МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА №41

ЗАЩИЩЕНА С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| К.т.н., доц. |  |  |  | Е.Л. Турнецкая |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №4 |
| Применение методов классификации |
| по курсу: Методология и технология проектирования информационных систем |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | М320М |  |  |  | П. Е. Лукьянец |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2023

*Цель работы:* изучить алгоритмы и методы классификации на практике.

**Ход выполнения работы**

Для работы был выбран третий датасет из списка под названием «3cancer». В данном датасете представлена информация о пациентах, болеющих раком: id, Толщина скопления, Однородность размера клеток, Однородность формы клеток, Краевая адгезия, Размер отдельных эпителиальных клеток, Голые ядра, Бледный хроматин, Нормальные ядрышки, Митозы, Класс (2 для доброкачественных, 4 для злокачественных). Чтобы начать работу, импортируем библиотеку, а затем считываем CSV файл. Выведем первые 20 строк с помощью метода head.. Часть данных этого датасета представлена на рисунке 1.

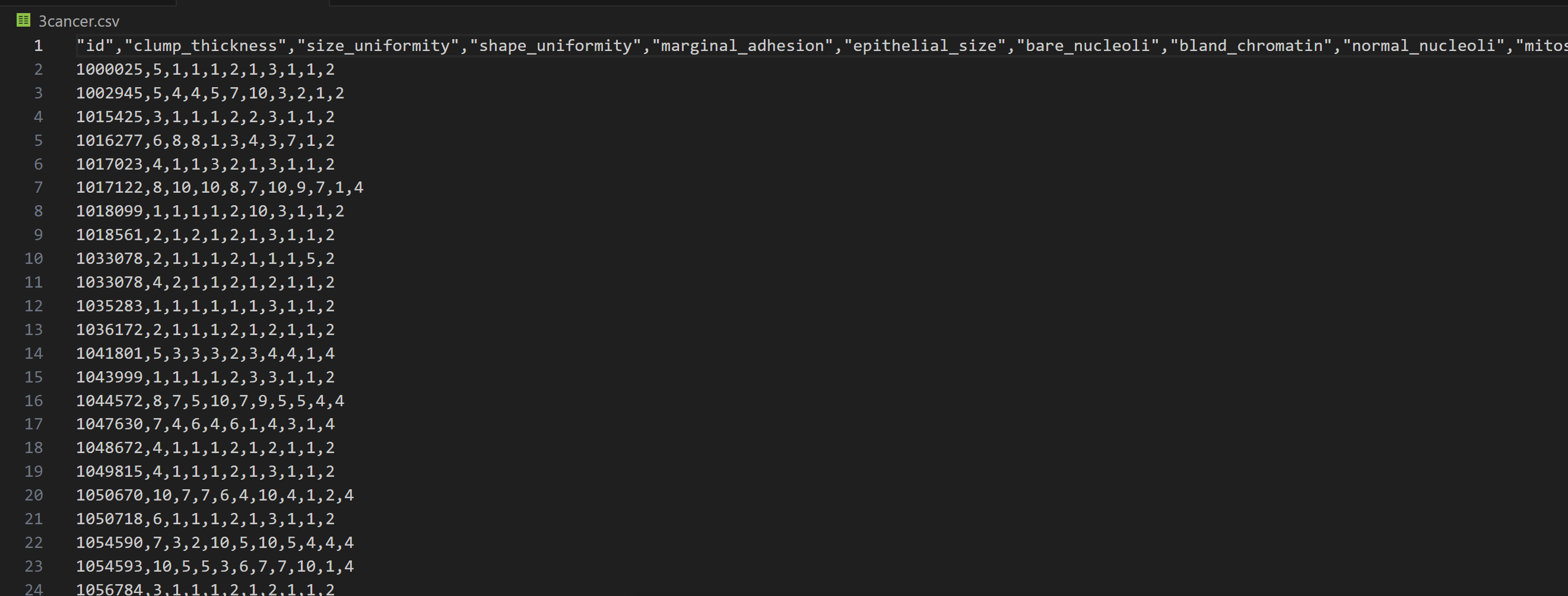


Рисунок 1 – используемый датасет

Для работы с ним использовалась библиотека Pandas.

Работа была выполнена при помощи Visual Studio Code, a также Jupyter Notebook.

Ссылка на GitHub репозиторий с файлами: https://github.com/NinjaCaratist/MTPIS

Чтобы начать работу, импортируем библиотеку, а затем считываем CSV файл.

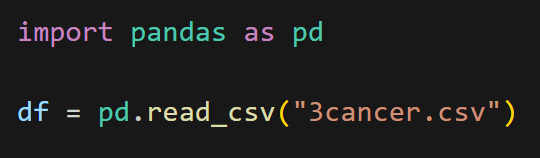


Рисунок 2 – скриншот кода

Выведем первые 20 строк с помощью метода head. На рисунке 3 показан код, а на рисунке 4 – результат его работы.

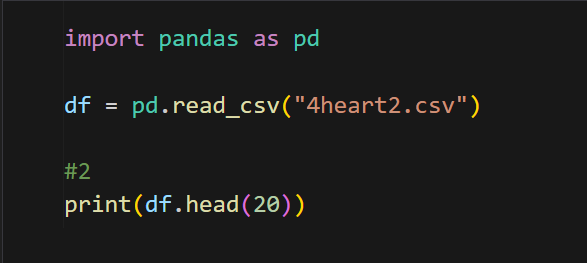
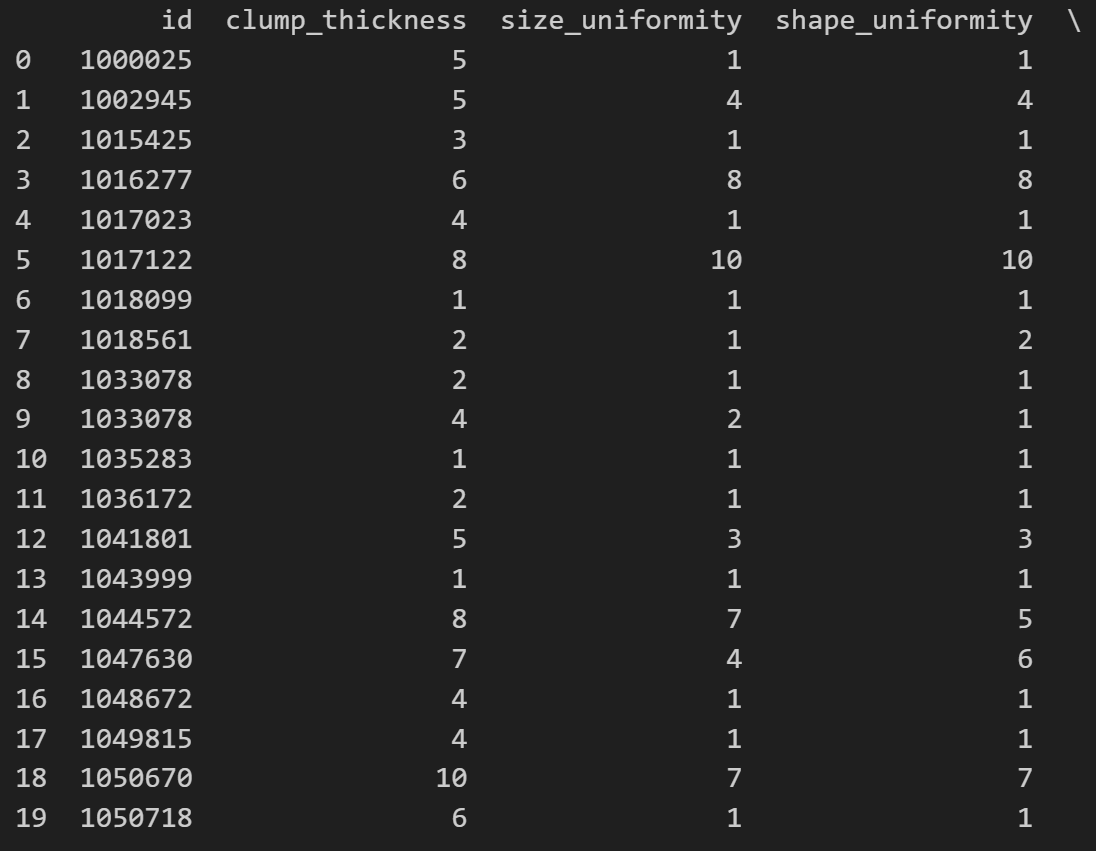
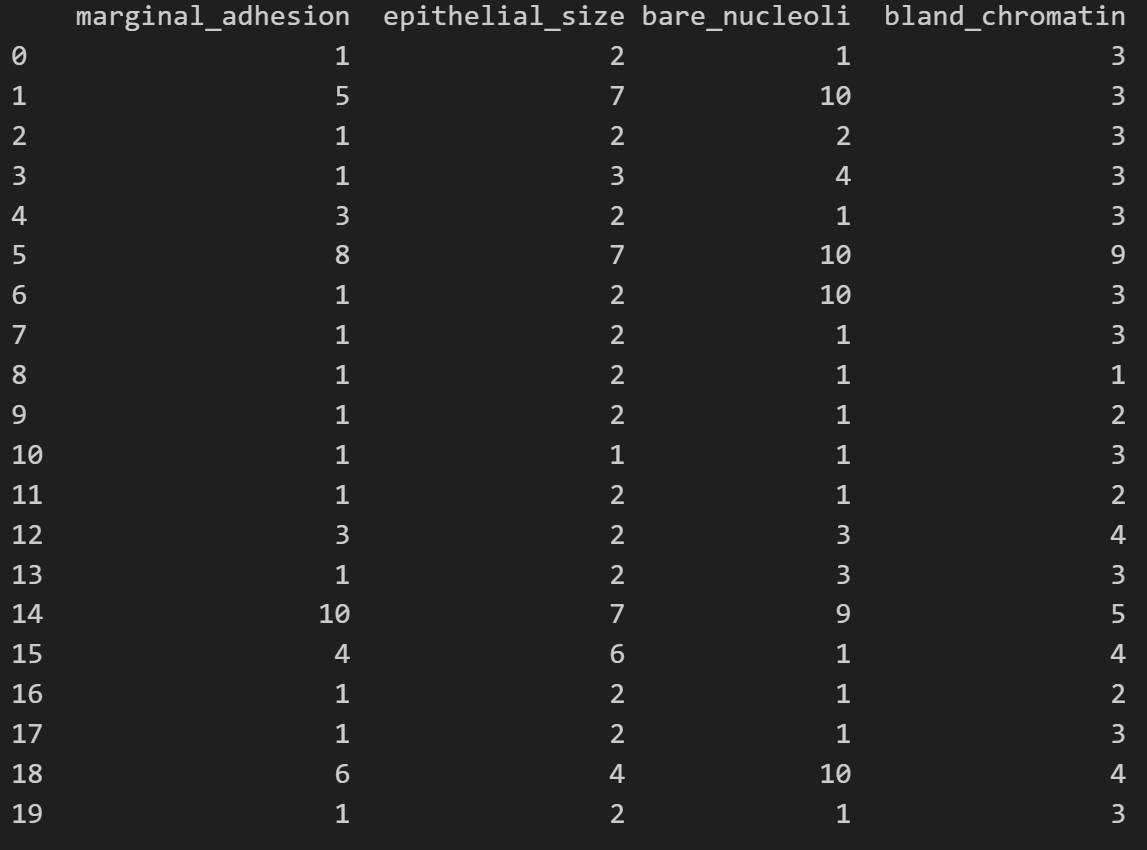


Рисунок 3 – скриншот кода





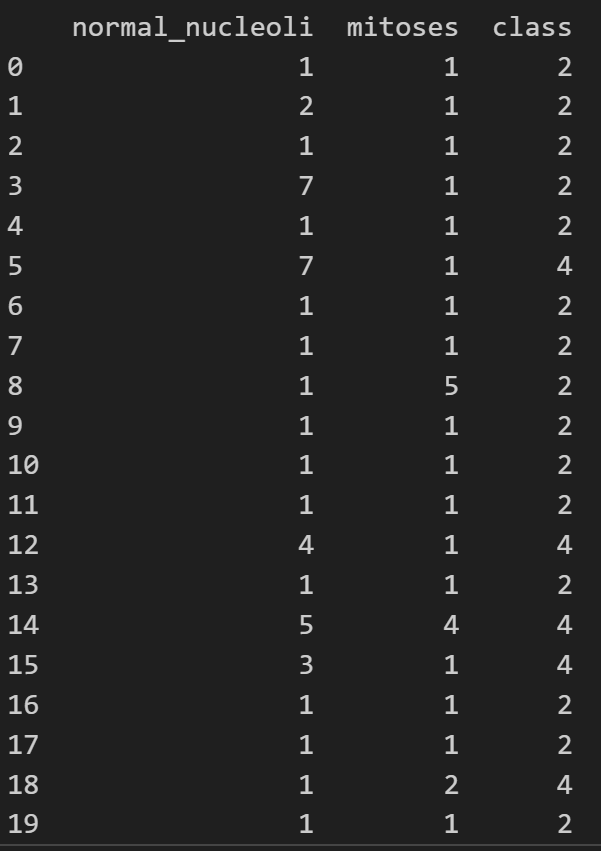


Рисунок 4 – результат вывода

Как уже было сказано, данная таблица содержит информацию об анализах пациентов с раком. Предметная область – медицина.

Опишем колонки подробнее:

1. id

2. Толщина скопления: 1–10

3. Однородность размера клеток: 1–10

4. Однородность формы клеток: 1–10

5. Краевая адгезия: 1–10

6. Размер отдельных эпителиальных клеток: 1 - 10

7. Голые ядра: 1 - 10

8. Бледный хроматин: 1 - 10

9. Нормальные ядрышки: 1 - 10

10. Митозы: 1 - 10

11. Класс: (2 для доброкачественных, 4 для

злокачественных)

Теперь с помощью метода «.info» оценим данные. Этот метод возвращает название столбцов, типы данных, количество ненулевых объектов каждом столбце. Этот метод возвращает название столбцов, типы данных, количество ненулевых объектов каждом столбце. Результат работы метода представлен на рисунке 5.

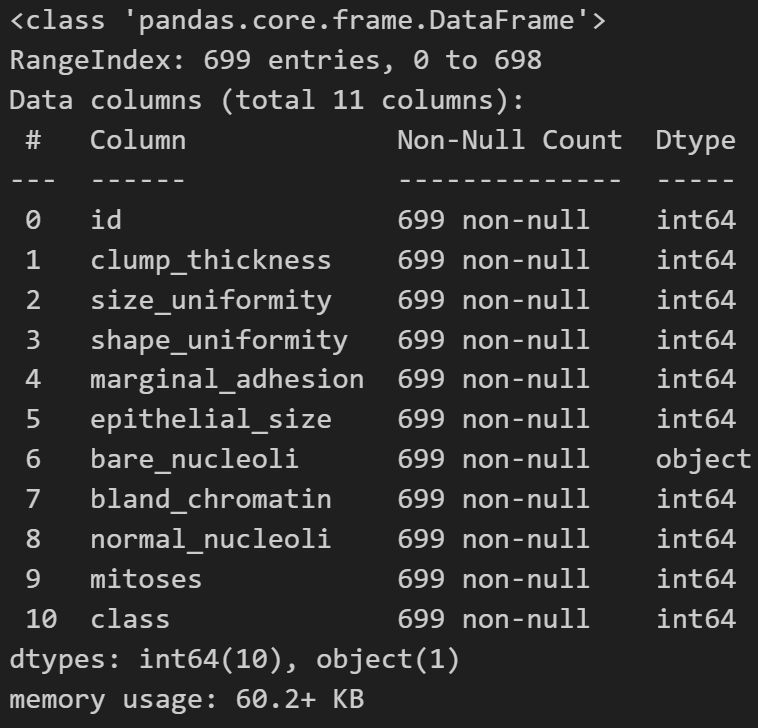


Рисунок 5 – результат вывода

Теперь выведем на экран названия столбцов с помощью df.columns. Названия всех колонок приемлемы.



Рисунок 6 – скриншот кода

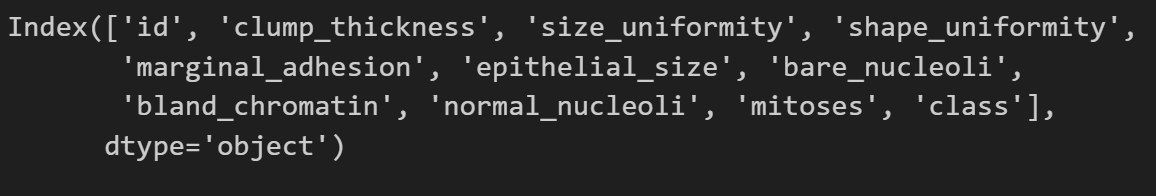


Рисунок 7 – результат вывода

Найдём пропуски и устраним их. При помощи метода «isna» найдём все пропуски в таблице, а также при помощи sum выведем количество пропусков в каждом столбце. Пропусков в данном наборе данных нет. Код представлен на рисунке 8.

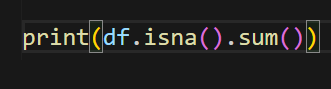


Рисунок 8 – скриншот кода

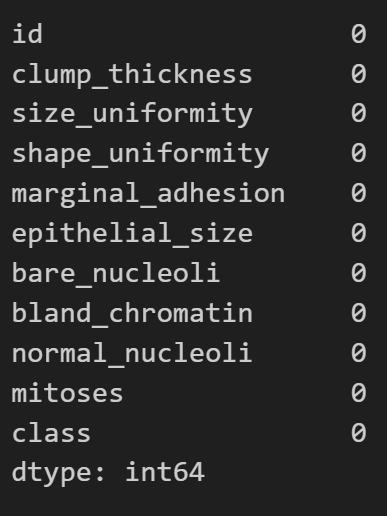
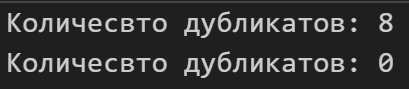


Рисунок 9 – результат вывода

Проверим данные на наличие дубликатов. Удалим дубликаты, при помощи метода "drop\_duplicates", а также перераспределим индексы. Также выведем уникальные значения каждого столбца. Столбец 'bare\_nucleoli' содержит некорректные значения '?'. Заменим их при помощи метода 'replace' на NaN, и выкинем их.



Рисунок 10 – скриншот кода



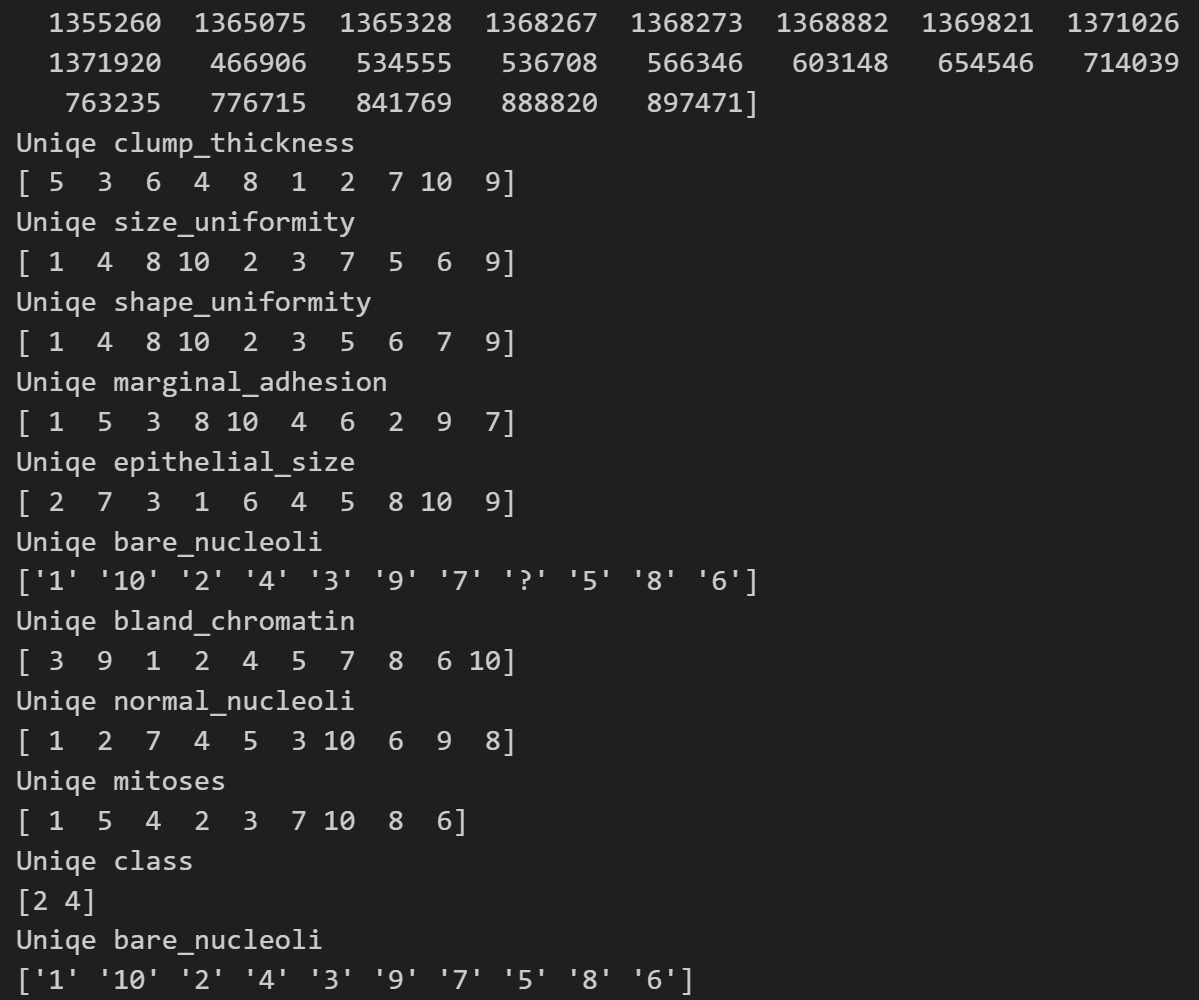
****

Рисунок 11 – результат вывода

Проверим все ли типы данных соответствуют действительности. Все столбцы, кроме Голых ядер соответствуют своему типу. Поэтому при помощи метода «to\_numeric» изменяем тип на int. Выведем информацию.

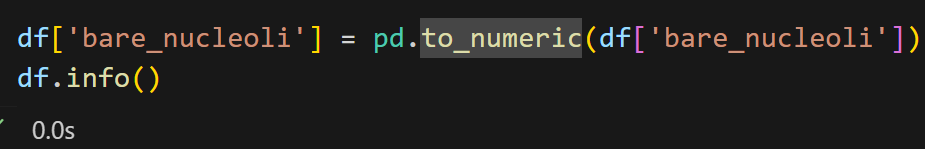


Рисунок 12 – скриншот кода

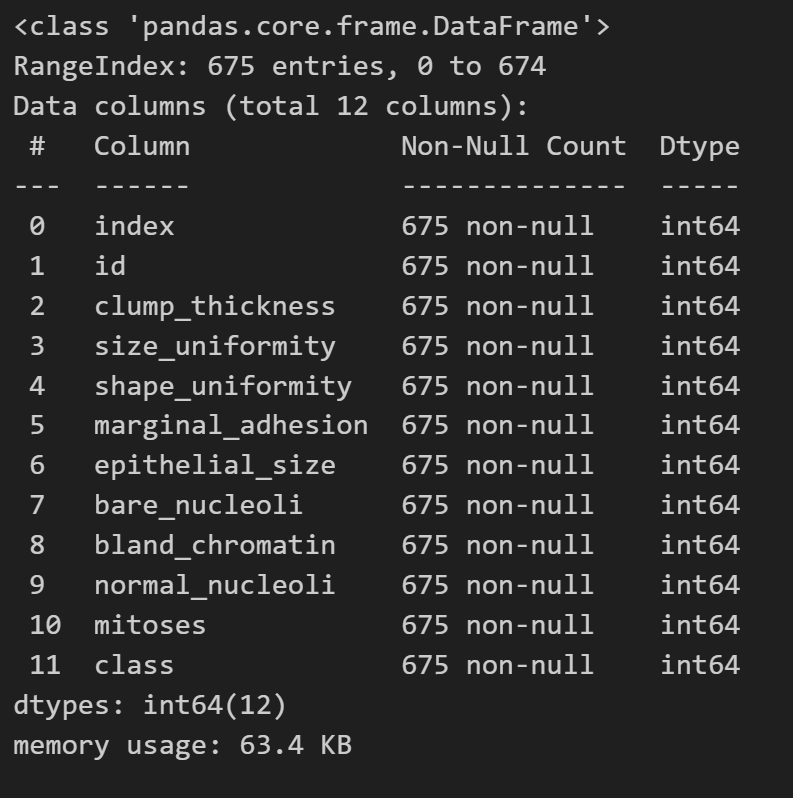


Рисунок 13 – результат вывода

Прежде чем отправить данные на вход модели и получить прогнозы, проводём — EDA, или исследовательский анализ данных. На этом этапе изучим распределения отдельных признаков и целевой переменной. Построим графики методом kdeplot().

Также построим тепловую карту корреляции, чтобы увидеть взаимосвязь между значениями и целевым признаком.

Удалим атрибут "id", так как он не несёт смысловой нагрузки для модели.



Рисунок 14 – скриншот кода

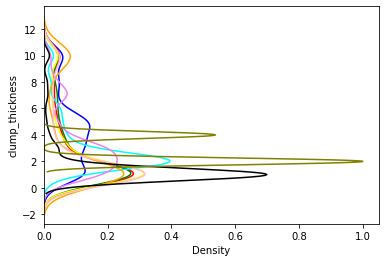


Рисунок 15 – построенный график

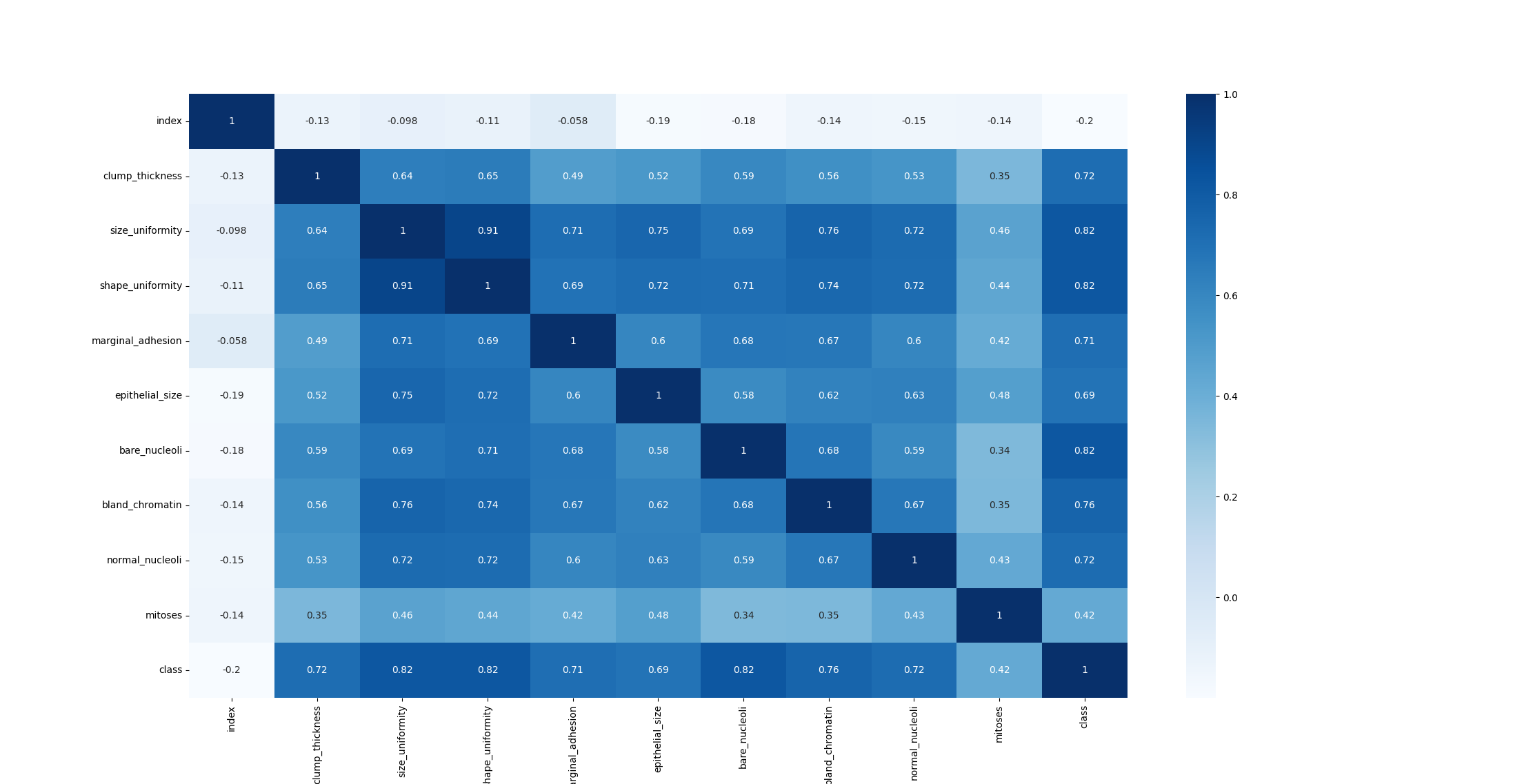


Рисунок 16 – построенная тепловая карта

Перед тем как разделять данные на тестовую и обучающую выборки и обучать на них различные модели, пропишем функцию проверки обученной модели, чтобы затем было проще её вызывать.

*Матрица ошибок*

Возможные значения классов: 2 и 4. Модель тоже может выдавать итоговый прогноз в виде значения одного из двух классов. Тогда для каждого объекта прогноз относится к одной из четырех групп:

1) Прогноз модели = 1, реальное значение = 1. Такие прогнозы называют True Positive («истинно положительные») — сокращённо TP.

2) Прогноз модели = 1, реальное значение = 0. Такие прогнозы называют False Positive («ложно положительные») — сокращённо FP.

3) Прогноз модели = 0, реальное значение = 1. Такие прогнозы называют False Negative («ложно отрицательные») — сокращённо FN.

4) Прогноз модели = 0, реальное значение = 0. Такие прогнозы называют True Negative («истинно отрицательные») — сокращённо TN

*Доля правильных ответов*

Это доля верно угаданных ответов из всех прогнозов. Чем ближе значение accuracy к 100%, тем лучше. Метрику рассчитывают функцией accuracy\_score из модуля metrics. На вход функция принимает верные и спрогнозированные значения классов на валидационной выборке.

*Точность (precision) и полнота (recall)*

Чтобы оценить модель без привязки к соотношению классов, рассчитывают эти метрики.

Precision говорит, какая доля прогнозов относительно "1" класса верна. То есть смотрим долю правильных ответов только среди целевого класса.

Вторая метрика нацелена на минимизацию противоположных рисков — recall показывает, сколько реальных объектов "1" класса вы смогли обнаружить с помощью модели.

Каждая метрика принимает значения от 0 до 1. Чем ближе к единице, тем лучше. Однако при настройке параметров модели — обычно порога вероятности, после которого мы относим объект к классу "1" — оптимизация одной метрики часто приводит к ухудшению другой. Метрики точности и полноты также реализованы в модуле metrics в функциях "precision\_score" и "recall\_score".

*F1-мера*

Так как precision и recall направлены на избежание противоположных рисков, нужна сводная метрика, учитывающая баланс между метриками. В sklearn.metrics F1-меру вычисляют методом "f1\_score".

*ROC - кривая*

Для оценки качества классификатора (модели классификации) применяют метрику roc\_auc, или площадь под кривой ошибок — AUC-ROC. Для оценки качества классификатора (модели классификации) применяют метрику roc\_auc, или площадь под кривой ошибок — AUC-ROC.

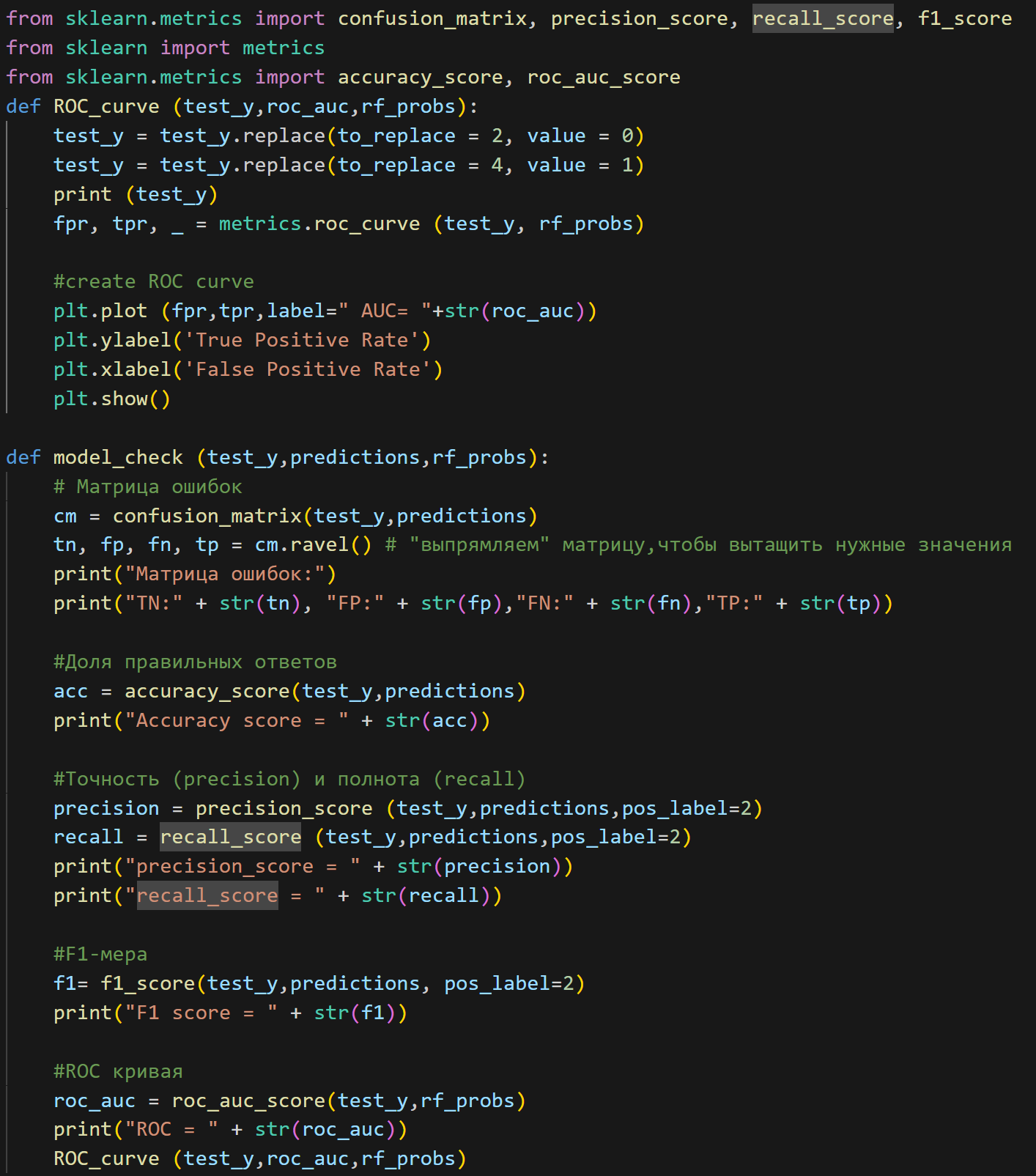


Рисунок 17 – скриншот кода

Теперь разделим набор данных на тестовую и обучающую выборки соотношением 3 к 7.

Произведём обучение и прогнозирование методом fit-predict. Каждой модели в sklearn соответствует отдельная структура данных. DecisionTreeClassifier (англ. «классификатор дерева решений») — это структура данных для классификации деревом решений.

В переменной model будет храниться модель. Что обучить модель, нужно запустить алгоритм обучения.

Разделим датафрейм на столбец с целевой переменной (class) и остальные данные. Теперь уже можно построить взаимосвязь и на её основании спрогнозировать y по новым X. Чтобы запустить обучение, вызовем метод fit() и передадим ему как параметр данные.

Чтобы построить прогнозы для набора данных, хватит одной строчки кода и вызова метода predict(). У нас осталась отложенная порция данных, для которых мы знаем признаки и ответы. На этом этапе мы берём только признаки, передаём их на вход обученной модели и сохраняем предсказанные значения.

Теперь вызываем функцию оценки качества и смотрим на оценки и график.

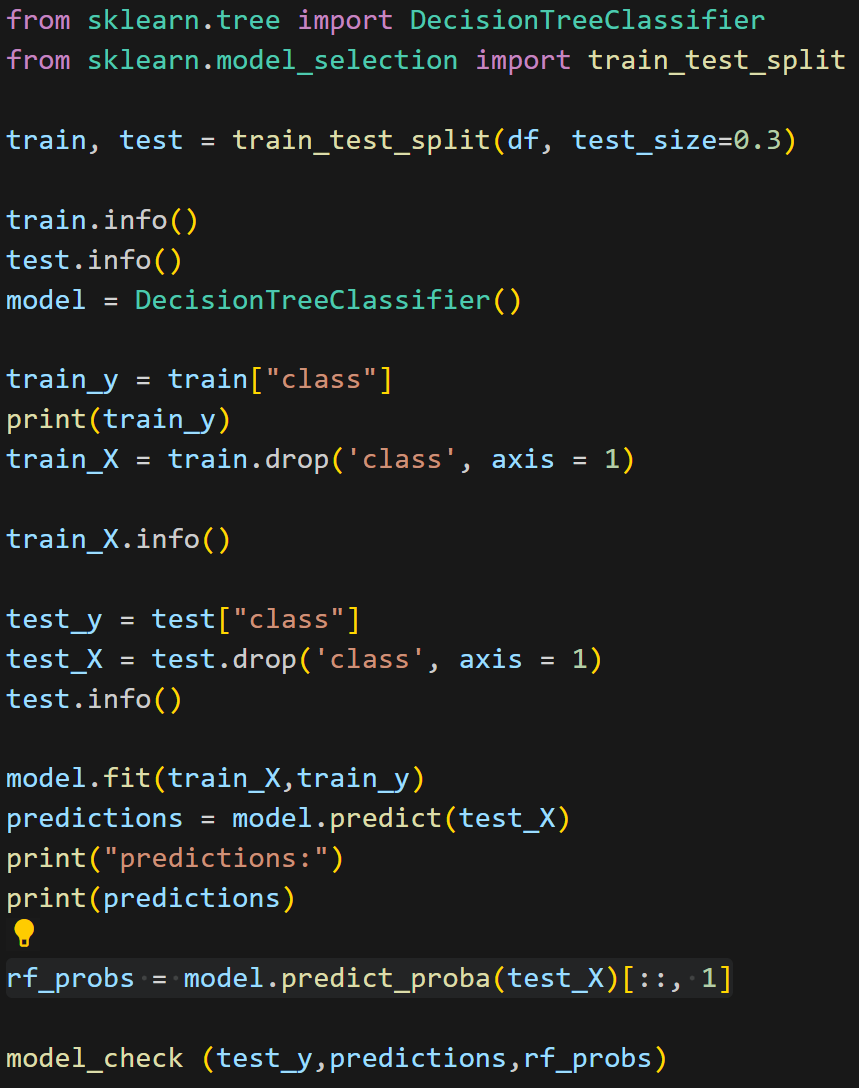
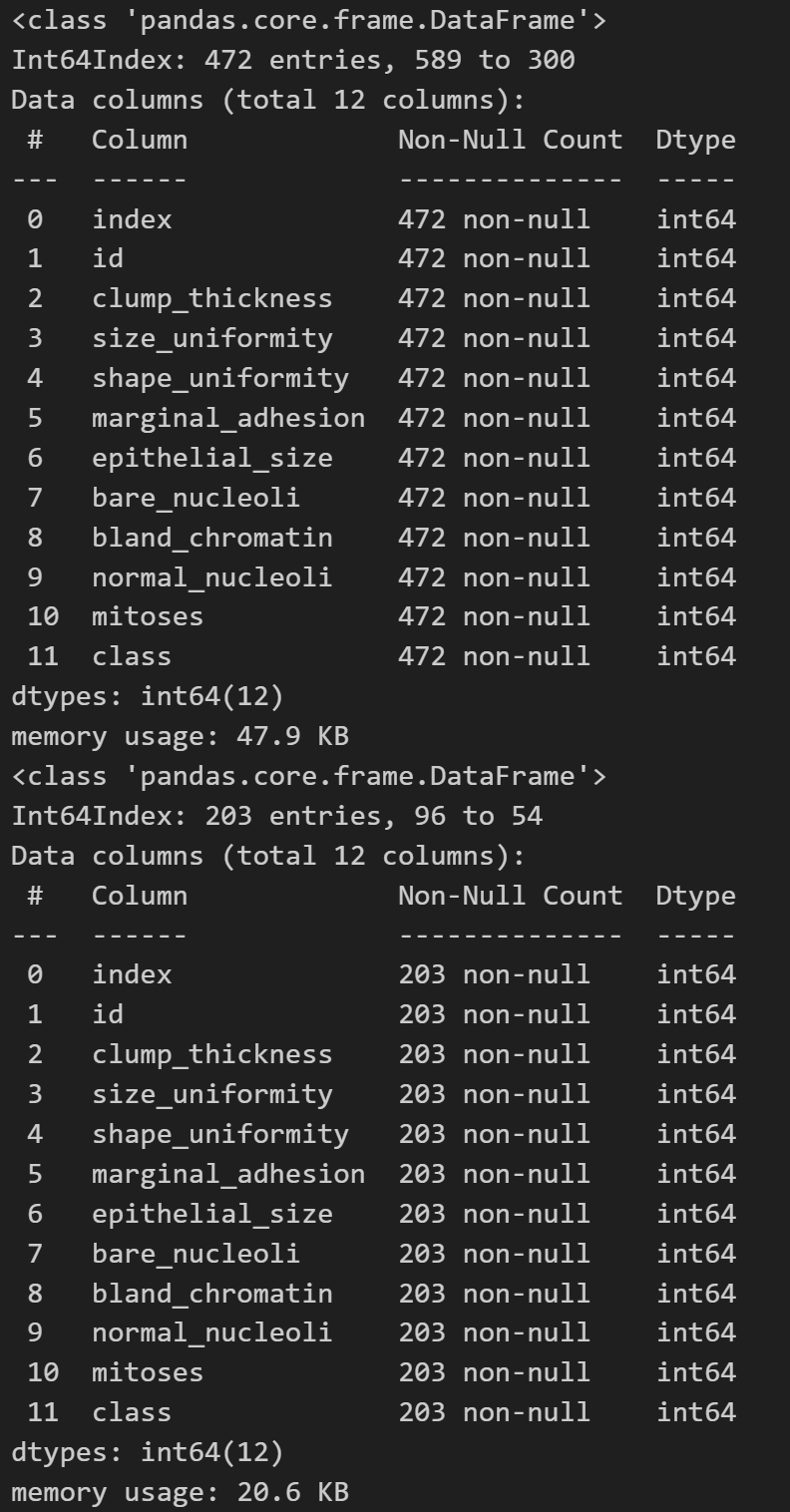


Рисунок 18 – скриншот кода



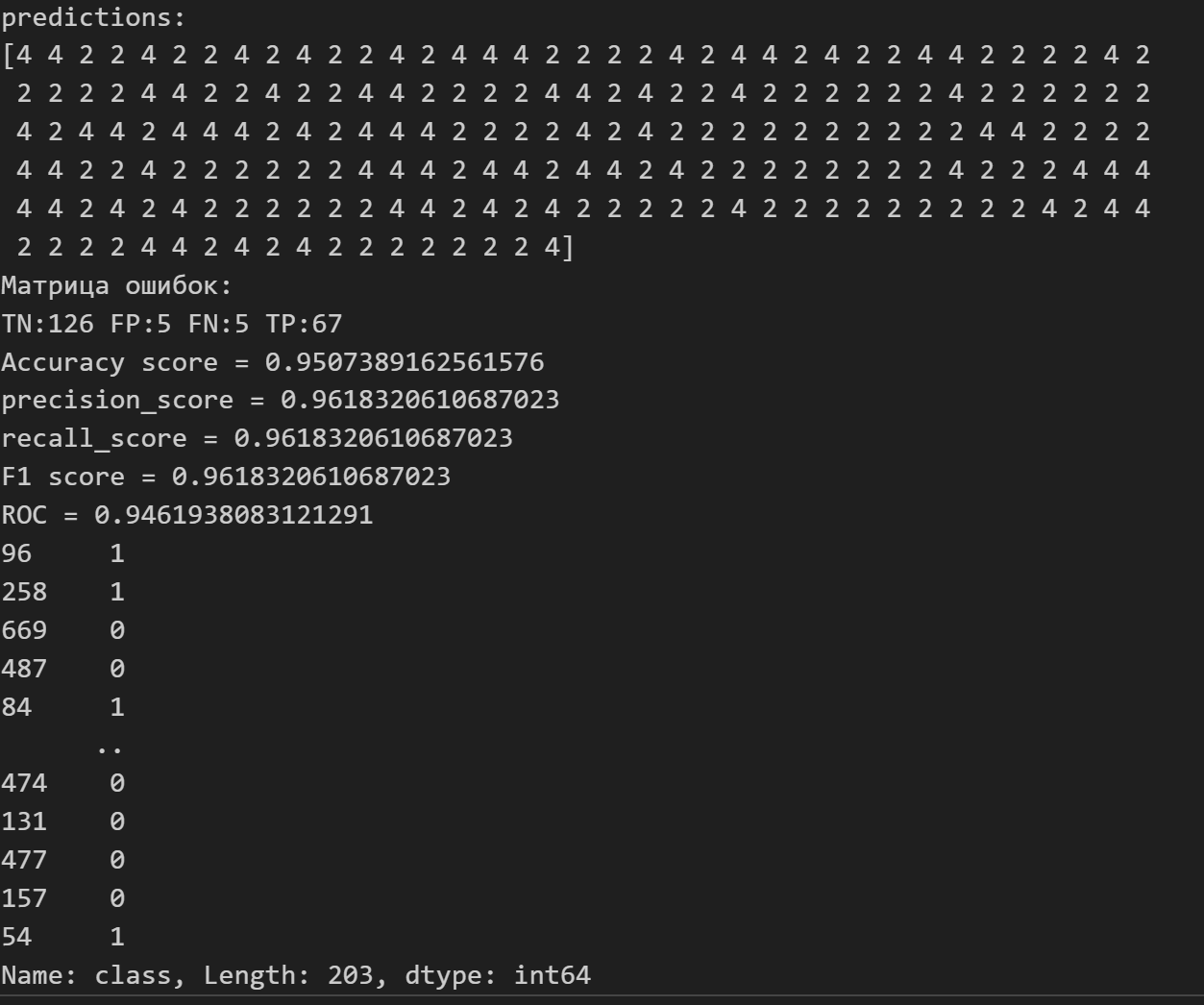


Рисунок 19 – результат вывода

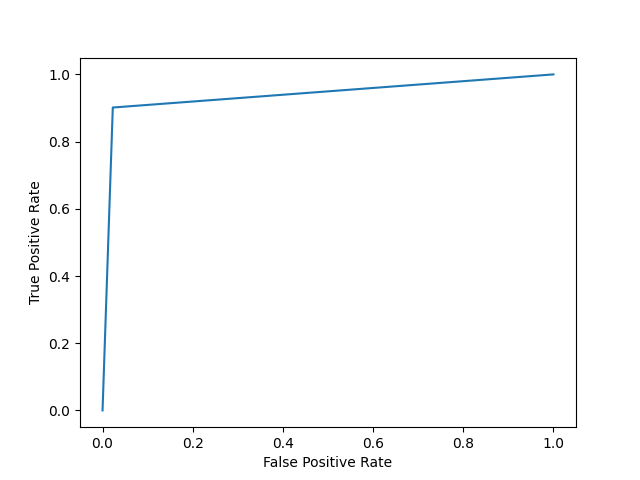


Рисунок 20 – построенный график

Снова разделим набор данных на тестовую и обучающую выборки соотношением 3 к 7.

Теперь произведём обучение и прогнозирование методом случайного леса - RandomForestClassifier.

В переменной clf будет храниться модель. Что обучить модель, нужно запустить алгоритм обучения.

Разделим датафрейм на столбец с целевой переменной (class) и остальные данные. Теперь уже можно построить взаимосвязь и на её основании спрогнозировать y по новым X. Чтобы запустить обучение, вызовем метод fit() и передадим ему как параметр данные.

Чтобы построить прогнозы для набора данных, хватит одной строчки кода и вызова метода predict(). У нас осталась отложенная порция данных, для которых мы знаем признаки и ответы. На этом этапе мы берём только признаки, передаём их на вход обученной модели и сохраняем предсказанные значения.

Теперь вызываем функцию оценки качества и смотрим на оценки и график.

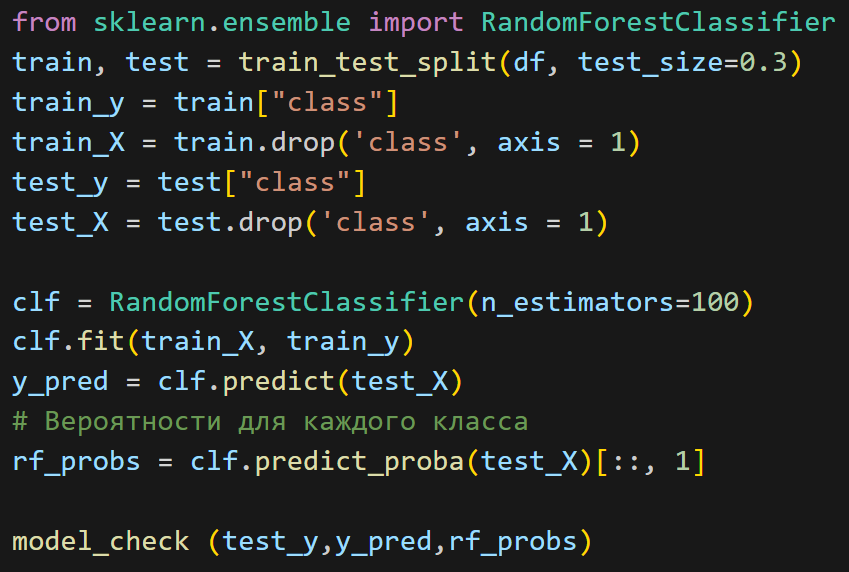


Рисунок 21 – скриншот кода

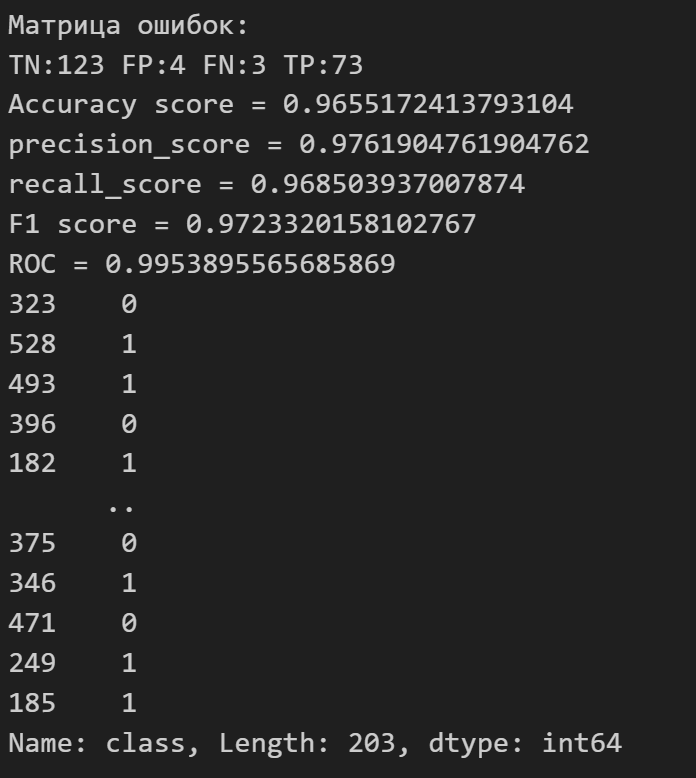


Рисунок 22 – результат вывода

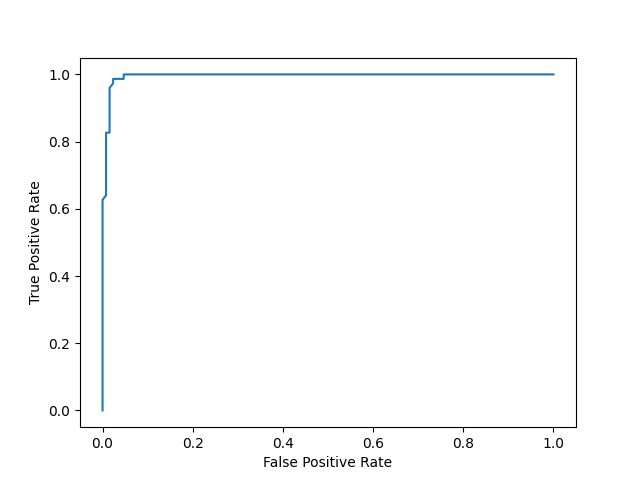


Рисунок 23 – построенный график

*Вывод:* Таким образом, в ходе выполнения лабораторной работы был выбран и описан выбранный датасет про пациентов с болезнью сердца, изучен интерфейс и возможности Jupyter Notebook, изучены базовые функции библиотеки Pandas и разработана программа, которая считывает данные, выводит о них информацию, удаляет дубликаты, пропуски, изменяет тип данных. Также были изучены методы классификации, стандартизированы данные, обучены различные модели, вычеслены метрики качества, получена матрица неточностей и построен график ROC-кривой.