УДК 004.85

Анализ данных по болезни сердца

Черкашин Александр Михайлович
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
Целью является написание скрипта по предсказание будущие появление хронических болезней сердца в течении следующих 10 лет. В работе использовано библиотека Scikit-learn и составления схемы работы в Orange с данными для создания и обучение модели и предсказание хронических болезней сердца. В результате обученная модель будет предсказывать хронических болезней сердца в течении следующих 10 лет.
Ключевые слова: Python, Scikit-learn, Orange-canvas, Логистическая регрессия.

Analysis of heart disease data

Cherkashin Alexander Mihailovich
Sholom-Aleichem Priamursky State University
Student

Abstract
The goal is to write a script for the prediction of the future appearance of chronic heart disease during the next 10 years. The work uses the Scikit-learn library and the preparation of the work scheme (Orange) with data to create and teach the model and the prediction of chronic heart disease. As a result, the trained model will predict chronic heart disease during the next 10 years.
Keywords: Python, Scikit-learn, Orange-canvas, Logistic regression.

1 Введение

1.1 Актуальность исследования

Данная статья описывает возможность написание скрипта для обучения моделей данные о болезни сердца.

1.2 Цель исследования

Целью исследования является написание скрипта на языке Python по обучение моделей и предсказание будущие появление хронических болезней сердца в течении следующих 10 лет.

1.3 Обзор исследований

М. В. Бобырь, Д. В. Титов, С. А. Кулабухов рассматривают подход принятия решений, основанный на использовании мягких арифметических операций в структуре нечеткой системы вывода [1]. А. Н. Тырсин и Е. В Васильева представили бинарную логистическую регрессию для решения

экономических задачах [2]. В. Г. Манжула, Д. С. Федяшов показал актуальность использования нейронных сетей в интеллектуальном анализе данных достоинства и недостатки данных сетей [3]. С. Г. Григорьев, Ю. В. Лобзин, Н. В. Скрипченко рассматривают модели логистической регрессии для прогноза вероятности наступления интересующего исследователя события при наличии двух возможных вариантов исхода диагноза [4]. Н. П. Васильев, А. А. Егоров показали расчет параметров логистической регрессии в качестве статистической модели для оценки зимостойкости растений, которой проблемой неустойчивого счета, при построения логистической регрессии сводится к нелинейной системе уравнений, для решения которой использовался метод Ньютона-Рафсона [5]. П. А. Попова, А. Н. Ротмистров применили логистическую регрессию для аспекта выявления детерминанта политического активизма [6].

2. Результаты и обсуждение

2.1 Исходные данные

Исходные данные взяты из источника автора курса https://stepik.org/course/73952 (дата обращения 2021-11-16) в каталоге https://stepik.org/lesson/414085/step/1 (дата обращения 2021-11-18) в ссылке на презентации, слайд 67 (Фрамингемское исследование сердца) в слайде 73 указано ссылка на скачивания данные.

Таблица 1. Столбцы данные.

#	Имя	Тип	Роль	Значения	Описания
1	male	Категориальный	Feature	0, 1	пол пациента 1-м, 0-ж
2	age	Числовой	Feature		возраст на момент начала участия в исследовании
3	education	Числовой	Feature		уровень образования - 1,2,3,4
4	currentSmoker	Категориальный	Feature	0, 1	курение
5	cigsPerDay	Числовой	Feature		курение
6	BPMeds	Категориальный	Feature	0, 1	прием лекарств от давлени
7	prevalentStroke	Категориальный	Feature	0, 1	история инсульта
8	prevalentHyp	Категориальный	Feature	0, 1	страдает высоким давлением
9	diabetes	Категориальный	Feature	0, 1	страдает диабетом
10	totChol	Числовой	Feature		
11	sysBP	Числовой	Feature		

12	diaBP	Числовой	Feature		
13	BMI	Числовой	Feature		индекс массы тела
14	heartRate	Числовой	Feature		пульс
15	glucose	Числовой	Feature		
16	TenYearCHD	Категориальный	Target	0, 1	категориальная переменная, появление хронических болезней сердца в течении следующих 10 лет (0\1)

Атрибуты (Роль):
Meta — Мета, не используется для обучение нейронный сети.
Target — Цель, значение который что предсказать.
Feature — Исходные данные, указывает что обучать.
Skip — Пропускать, не используется в определении столбец таблицы.
Указанный в таблице 1 и рис 2.1.

Для модели искусственного интеллекта значение TenYearCHD определяет данные (атрибут Target) для проверяющего, ответ который должно получиться, все остальные столбцы (атрибут Feature) используется данные для обучение нейронный сеть (рис 2.1) (табл. 1). Данная модель используется для обучения с учителем, используется логистическая регрессия. Применяется приложение Orange-canvas-core [7] http://orange.biolab.si/ (https://orangedatamining.com/) для составления схемы работы с данными (рис 2.2).

Исходный код 2.1 написан на языке Python, использует библиотеки: matplotlib [8] для визуализация данных (построение графики) [9], scikit-learn (sklearn) для машинного обучения [10], Pandas для обработки и анализа данных [11], Seaborn для работы визуализация данных на высоком уровнем интерфейс основанным на Matplotlib [12].

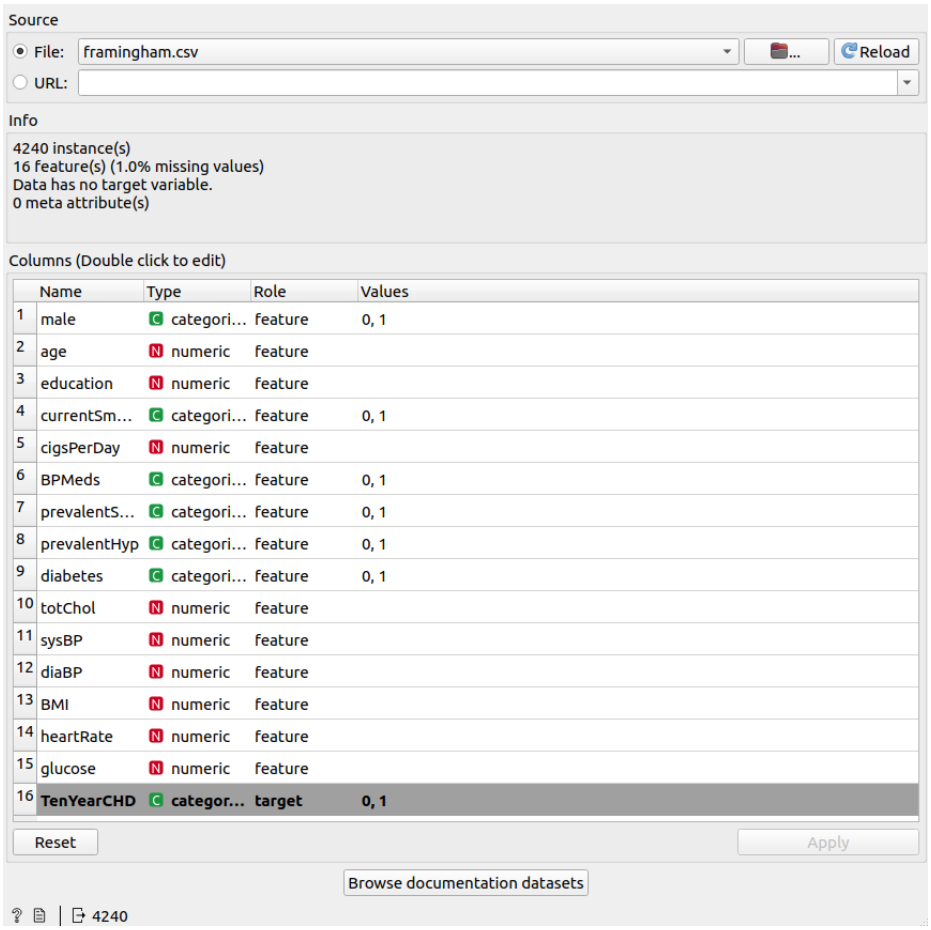

Рисунок 2.1. Загрузка файла (File) framingham.csv (Orange-canvas)

В Orange загружаем данные (рис 2.1), в строке 11 (листинг 2.1) импортируем данные используем библиотеку Pandas.

В строке 12 (листинг 2.1). Удаляем пропущенные значения (NaN) в Orange-canvas не удаляли пропущенные данные.

В строке 13 — 19 (листинг 2.1) устанавливаем тип Category (таблица 1) (рис 2.1).

В строке 21 - 24 (листинг 2.1) выделяем столбцы переменная df_train (роль Feature), df_target (роль Target) (таблица 1) (рис 2.1).

В Data Table — отображает таблицу (рис 2.2).

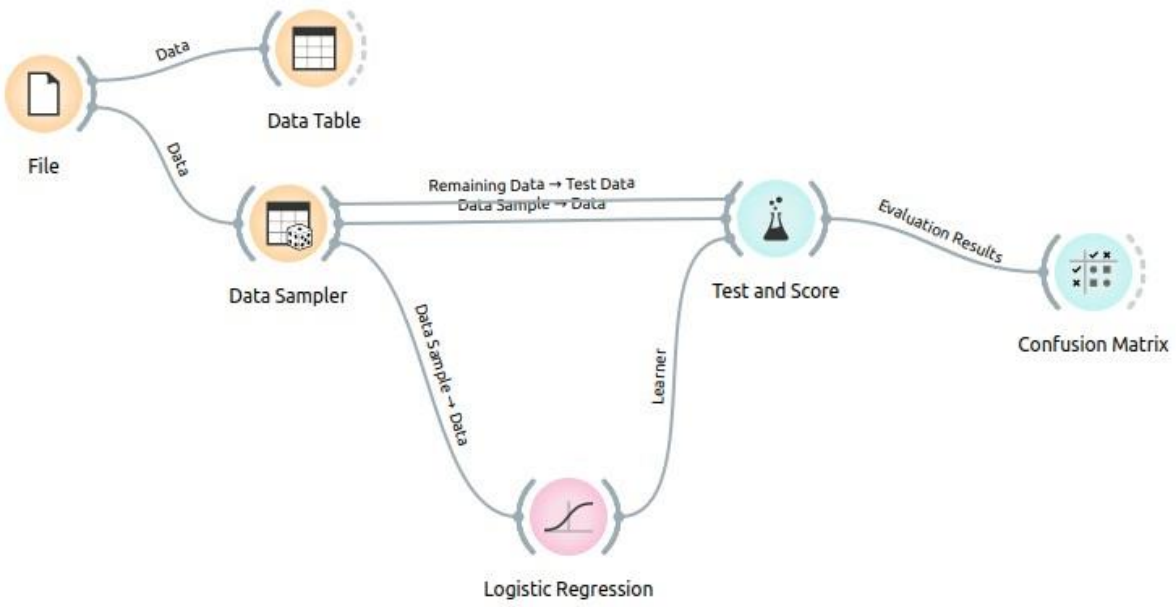

Рисунок 2.2. Схема построение с использование Orange-canvas

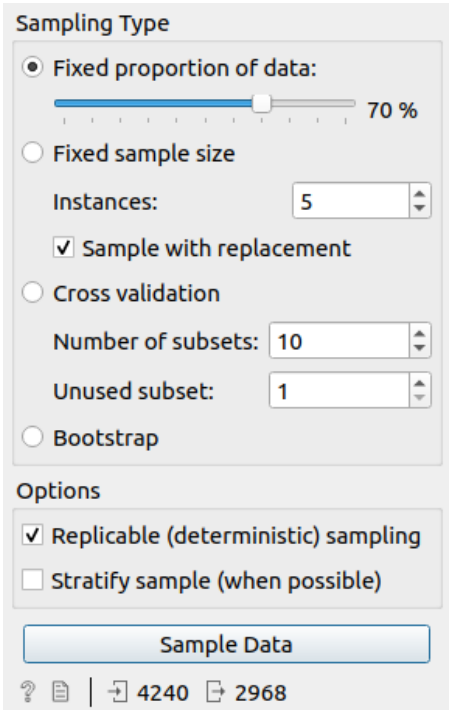

Рисунок 2.3. Окно Data Sampler Orange-canvas

Data Sampler использует для разбиения на 70% данные на две части (Рис 2.3), в строке 26 использует функция train_test_split, аргумента test_size — это процент разбиение данные (в данном случае заданно 0.7) и random_state — зерно случайно значение, и возвращает четыре перемены x_train и y_train — это цель обучение модель, x_test и y_test — это цель тестирования модель (листинг 2.1).

Remaining Data — Выбранные данные в 30% (Рис 2.3) соединен в Test Data (Test and Score) (рис 2.4).

Data Sample — Данные в 70% (Рис 2.3) соединен в Data (Test and Score) (рис 2.4).

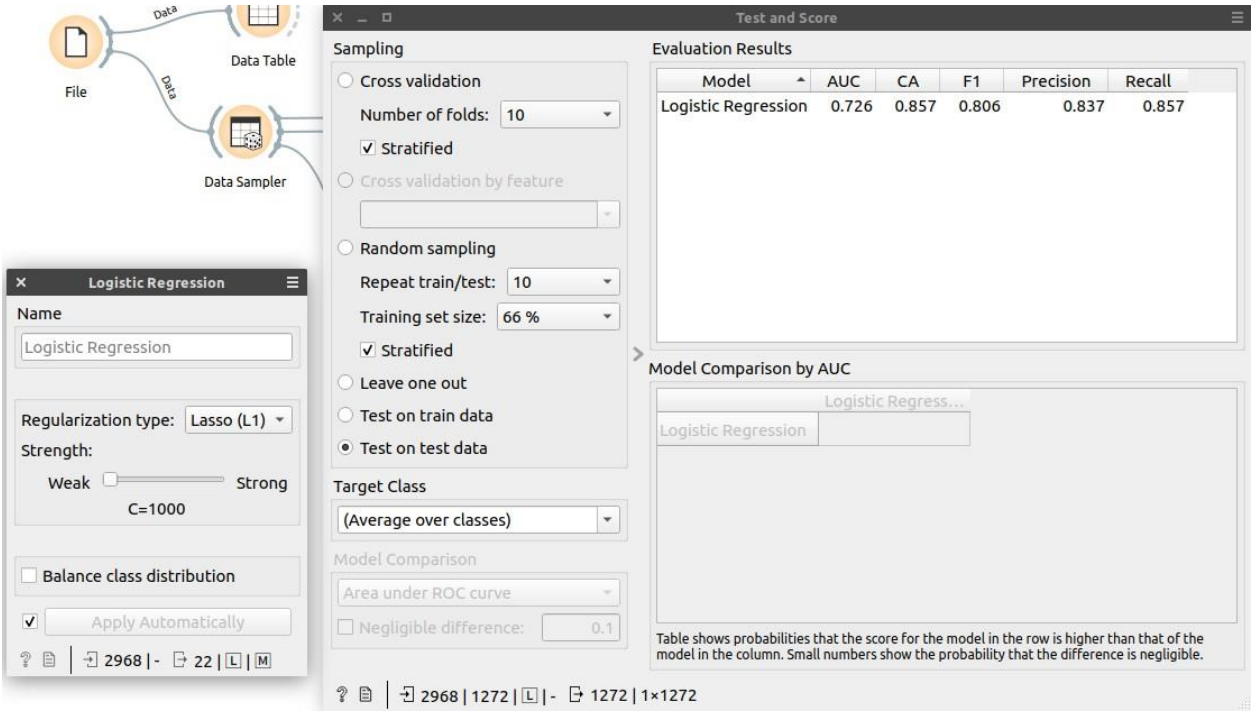

Рисунок 2.4. Test and Score результат (Orange-canvas)

В строке 28 и 29 (листинг 2.1) обучаем модель используем логистическая регрессия. Класс LogisticRegression реализует логистическую регрессию, Аргумент penalty — указываем норму штрафа для модели (указано L1), solver — решатель (указан liblinear) [13]. Fit — Обучает модель.

Строка 30 (листинг 2.1) Вводим модель, для получения результата насколько модель подстроилась к исходным данным для получения ответа.

Строка 32 - 36 (листинг 2.1) Метрика AUC, CA, F1, Precision, Recall — оценка качество модели (рис 2.4).

Метрика AUC — оценка качество бинарной классификации, отображает соотношение между долей объектов от общего количество носителей признака [14].

Метрика CA — оценка точности предсказания образца [15].

Метрика F1 — оценка способности классификатора находить все положительные образцы. Значение 0 — худшее, 1 — лучшее [16]

Метрика Precision — оценка точности способности классификатора не отмечать как положительный образец как отрицательный образец [17].

Метрика Recall — Оценка способности классификатора найти все положительные образцы [18].

В строке 38 (листинг 2.1) выводится оценка ошибок первого и второго рода.

В строке 39 - 42 (листинг 2.1) Построение визуальную матрицу 4x4 с использования Seaborn, представляющий собой оценка ошибка первого и второго рода и выводит рисунок (рис 2.4).

Листинг 2.1 Исходный код

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-

import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
from sklearn.model_selection import train_test_split
import pandas as pd
import seaborn as sn

df = pd.read_csv("framingham.csv")
df = df.dropna(how='any',axis=0)
df["male"] = df["male"].astype("category")
df["currentSmoker"] = df["currentSmoker"].astype("category")
df["BPMeds"] = df["BPMeds"].astype("category")
df["prevalentStroke"] = df["prevalentStroke"].astype("category")
df["prevalentHyp"] = df["prevalentHyp"].astype("category")
df["diabetes"] = df["diabetes"].astype("category")
df["TenYearCHD"] = df["TenYearCHD"].astype("category")

df_train = df[['male', 'age', 'education', 'currentSmoker', 'cigsPerDay', 'BPMeds',
      'prevalentStroke', 'prevalentHyp', 'diabetes', 'totChol', 'sysBP',
      'diaBP', 'BMI', 'heartRate', 'glucose']]
df_target = pd.DataFrame({"TenYearCHD": df["TenYearCHD"]})

x_train, x_test, y_train, y_test = train_test_split(df_train, df_target, test_size=0.7, random_state=1)

logreg = LogisticRegression(penalty="l1", C=1000, solver="liblinear")
logreg.fit(x_train, y_train.values.ravel())
x_test_pred = logreg.predict(x_test)

print("AUC: %.3f" % metrics.roc_auc_score(y_test, x_test_pred, average='weighted'))
print("CA: %.3f" % metrics.accuracy_score(y_test, x_test_pred))
print("F1: %.3f" % metrics.f1_score(y_test, x_test_pred, average='weighted'))
print("Precision: %.3f" % metrics.precision_score(y_test, x_test_pred, average='weighted'))
print("Recall: %.3f" % metrics.recall_score(y_test, x_test_pred, average='weighted'))

cnf_matrix = metrics.confusion_matrix(y_test, x_test_pred)
plt.figure(figsize = (5,5))
sn.set(font_scale=1.2)
sn.heatmap(cnf_matrix, annot=True, annot_kws={"size": 16}, cmap='Greens', fmt='')
plt.show()
```

В результате выполнение программы вывод

```
AUC: 0.547
CA: 0.854
F1: 0.808
Precision: 0.822
Recall: 0.854
```

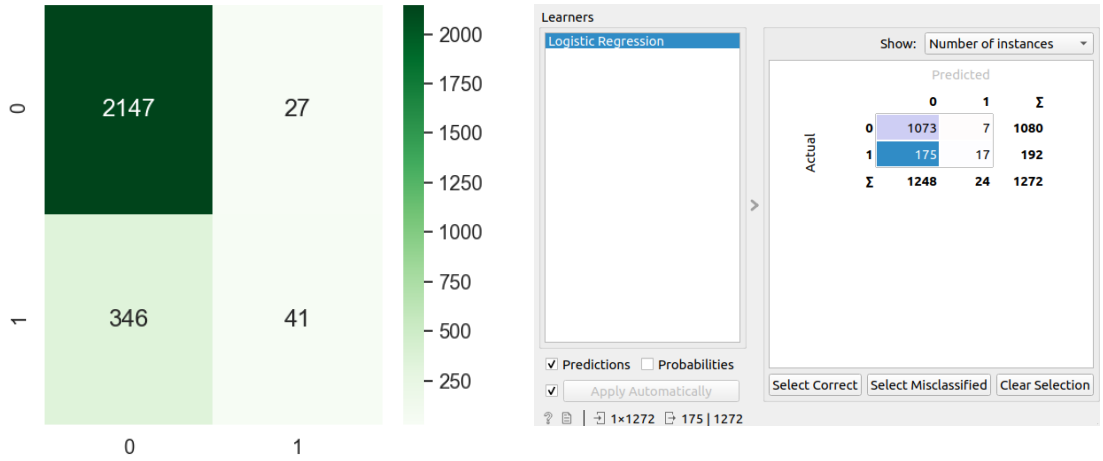

Рисунок 2.5. Вывод программы (листинг 2.1) Confusion matrix (слева), вывод программы Orange-canvas Confusion matrix (с право).

В 1, 0 — 346 - Ошибка первого рода — ложно положительный диагноз заболевания сердце то есть заболевания нет но в диагноз оно положительно (рис 2.5 слева) [19].

В 0, 1 — 27 — Ошибка второго рода — ложно отрицательный диагноз заболевания сердце то есть заболевания есть но в диагноз оно отрицательный (рис 2.5 слева) [19].

В 0, 0 — 2147 - Истинно положительный диагноз, то есть заболевания есть диагноз положительно (рис 2.5 слева) [19].

В 1, 1 — 41 - Истинно отрицательный диагноз, то есть заболевания нет диагноз отрицательный (рис 2.5 слева) [19].

3 Выводы

В данном статье был написан скрипт по обучение моделей и предсказание будущие появление хронических болезней сердца в течении следующих 10 лет по модели квалификации и оценка ошибка первого и второго рода. Использовались два способа работы с программы. 1. Способ через Orange-canvas. 2. Способ через написание на языке Python. В результате получаем график Confusion matrix который оценивает ошибкки первого и второго рода, также использовались различные метрики оценки качество модели, такие как AUC, CA, F1, Precision, Recall.

Библиографический список

1. Бобырь М. В., Титов Д. В., Кулабухов С. А. О некоторых свойствах мягкого алгоритма нечетко-логического вывода //Известия Юго-Западного государственного университета. 2015. Т. 59. №. 2. С. 39.
2. Тырсин А. Н., Васильева Е. В. Бинарная логистическая регрессия как модель управления на примере задачи повышения качества жизни населения //Фундаментальные исследования. – 2020. №. 10. С. 96-102.
3. Манжула В. Г., Федяшов Д. С. Нейронные сети Кохонена и нечеткие

нейронные сети в интеллектуальном анализе данных //Фундаментальные исследования. 2011. №. 4.
4. Григорьев С. Г., Лобзин Ю. В., Скрипченко Н. В. Роль и место логистической регрессии и ROC-анализа в решении медицинских диагностических задач //Журнал инфектологии. 2016. Т. 8. №. 4. С. 36-45.
5. Васильев Н. П., Егоров А. А. Опыт расчета параметров логистической регрессии методом Ньютона–Рафсона для оценки зимостойкости растений //Математическая биология и биоинформатика. 2011. Т. 6. №. 2. С. 190-199.
6. Попова П. А., Ротмистров А. Н. Логистическая регрессия с категориальными предикторами и эффектами взаимодействия и CHAID: сравнительный анализ на эмпирическом примере //Социология: методология, методы, математическое моделирование (Социология: 4М). 2016. №. 43. С. 63-99.
7. orange-canvas-core · PyPI // PyPI URL: https://pypi.org/project/orange-canvas-core/ (дата обращения: 2022-01-15).
8. matplotlib · PyPI // PyPI URL: https://pypi.org/project/matplotlib/ (дата обращения: 2022-01-15).
9. Matplotlib — Visualization with Python // matplotlib URL: https://matplotlib.org (дата обращения: 2022-01-15).
10. scikit-learn: machine learning in Python — scikit-learn 1.0.2 documentation // scikit-learn URL: https://scikit-learn.org/ (дата обращения: 2022-01-15).
11. pandas - Python Data Analysis Library // pandas URL: https://pandas.pydata.org/ (дата обращения: 2022-01-15).
12. seaborn: statistical data visualization — seaborn 0.11.2 documentation // seaborn URL: https://seaborn.pydata.org/ (дата обращения: 2022-01-20).
13. sklearn.linear_model.LogisticRegression — scikit-learn 1.0.2 documentation // scikit-learn URL: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html (дата обращения: 2022-01-20).
14. sklearn.metrics.roc_auc_score — scikit-learn 1.0.2 documentation // scikit-learn URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_auc_score.html (дата обращения: 2022-01-20).
15. sklearn.metrics.accuracy_score — scikit-learn 1.0.2 documentation // scikit-learn URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html (дата обращения: 2022-01-20).
16. sklearn.metrics.f1_score — scikit-learn 1.0.2 documentation // scikit-learn URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html (дата обращения: 2022-01-20).
17. sklearn.metrics.precision_score — scikit-learn 1.0.2 documentation // scikit-learn URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_score.html (дата

обращения: 2022-01-20).
18. sklearn.metrics.recall_score — scikit-learn 1.0.2 documentation // scikit-learn URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html (дата обращения: 2022-01-20).
19. Ошибки первого и второго рода — Википедия // Википедия URL: https://ru.wikipedia.org/wiki/Ошибки_первого_и_второго_рода (дата обращения: 2022-01-21).

Приложения

Рисунок 3.1. Исходный код (файл framingham_data.png)

Листинг 3.1. SHA 256:

```
00f6d2f43c2467148cc700bb8a2263329a7c395de44119581ebb7ceddb9ae8bb
```

Команда convert входит в состав программы ImageMagick (https://imagemagick.org) для обработки изображения. Используем для преобразования изображения (рис 2.6) в архив.

В Листинг 3.2. использовался ОС Ubuntu.

В Листинг 3.3. использовался ОС Windows 10.

Файл (рис 3.1) содержится 1 бит глубина цвета и 3 компонента (RGB) размер изображения 384×362.

Процедура преобразования:

1. В файл изображения (рис 3.1) преобразовать в архив
2. Удалить лишний 4 байта.
3. Проверить целостность (листинг 3.1).
4. Распаковать файл архив.

Листинг 3.2. Под Unix

```
1 convert -depth 1 framingham_data.png rgb:framingham_data.cpio.xz
2 truncate -s -4 framingham_data.cpio.xz
3 shasum -a 256 framingham_data.cpio.xz
4 xz -cd framingham_data.cpio.xz | cpio -id
```

Листинг 3.1, строка 1. используется программа ImageMagick для чтения изображения файл (рис 3.1) формата png и преобразования в файл.

Листинг 3.1, строка 2. используется утилита для изменения размера файла, входящий пакет GNU core utilities (https://www.gnu.org/software/coreutils/) вычитается 4 байта.

Листинг 3.1, строка 3. используется утилита для проверки целостность файла (SHA 256) хеш должно совпадать в листинг 3.1.

Листинг 3.1, строка 4. используется утилита (пакет xz-utils https://tukaani.org/xz/) и cpio (https://www.gnu.org/software/cpio/manual/cpio.html) для распаковки файла framingham_data.cpio.xz.

Листинг 3.3. Под Windows

```
1 convert -depth 1 framingham_data.png rgb:framingham_data.cpio.xz
2 wmic datafile where Name='<Полный путь>\\framingham_data.cpio.xz' get Size
3 FSUTIL file seteof framingham_data.cpio.xz <Размер файла>
4 certUtil -hashfile framingham_data.cpio.xz SHA256
```

В Листинг 3.3, строка 1. используется программа ImageMagick для чтения изображения формата png и преобразования в файл (рис 3.1).

В Листинг 3.3, строка 2. Используется утилита для получения размер файла. Указывается полный путь, путь должен быть \\ а не \.

В Листинг 3.3, строка 3. Используется утилита для изменения размер файла в данном случае указывается текущий размер файла и вычитать 4 байта (<размер файла> - 4).

В Листинг 3.3, строка 4. Используется утилита, используется для проверки целостность файла (SHA 256) хеш должно совпадать в листинг 3.1. Файл framingham_data.cpio.xz распаковывается любым архиватором используя например 7-zip (https://www.7-zip.org/) или (https://tukaani.org/xz/).