Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«Сибирский федеральный университет»

**ИТОГОВАЯ АТТЕСТАЦИОННАЯ РАБОТА**   
по программе профессиональной переподготовки   
«Аналитика данных»

на тему: «Распознавание сердечно-сосудистых заболеваний у пациентов»

|  |  |
| --- | --- |
| Слушатель: Гопиенко А.Д. |  |
| Группа: КИ22-17/2Б |  |
| Руководитель итоговой аттестационной работы: Пересунько П.В.,  старший преподаватель  кафедры Программной инженерии  ИКИТ СФУ |  |

Красноярск

2024

СОДЕРЖАНИЕ

[Введение 3](#_Toc168861896)

[1 Анализ предметной области 5](#_Toc168861897)

[1.1 Постановка задачи 5](#_Toc168861898)

[1.2 Оценка качества модели 5](#_Toc168861899)

[1.3 Выбор средств разработки 9](#_Toc168861900)

[2 Проектирование приложения 11](#_Toc168861901)

[**2.1 Описание данных** 11](#_Toc168861902)

[**2.2 Описание функционала приложения** 12](#_Toc168861903)

[2.3 Требования к программе или программному изделию 12](#_Toc168861904)

[2.4 Стадии и этапы разработки 12](#_Toc168861905)

[**3 Реализация приложения** 14](#_Toc168861906)

[3.1 Предварительная обработка данных 14](#_Toc168861907)

[3.2 Разработка и обучение моделей 18](#_Toc168861908)

[Заключение 23](#_Toc168861909)

[Список использованных источников 24](#_Toc168861910)

# Введение

В настоящее время сердечно-сосудистые заболевания (ССЗ) являются основной причиной смертности и инвалидизации населения по всему миру. В связи с этим возникает необходимость развития эффективных методов распознавания и диагностики ССЗ, способных удовлетворить потребности системы здравоохранения. Одним из таких методов является использование машинного обучения для анализа медицинских данных и распознавания заболевания у пациента.

Основной задачей диагностики ССЗ является раннее выявление заболевания у пациента. Осуществление диагностики проводится с помощью медицинских данных пациента. Использование машинного обучения позволяет автоматизировать процесс анализа данных и повысить точность диагностики.

Основы применения машинного и глубокого обучения в медицине сформировались в последние десятилетия. В отличие от классического программирования алгоритмы машинного и глубокого обучения базируются на математическом и статическом моделировании и способны эффективно обрабатывать нелинейные зависимости, а также учитывать различные факторы, влияющие на результаты диагностики.

Целью данной работы является автоматизация процессов распознавания сердечно-сосудистых заболеваний у пациентов.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Изучить предметную область.

2. Провести анализ существующих и потенциальных методов предсказания сердечно-сосудистых заболеваний у пациентов с помощью методов машинного обучения и искусственного интеллекта.

3. Провести обучение и сравнение выбранных методов машинного обучения.

# **1 Анализ предметной области**

## 1.1 Постановка задачи

Распознавание ССЗ у пациентов можно рассматривать как задачу бинарной классификации. Постановка задачи классификации выглядит следующим образом. Пусть имеется множество размерности *n* объектов , каждый из которых характеризуется вектором признаков размерности *m* . Для каждого исследуемого объекта известна его принадлежность к тому или иному классу . Задача классификации состоит в следующем. Требуется сформулировать правило (алгоритм) , которое позволит по заданным значениям признаков назначить ранее неизвестному объекту тот или иной класс.

Применительно к проблеме распознавания ССЗ у пациентов, задача классификации сводится к разделению пациентов на две категории: с наличием заболевания и без заболевания. На основе выборки исходных данных требуется построить алгоритм, способный для заданных значений параметров здоровья пациента выдать ответ о наличии или отсутствии у него сердечно-сосудистого заболевания с достаточно высокой точностью.

Точность решения задачи зависит от объема и качества исходных данных, выбранных методов классификации, критериев качества прогнозирования, значимости контролируемых параметров состояния здоровья пациента.

## Оценка качества модели

В качестве метрик для задач классификации обычно используются следующие показатели:

* точность (ассuracy) – отношение всех правильно классифицированных объектов к общему количеству всех классифицированных объектов:

, (1)

где – истинно-положительные исходы (объекты, классифицированные как «положительные» и которые действительно являются «положительными», т.е. относятся к классу ), – истинно-отрицательные исходы (объекты, классифицированные как «отрицательные» и которые действительно являются «отрицательными», т.е. относятся к классу ), – ложно-положительные исходы (объекты, классифицированные как «положительные», но которые фактически являются «отрицательными», т.е. относятся к классу ), – ложно-отрицательные исходы (объекты, классифицированные как «отрицательные», но которые фактически являются «положительными», т.е. относятся к классу .

В случае дисбаланса классов обычная точность заменяется на сбалансированную:

, (2)

* точность (precision) – отношение всех объектов, классифицированных как «положительные» и которые действительно являются «положительными» к общему количеству объектов, классифицированных как «положительные»:

. (3)

Мера точности характеризует способность прогнозной модели правильно находить истинные «положительные» объекты с учетом и при этом не допускать ложных «положительных» объектов.

* полнота (recall) – отношение всех объектов, классифицированных как «положительные» и которые действительно являются «положительными» к общему количеству действительно «положительных» объектов:

. (4)

Мера полноты характеризует способность модели находить настоящие «положительные» объекты среди всего множества объектов для классификации, без учета ложных «положительных».

* специфичность (specificity) – отношение всех объектов, классифицированных как «отрицательные» и которые действительно являются «отрицательные» к общему количеству действительно «отрицательных» объектов:

. (5)

Мера специфичности характеризует способность модели находить настоящие «отрицательные» объекты среди всего множества объектов для классификации, без учета ложных «отрицательных».

* F-мера (F1 score) – гармоническое средние между точностью и полнотой:

. (6)

F-мера демонстрирует, как много случаев классифицируется моделью правильно, и сколько истинных экземпляров модель сможет правильно классифицировать.

Для многоклассовых задач стараются свести подсчет качества к вычислению одной из рассмотренных выше двухклассовых метрик. Выделяют два подхода к такому сведению: микро- и макро-усреднение.

Пусть выборка состоит из классов. Таким образом, имеется двухклассовых задач, каждая из которых заключается в отделении своего класса от остальных, то есть целевые значения для -й задачи вычисляются как . Для каждой из них можно вычислить различные характеристики алгоритма ().

При микро-усреднении сначала эти характеристики усредняются по всем классам, а затем вычисляется итоговая двухклассовая метрика

, (7)

. (8)

При макро-усреднении сначала вычисляется итоговая метрика для каждого класса, а затем результаты усредняются по всем классам:

, (9)

где ,

, (10)

где .

## 1.3 Выбор средств разработки

Выбор средств разработки осуществляется на первых стадиях работы над проектом, после определения требований к создаваемому программному продукту.

Существующие на сегодняшний день языки программирования можно выделить в следующие группы:

* универсальные языки высокого уровня;
* специализированные языки разработчика ПО;
* специализированные языки пользователя;
* языки низкого уровня.

Для разработки приложения было решено рассматривать универсальные высокоуровневые языки общего назначения. В данном проекте выбор пал на язык Python.

Python – высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ.

Python является мультипарадигмальным языком программирования, поддерживающим императивное, процедурное, структурное, объектно-ориентированное программирование, мета-программирование и функциональное программирование. Задачи обобщённого программирования решаются за счёт динамической типизации. Основные архитектурные черты – динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных вычислений с глобальной блокировкой интерпретатора (GIL), высокоуровневые структуры данных. Поддерживается разбиение программ на модули, которые, в свою очередь, могут объединяться в пакеты.

Кроме этого, для решения задач машинного обучения на языке Python существует большое количество библиотек.

# 2 Проектирование приложения

**2.1 Описание данных**

В качестве исходных данных были использованы данные о пациентах в формате .csv.

Данные о пациентах были взяты с соревнования на Kaggle и представлена на Рисунке 1.

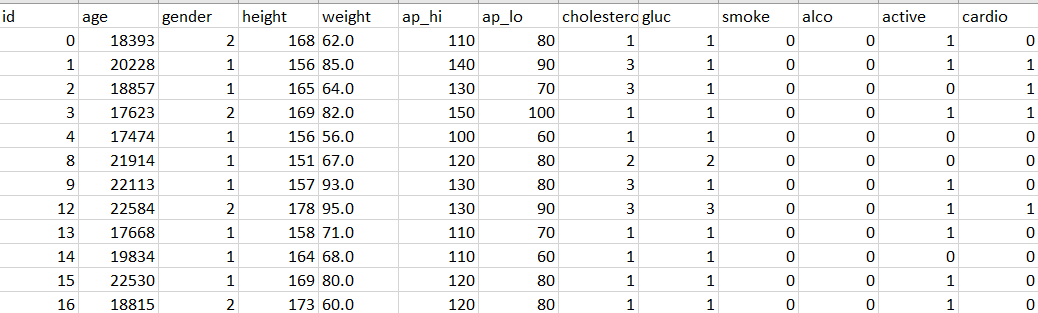


Рисунок 1 – Данные о пациентах

Id – id пациента

Age – возраст в пациента в днях с момента рождения

Gender – пол пациента (1 – женщина, 2 – мужчина)

Height – рост пациента в сантиметрах

Weight – вес пациента в килограммах

Ap\_hi – систолическое кровяное давление

Ap\_lo – диастолистическое кровяное давление

Cholesteric – уровень холестерола у пациента (1 – нормальный, 2 – выше нормального, 3 – сильно выше нормального)

Gluc – уровень глюкозы у пациента (1 – нормальный, 2 – выше нормального, 3 – сильно выше нормального)

Smoke – курение (1 – пациент курит, 0 – пациент не курит)

Alco – употребление алкоголя (1 – пациент употребляет алкоголь, 0 – пациент не употребляет алкоголь)

Active – физическая активность пациента (1 – пациент ведёт активный образ жизни, 0 – пациент ведёт сидячий образ жизни)

Cardio – результат распознавания заболевания у пациента (1 – пациент имеет заболевание, 0 – пациент здоров)

**2.2 Описание функционала приложения**

Для пользователей приложение должно предоставлять следующие возможности:

1. Загрузка данных о пациенте – загрузка данных о пациентах в формате .csv. Функциональные требования: уведомление пользователя при загрузке некорректного файла или при ошибках обработки файла.
2. Чтение и просмотр данных в графическом виде из файлf. Функциональные требования: графическое представление всей информации, содержащейся в файле; уведомление пользователя при ошибках чтения файла.
3. Прогнозирование типа диагноза пациента по данным медицинского исследования.

## 2.3 Требования к программе или программному изделию

Точность прогноза модели должна составлять не менее 70%. Время определения результатов – менее 5 минут.

Сохранность данных системы должна обеспечиваться при всех аварийных ситуациях.

Вероятность безотказной работы системы должна составлять не менее 99.99%.

## 2.4 Стадии и этапы разработки

Разработка должна быть проведена в три стадии:

1) определение функциональных требований к приложению;

2) рабочее проектирование: анализ обучающей выборки, разработка

3) тестирование и передача в эксплуатацию.

**2.5 Архитектура приложения**

Архитектура приложения для прогнозирования литологии по данным геофизических исследований скважин может включать в себя следующие компоненты:

1. Система управления файлами – производит чтение данных (в формате xlsx).
2. Система обработки данных – система выполняет предобработку исходных данных (удаление пропусков, преобразование типов данных).
3. Система анализа данных – включает в себя обученную модель машинного обучения с настроенными параметрами, принимающую на вход xlsx файлы. С помощью панели анализа пользователь инициирует эти действия.
4. Система выгрузки данных – сохранение обработанных данных в формате .csv

**3 Реализация приложения**

## Предварительная обработка данных

Сперва необходимо привести информацию о датасете.

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 70000 entries, 0 to 69999

Data columns (total 13 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 id 70000 non-null int64

1 age 70000 non-null int64

2 gender 70000 non-null int64

3 height 70000 non-null int64

4 weight 70000 non-null float64

5 ap\_hi 70000 non-null int64

6 ap\_lo 70000 non-null int64

7 cholesterol 70000 non-null int64

8 gluc 70000 non-null int64

9 smoke 70000 non-null int64

10 alco 70000 non-null int64

11 active 70000 non-null int64

12 cardio 70000 non-null int64

dtypes: float64(1), int64(12)

Приведём описательные статистики датасета

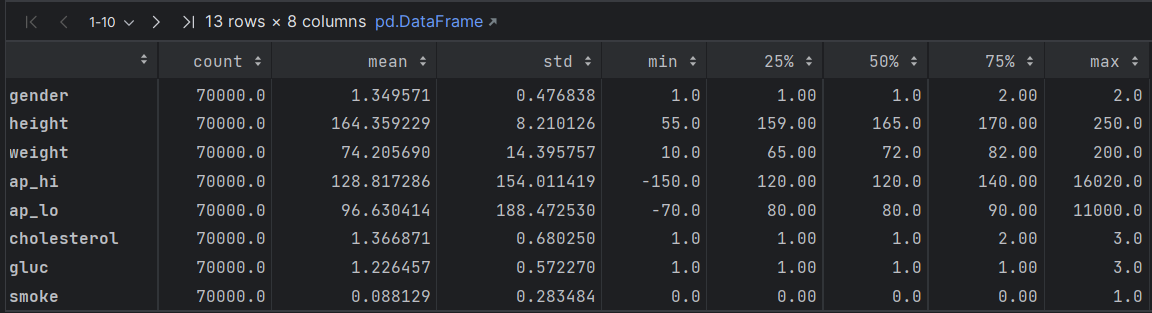


Рисунок 2 – описательные статистики датасета

Удаляем столбец id из датасета.

Следующим шагом производится восстановление пропусков. Данные содержат большое количество пропусков во входных параметрах (рисунок 2).

Код для построения тепловой карты распределения пропусков в данных:

missingno.matrix(dataset)

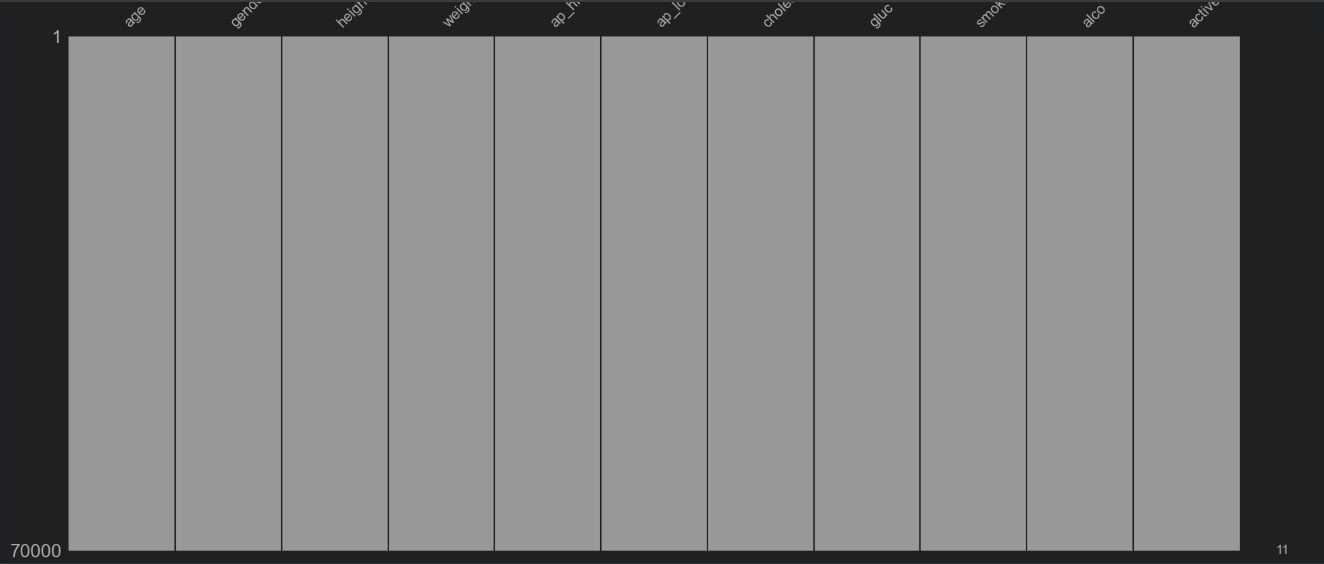


Рисунок 3 – Визуализация пропусков в данных

Визуализация показывает, что датасет не содержит пропуски.

Следующим шагом определим количество выбросов в датасете.

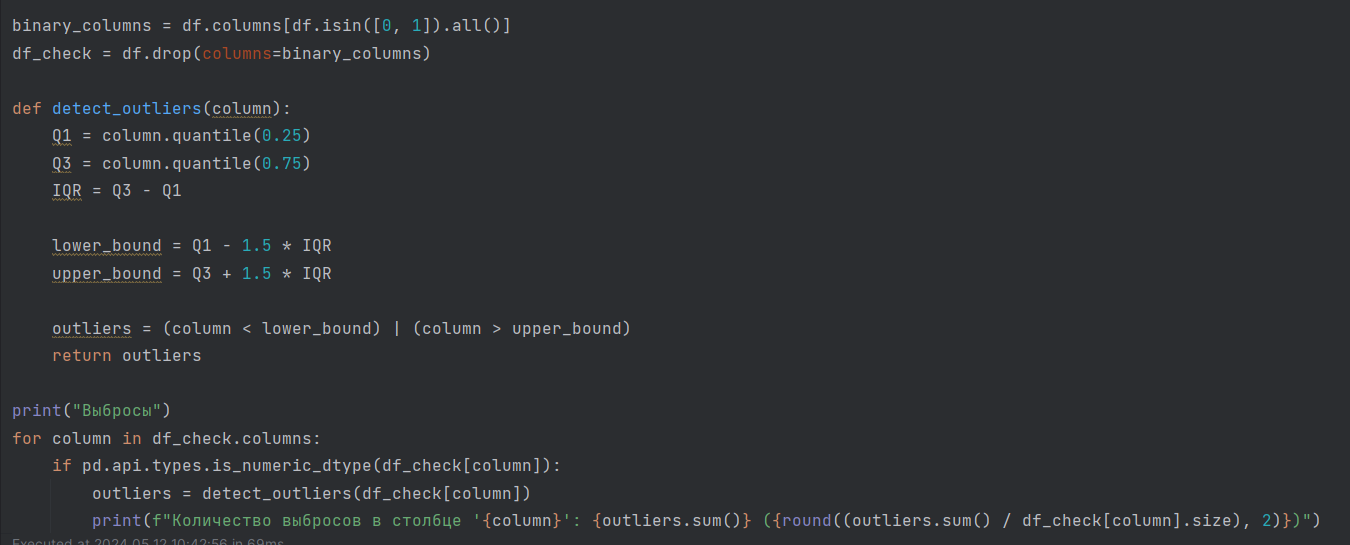


Рисунок 4 – определение выбросов в датасете

Выбросы

Количество выбросов в столбце 'age': 4 (0.0)

Количество выбросов в столбце 'gender': 0 (0.0)

Количество выбросов в столбце 'height': 519 (0.01)

Количество выбросов в столбце 'weight': 1819 (0.03)

Количество выбросов в столбце 'ap\_hi': 1435 (0.02)

Количество выбросов в столбце 'ap\_lo': 4632 (0.07)

Количество выбросов в столбце 'cholesterol': 0 (0.0)

Количество выбросов в столбце 'gluc': 10521 (0.15)

Датасет содержит выбросы. Избавимся от них с помощью метода межквартильного интервала (IQR).

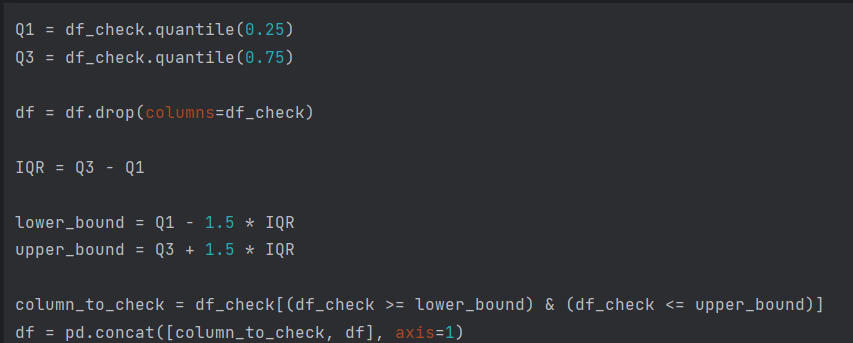


Рисунок 5 – применение метода межквартильного интервала

После удаления выбросов в датасете образовались пропуски. Избавимся от них с помощью метода ближайших соседей (KNN), предварительно выполнив нормализацию данных.

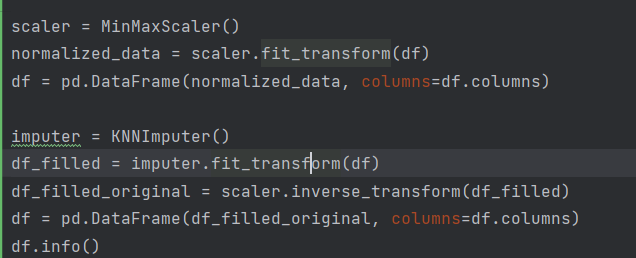


Рисунок 6 – устранение пропусков с помощью KNN

Следующим шагом выполним нормализацию и стандартизацию данных.

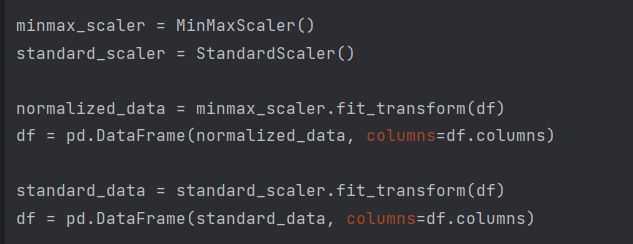


Рисунок 7 – нормализация и стандартизация данных

Проверим сбалансированность данных. При необходимости применим сэмплирование.

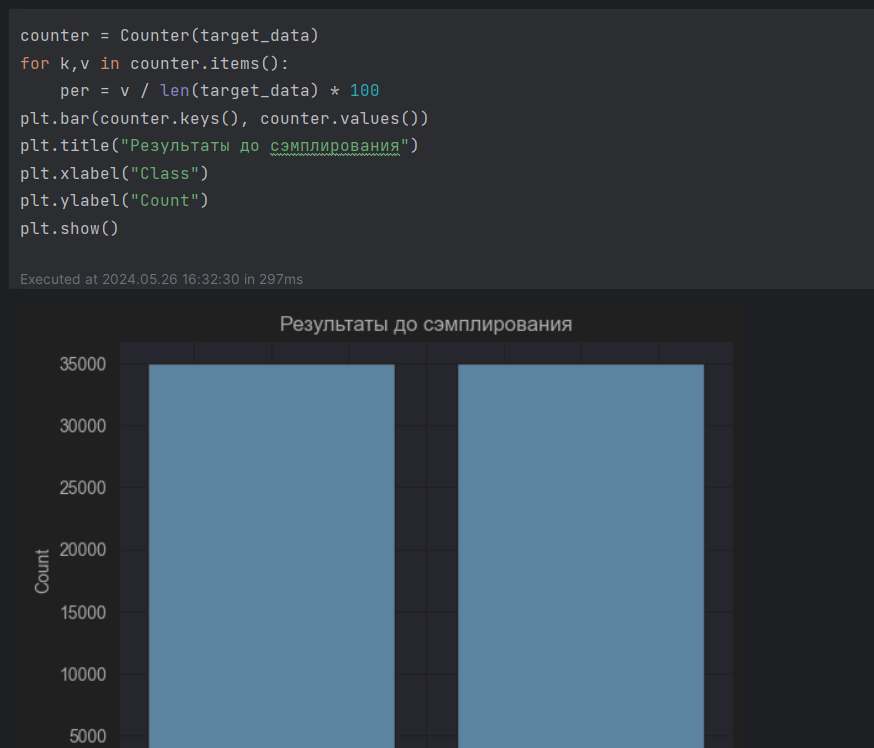


Рисунок 8 – проверка данных на сбалансированность

## 

## 3.2 Разработка и обучение моделей

Для решения поставленной задачи было использовано несколько готовых классификаторов:

1. BaggingClassifier
2. AdaBoost
3. CradientBoostingClassifier

Также были построены 3 модели нейронных сетей.

*BaggingClassifier*. Это мета-алгоритм композиционного обучения машин, предназначенный для улучшения стабильности и точности алгоритмов машинного обучения, используемых в статистической классификации и регрессии. Алгоритм также уменьшает дисперсию и помогает избежать переобучения. Хотя он обычно применяется к методам обучения машин на основе деревьев решений, его можно использовать с любым видом метода. Бэггинг является частным видом усреднения модели.

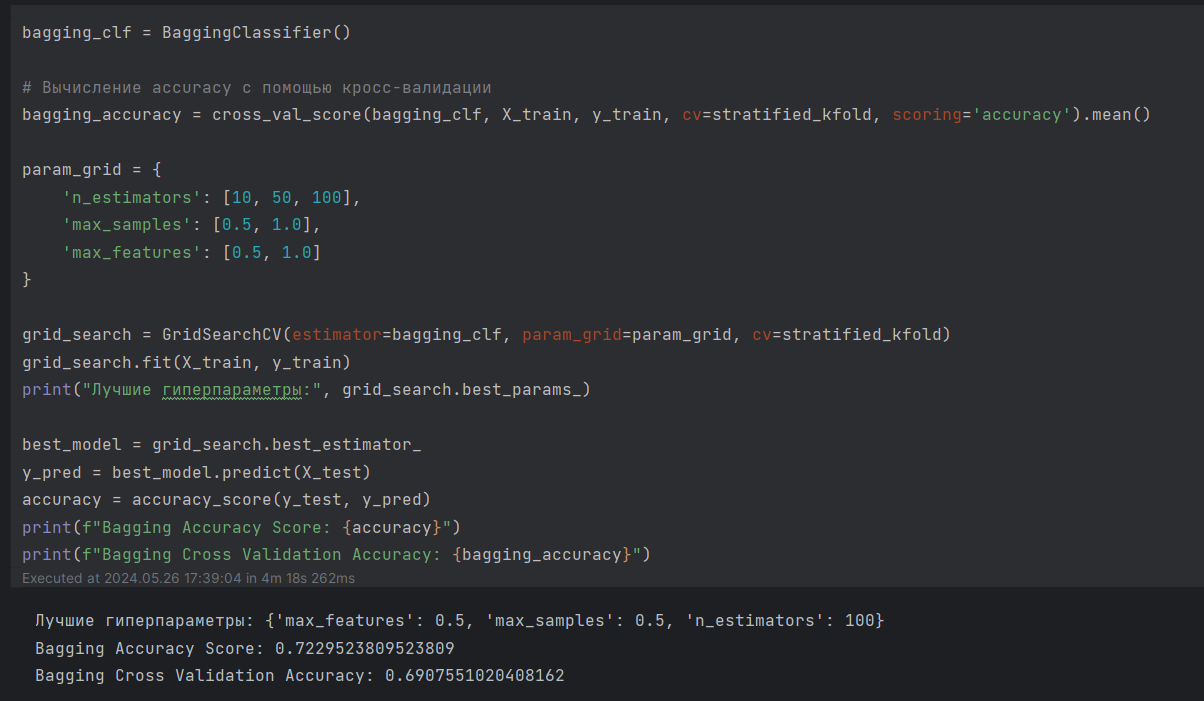


Рисунок 9 – BaggingClassifier

*AdaBoostClassifier*. Алгоритм машинного обучения, предложенный Йоавом Фройндом и Робертом Шапире. Этот алгоритм может использоваться в сочетании с несколькими алгоритмами классификации для улучшения их эффективности. Алгоритм усиливает классификаторы, объединяя их в «комитет». AdaBoost является адаптивным в том смысле, что каждый следующий комитет классификаторов строится по объектам, неверно классифицированным предыдущими комитетами. AdaBoost чувствителен к шуму в данных и выбросам. Однако он менее подвержен переобучению по сравнению с другими алгоритмами машинного обучения.

**

Рисунок 10 – AdaBoostClassifier

*GradientBoostClassifier*. Это техника машинного обучения для задач классификации и регрессии, которая строит модель предсказания в форме ансамбля слабых предсказывающих моделей, обычно деревьев решений.

**

Рисунок 11 – GradientBoostingClassifier

Neural Network – это техника машинного обучения для задач классификации и регрессии, которая строит модель предсказания на основе взаимосвязанных слоев нейронов, имитирующих работу биологических нейронов. Нейронная сеть применяется для бинарной классификации, эффективно решая задачи, где требуется разделение данных на две категории.

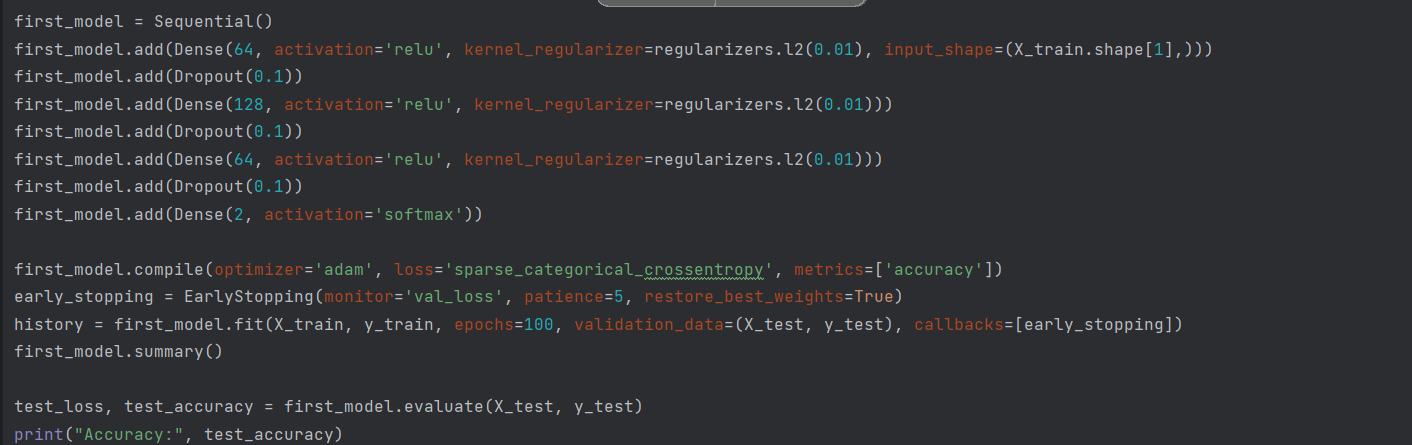


Рисунок 12 – первая модель нейронной сети



Рисунок 13 – вторая модель нейронной сети



Рисунок 14 – третья модель нейронной сети (функция build\_model)



Рисунок 15 – третья модель нейронной сети

Результаты тестирования обученных моделей представлены в таблице 1.

Таблица 1 – Результаты работы классификаторов

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Классификатор** | 1 | 2 | 3 | 4 | 5 | 6 |
| **Точность на тестовой выборке** | 72.295% | 72.671% | 73.190% | 72.995% | 51.700% | 73.371% |

На основании представленных результатов можно сказать, что лучше всего с задачей справляется третья модель нейронной сети.

# Заключение

В ходе выполнения итоговой работы было произведено изучение предметной области прогнозирования сердечно-сосудистых заболеваний, выполнен обзор подходов и существующих решений. На основании обзора было принято решение о целесообразности применения методов машинного обучения для решения этой задачи.

В процессе изучения данных была выполнена предварительная обработка данных, включающая в себя удаление выбросов, восстановление пропусков, нормализацию и стандартизацию данных.

Был произведен обзор основных методов машинного обучения и их особенностей, и отобраны методы для проверки. Для выбранных методов было произведено обучение на имеющихся данных медицинских обследований, произведено предсказание на тестовой выборке и оценена точность.

Было произведено сравнение методов машинного обучения на основании рассчитанных при оценке качества предсказания метрик. Наилучший результат, согласно полученным метрикам, оказался у модели *третьей модели нейронной сети*.

В ходе исследования был создан прототип программной системы. Для решения задачи был использован язык программирования Python.

# Список использованных источников

1. [Бурков](https://www.chitai-gorod.ru/books/authors/burkov_a/) А. Машинное обучение без лишних слов. [Библиотека программиста](https://www.chitai-gorod.ru/books/publishers/piter/biblioteka_programmista/). – М., 2020. – 192 с.

2. Грас Д. Data Science. Наука о данных с нуля: пер. с англ. – СПб.:  
 БХВ-Петербург, 2017. – 336 с.

3. Джоши П. Искусственный интеллект с примерами на Python. – М., 2019. – 448 с.

4. Маккинли У. Python и анализ данных. – Саратов: Профобразование, 2019. – 482 c.

5. Мирджалили В., Рашка С. Python и машинное обучение. Машинное и глубокое обучение с использованием Python, scikit-learn и TensorFlow. – М., 2020. – 848 с.

6. Мыльников Л.А. Статистические методы интеллектуального анализа данных. – СПб.: БХВ-Петербург, 2021 – 240 с.

7. Мюллер А., Гвидо С. Введение в машинное обучение с помощью Python. Руководство для специалистов по работе с данными. – М., 2017. – 480 с.

8. Плас Дж. В.Python для сложных задач: наука о данных и машинное обучение. – СПб.: Петербург, 2018 – 576 с.

9. Силен Д., Мейсман А., Али М. Основы Data Science и Big Data, Python и наука о данных. – М., 2017. – 336 с.

10. Скиена С. Наука о данных: учебный курс. Пер. с англ. – СПб.: ООО «Диалектика», 2020. – 544 с.

11. Флах П. Машинное обучение. Наука и искусство построения алгоритмов, которые извлекают знания из данных: пер. с англ. А.А. Слинкина. – М.: ДМК Пресс, 2015. – 400 с.

12. Хасти Т., Тибришани Р. Основы статистического обучения: интеллектуальный анализ данных, логический вывод и прогнозирование. – М., 2020. – 768 с.

13. Элбон К. Машинное обучение с использованием Python. Сборник рецептов. – СПб.: БХВ-Петербург, 2020. – 384 с.

14. Соревнование на Kaggle: https://www.kaggle.com/datasets/sulianova/cardiovascular-disease-dataset