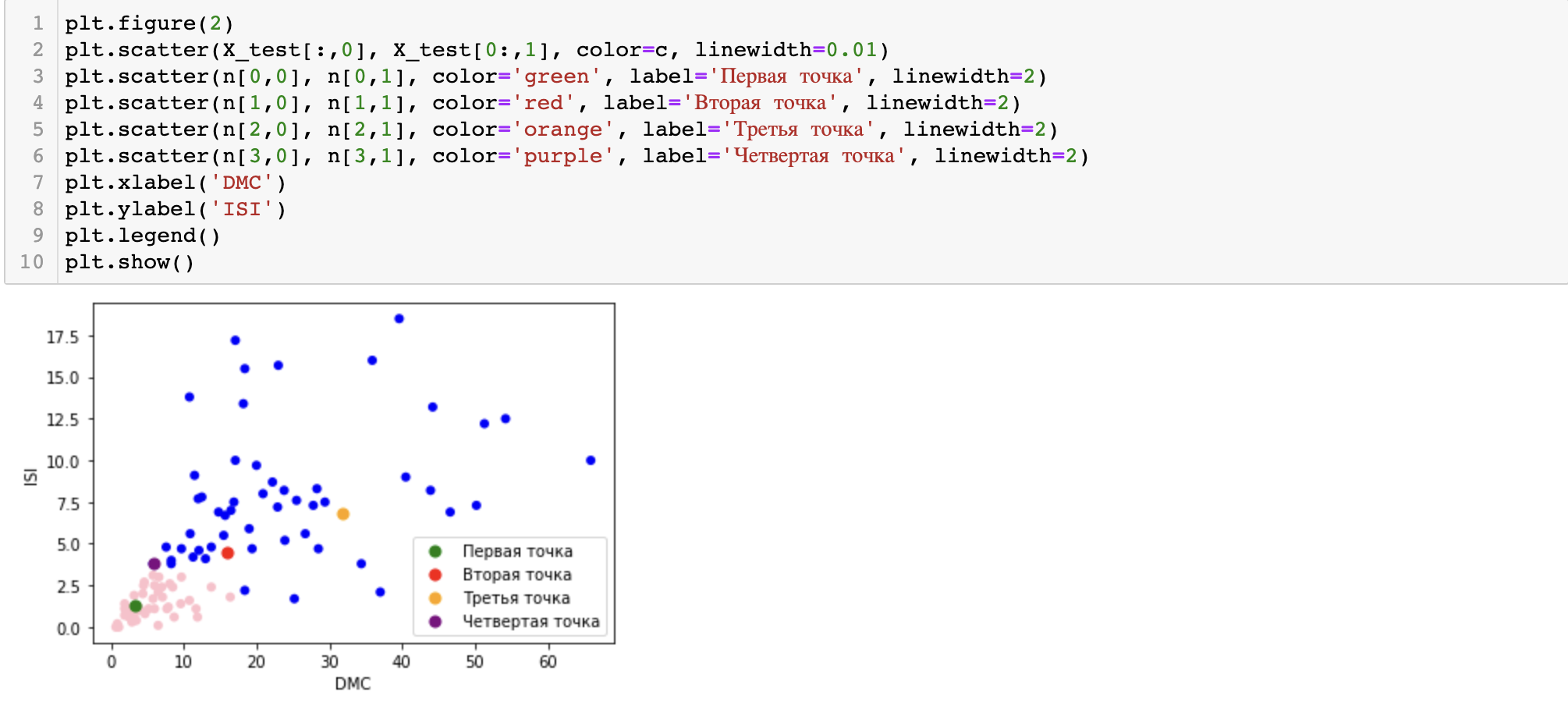
* «fire» - 53,06 %
* «not fire» - 46,94%

Построим кросс-валидационную таблицу и сделаем вывод о точности решения задачи классификации



Модель допустила ошибки в 5 случаях из 98, что соответствует точности прогноза в 94,9 %.

Зададим несколько новых данных и покажем соответствующие точки на графике.

Определим класс для данных из тестовый выбоки:



**Приложение:**

|  |
| --- |
| import pandas as pd  import numpy as np  import matplotlib.pyplot as plt  import seaborn as sns  from sklearn.model\_selection import train\_test\_split  from sklearn import svm  from sklearn.naive\_bayes import GaussianNB |
| data = pd.read\_excel('AAlgerian\_forest\_fires\_dataset\_UPDATE.xlsx')  data.head() |
| data.dtypes |
| import plotly.express as px  fig = px.imshow(data.corr())  fig.show() |
| data['Classes'] = data['Classes'].str.strip() |
| X, y = data[['DMC', 'ISI']].values, data['Classes'] |
| X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.4, random\_state = 1)  print(data[['DMC', 'ISI', 'Classes']].head())  print()  print(X\_train.shape, y\_train.shape)  print()  print(X\_test.shape, y\_test.shape) |
| gnb = GaussianNB()  y\_pred = gnb.fit(X\_train, y\_train).predict(X\_test)  print("Number of mislabeled points out of a total %d points : %d" % (X\_test.shape[0], (y\_test != y\_pred).sum()))  print(y\_pred) |
| plt.figure(1)  c = ['blue' if e == 'fire' else 'pink' for e in y\_pred]  plt.scatter(X\_test[:,0], X\_test[:,1], color=c, linewidths=0.1)  plt.xlabel('DMC')  plt.ylabel('ISI') |
| dat = {'y\_Actual': y\_test, 'y\_Predicted': y\_pred}  dff = pd.DataFrame(dat, columns=['y\_Actual', 'y\_Predicted'])  cross\_table = pd.crosstab(dff['y\_Actual'], dff['y\_Predicted'],  rownames=['Actual'], colnames=['Predicted'], margins = True)  print(cross\_table) |
| n = np.array([[3.4, 1.3],  [16.0, 4.5],  [31.9, 6.8],  [5.8, 3.8]]) |
| plt.figure(2)  plt.scatter(X\_test[:,0], X\_test[0:,1], color=c, linewidth=0.01)  plt.scatter(n[0,0], n[0,1], color='green', label='Первая точка', linewidth=2)  plt.scatter(n[1,0], n[1,1], color='red', label='Вторая точка', linewidth=2)  plt.scatter(n[2,0], n[2,1], color='orange', label='Третья точка', linewidth=2)  plt.scatter(n[3,0], n[3,1], color='purple', label='Четвертая точка', linewidth=2)  plt.xlabel('DMC')  plt.ylabel('ISI')  plt.legend()  plt.show() |
| new\_points = gnb.fit(X\_train, y\_train).predict(n)  print(new\_points) |