



МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт информационных технологий
Кафедра инструментального и прикладного программного обеспечения

РАБОТА ДОПУЩЕНА К ЗАЩИТЕ


Заведующий
кафедрой _____ Р.Г. Болбаков

« ____ » _____ 2023 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
по направлению подготовки бакалавров 09.03.04 Программная инженерия

На тему: Интерактивное веб-приложение для анализа и визуализации данных с
использованием алгоритмов машинного обучения

Обучающийся


подпись

Миронов Дмитрий Сергеевич
Фамилия, имя, отчество

шифр 19И1366
группа ИКБО-20-19

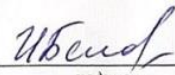
Руководитель работы


подпись

к.т.н., доцент,
доцент

Плотников С.Б.

Консультант


подпись

старший
преподаватель

Белоусова И.В.

Москва 2023 г.



МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт информационных технологий

Кафедра инструментального и прикладного программного обеспечения

СОГЛАСОВАНО

Заведующий
кафедрой

подпись

Болбаков Роман Геннадьевич

УТВЕРЖДАЮ

Директор
института

подпись

Зуев Андрей Сергеевич

«07»

марта

2023 г.

«07»

марта

2023 г.

ЗАДАНИЕ

на выполнение выпускной квалификационной работы бакалавра

Обучающийся

Миронов Дмитрий Сергеевич

Фамилия Имя Отчество

Шифр

19И1366

Направление
подготовки

09.03.04

индекс направления

Программная инженерия

наименование направления

Группа

ИКБО-20-19

1. Тема выпускной квалификационной работы: Интерактивное веб-приложение

для анализа и визуализации данных с использованием алгоритмов машинного обучения

2. Цель и задачи выпускной квалификационной работы

Цель работы: спроектировать и разработать интерактивное веб-приложение для анализа и визуализации данных с использованием алгоритмов машинного обучения.


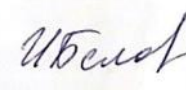
Задачи работы: провести анализ предметной области, в том числе конкурентных решений; определить информационные процессы предметной области и формализовать их; формализовать задачи на проектирование и разработку интерактивного веб-приложения; спроектировать веб-приложение (архитектуру, функциональную схему, адаптированную модель жизненного цикла, интерфейс и базу данных); определить и обосновать информационные, технические, программные средства для разработки веб-приложения; разработать интерактивное веб-приложение; произвести расчет вычислительной и ёмкостной сложности информационной системы; произвести тестирование информационной системы; рассчитать экономическую эффективность и стоимость проведения работ; оформить пояснительную записку согласно ГОСТ 7.32-2017.

3. Этапы выпускной квалификационной работы

№ этапа	Содержание этапа выпускной квалификационной работы	Результат выполнения этапа ВКР	Срок выполнения
1	Исследовательский раздел		22.03.2023
1.1	Анализ существующих веб-приложений для анализа и визуализации данных		
1.2	Анализ методов визуализации данных		
1.3	Анализ методов машинного обучения		
1.4	Выбор средств разработки веб-приложения		
1.5	Постановка задачи к разработке и исследованию веб-приложения		
1.6	Вывод к разделу 1		
2	Проектный раздел		22.03.2023
2.1	Проектирование адаптированной модели жизненного цикла разработки и исследования веб-приложения		
2.2	Разработка архитектуры, структурной и функциональной схемы веб-приложения		
2.3	Разработка архитектуры клиентской части веб-приложения		
2.4	Разработка архитектуры серверной части веб-приложения		
2.5	Вывод к разделу 2		
3	Технологический раздел		19.04.2023
3.1	Разработка интерфейса приложения		
3.2	Расчет вычислительной и емкостной сложности		
3.3	Тестирование приложения		
3.4	Вывод к разделу 3		
4	Экономический раздел		17.05.2023
4.1	Организация и планирование работ по теме		
4.2	Расчет стоимости проведения работ по теме		17.05.2023
5	Введение, заключение, список источников, приложения		
6	Презентационный материал, аннотация		
7	Нормоконтроль		26.05.2023

4. Перечень разрабатываемых документов и графических материалов: печатная и электронная версии выпускной квалификационной работы бакалавра, презентационный материал с основными результатами выпускной квалификационной работы бакалавра.

5. Руководитель и консультант выпускной квалификационной работы

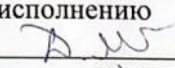
Функциональные обязанности	Должность в Университете	Фамилия, имя, отчество	Подпись
Руководитель ВКР	к.т.н., доцент, доцент	Плотников Сергей Борисович	
Консультант по экономическому разделу	старший преподаватель	Белоусова Ирина Викторовна	

Задание выдал
Руководитель ВКР:


подпись

«07» марта 2023 г.

Задание принял к исполнению
Обучающийся:


подпись

«07» марта 2023 г.

УДК 004

Руководитель ВКР: к.т.н., доцент С.Б. Плотников

Консультант ВКР: старший преподаватель, И.В. Белоусова

Д.С. Миронов. Выпускная квалификационная работа по направлению подготовки бакалавров 09.03.04 «Программная инженерия» на тему «Интерактивное веб-приложение для анализа и визуализации данных с использованием алгоритмов машинного обучения»: М. 2023 г., МИРЭА – Российский технологический университет, Институт информационных технологий (ИТ), кафедра Инструментального и Прикладного Программного Обеспечения (ИиППО) – стр. 54, рис. 23, табл. 5, ист. 30 (в т.ч. 10 на англ. яз.), прил. 4.

Ключевые слова: визуализация, машинное обучение, регрессия, классификация, кластеризация, Streamlit.

Целью работы является разработка программного инструмента, позволяющего пользователям вводить и анализировать данные с использованием алгоритмов машинного обучения и визуализировать результаты в интерактивном формате. Так же является проектирование архитектуры, реализация алгоритмов машинного обучения, разработка спроектированного решения для клиентской и серверной стороны, разработка базы данных.

D.S. Mironov. Final qualifying work in the bachelor's degree program 09.03.04 "Software Engineering" on the topic "Interactive web application for data analysis and visualization using machine learning": М. 2023, MIREA - Russian Technological University, Institute of Information Technology (IT), Department of Instrumental and Applied Software (IiPPO) – p. 56, Fig. 22, Table 5, ist. 30 (including 15 in English), adj. 4.

Keywords: visualization, machine learning, regression, classification, clustering, Streamlit.

The aim of the work is to develop a software tool that allows users to enter and analyze data using machine learning algorithms and visualize the results in an interactive format. It also includes architecture design, implementation of machine learning algorithms, development of a designed solution for the client and server side, database development.

МИРЭА – Российский технологический университет: 119454, Москва, пр-т Вернадского, д. 78

Тираж: 1 экз. (на правах рукописи)

Файл: «090304_19И1366_Миронов_ДС.pdf», исполнитель Миронов Д.С., email: mironov.d.s@edu.mirea.ru

© Д.С. Миронов.

ОГЛАВЛЕНИЕ

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ	7
ВВЕДЕНИЕ	8
1 Исследовательский раздел	9
1.1 Анализ существующих веб-приложений для анализа и визуализации данных.....	9
1.1.1 Ember Charts.....	9
1.1.2 ChartBlocks.....	9
1.1.3 BuLiAn – Анализ Бундеслиги	10
1.2 Анализ методов визуализации данных	10
1.2.1 Стандартные 2D/3D-образы.....	11
1.2.2 Отображение иконок	11
1.2.3 Методы ориентированные на пиксели	12
1.3 Анализ методов машинного обучения	12
1.3.1 Регрессия.....	12
1.3.2 Классификация	13
1.3.3 Кластеризация	14
1.4 Выбор средств разработки веб-приложения.....	15
1.4.1 Выбор языка разработки	15
1.4.2 Выбор алгоритмов машинного обучения.....	15
1.4.3 Выбор СУБД.....	15
1.5 Постановка задачи к разработке и исследованию веб-приложения	15
1.6 Вывод по первой главе	16
2 Проектный раздел	17
2.1 Проектирование адаптированной модели жизненного цикла разработки и исследования веб-приложения	17

2.2Разработка архитектуры, структурной и функциональной схемы веб-приложения.....	18
2.2.1Разработка структурной схемы веб-приложения	18
2.2.2Разработка функциональной схемы веб-приложения	19
2.3Разработка архитектуры клиентской части веб-приложения	21
2.4Разработка архитектуры серверной части веб-приложения	22
2.5Вывод по второй главе	23
3Технологический раздел	24
3.1Разработка клиентской части приложения	24
3.2Разработка серверной части веб-приложения	29
3.3Расчет вычислительной и емкостной сложности	31
3.4Тестирование приложения.....	32
3.4.1Чек-лист	32
3.4.2Тест-кейсы	32
3.4.3Анализ проведенного тестирования	34
3.5Вывод к разделу 3.....	34
4Экономический раздел.....	36
4.1Организация и планирование работ по теме.....	36
4.2Организация работ	36
4.3График проведения работ	38
4.4Расчёт стоимости проведения работ	39
4.5Выводы по четвертой главе	42
ЗАКЛЮЧЕНИЕ	43
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	45
ПРИЛОЖЕНИЕ А	48

ПРИЛОЖЕНИЕ В	49
ПРИЛОЖЕНИЕ С	53
ПРИЛОЖЕНИЕ D	54

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

СУБД – система управления базами данных.

HTTP – HyperText Transfer Protocol («протокол передачи гипертекста»).

IDEF – Integrated DEFinition (методология для решения задач моделирования сложных систем)

ORM – Object-Relational Mapping («объектно-реляционное отображение»).

SSL – Secure Sockets Layer («уровень защищённых сокетов»)

SQL – Structured Query Language («язык структурированных запросов»).

WSGI – Web-Server Gateway Interface («интерфейс шлюза веб-сервера»)

2D – 2-dimensional (два измерения)

3D – 3-dimensional (три измерения)

ВВЕДЕНИЕ

В современном мире, управляемом данными, потребность в эффективных методах визуализации данных становится все более важной. Использование алгоритмов машинного обучения очень эффективно при анализе и интерпретации больших объемов данных. Таким образом, разработка веб-приложения, использующего алгоритмы машинного обучения для визуализации данных, является перспективным направлением для улучшения понимания сложных наборов данных. Целью данной работы является представление дизайна и реализации такого веб-приложения, которое предоставит пользователям интуитивно понятный и интерактивный способ изучения и визуализации данных. Приложение также будет включать в себя различные алгоритмы машинного обучения, чтобы предоставить пользователям мощное понимание их данных. Цель этой работы — спроектировать и разработать интерактивное веб-приложение для анализа и визуализации данных с использованием алгоритмов машинного обучения.

В процессе написания выпускной квалификационной работы автор (дипломник) руководствовался следующими нормативными актами:

1. «О защите населения и территории от чрезвычайных ситуаций природного и техногенного характера» от 21.12.1994 № 68-ФЗ.
2. «Об основах охраны здоровья граждан в Российской Федерации» от 21.11.2011 № 323-ФЗ.
3. «О гражданской обороне» от 12.02.1998 № 28-ФЗ.
4. Приказ Минздравсоцразвития РФ от 04.05.2012 № 477н «Об утверждении перечня состояний, при которых оказывается первая помощь, и перечня мероприятий по оказанию первой помощи».
5. Трудовой кодекс Российской Федерации от 30.12.2001 № 197-ФЗ.
6. СанПиН 2.2.2/2.4.1340-03 «Гигиенические требования к персональным электронно-вычислительным машинам и организации работы».

1 Исследовательский раздел

1.1 Анализ существующих веб-приложений для анализа и визуализации данных

На сегодняшний день существует большое количество различных веб-приложений для анализа и визуализации данных. Они нужны чтобы понять в удобном формате большой объем данных и увидеть их закономерность.

1.1.1 Ember Charts

Ember Charts [1] является некой библиотекой диаграмм, созданную с помощью фреймворков Ember.js и d3.js. С его помощью [1] можно изучать графики, точечные и круговые диаграммы и гистограммы. Более того, он помогает вам с лёгкостью расширять и как-либо изменять диаграммы, так как предоставляет различный выбор настроек.

Основной функционал данного веб-приложения:

- предлагает мощные и совершенные диаграммы;
- он совершенно бесплатен;
- может расширить класс для создания своих специальных гистограмм.

Это веб-приложение идеально подходит для компаний, которым часто требуются статистические графики. Кроме того, он служит ценным инструментом для тех, кому требуются различные диаграммы, такие как гистограммы, горизонтальные диаграммы, круговые диаграммы, линейные диаграммы с накоплением данных и точечные диаграммы. Для людей, работающих с большими объемами данных, приложение предлагает простые в использовании функции и простые решения.

1.1.2 ChartBlocks

ChartBlocks [2]– это высокотехнологичное программное обеспечение для визуализации данных, которое помогает импортировать данные за короткое время. Более того, вы можете немедленно же обновить собственные данные в приложении ChartBlocks и выполнить необходимый импорт.

Основной функционал данного веб-приложения:

- упрощает загрузку массивов данных;

- позволяет создавать и опубликовывать диаграммы, столбцы и гистограммы за считанные минуты;
- может использовать одинаковые данные для создания разных диаграмм;
- совместимо со всеми типами мобильных устройств и подходит для экрана любого размера.

Этот инструмент предоставляет функцию «построитель диаграмм», которая помогает создавать индивидуальные диаграммы. После этого легко встроить диаграммы на свой веб-сайт или в любую платформу социальных сетей. Он имеет особенно продвинутый набор методов.

1.1.3 BuLiAn – Анализ Бундеслиги

Интерактивное приложение для анализа данных Бундеслиги с сезона 2013/2014 по сезон 2019/2020 [3].

Это интерактивное приложение, содержащее данные Бундеслиги с сезона 2013/2014 по сезон 2019/2020, позволяет вам узнать полную статистику игроков немецкой Бундеслигой.

Основной функционал данного веб-приложения:

- может использовать одни и те же данные для создания различных диаграмм;
- поставляется со встроенными инструментами для социальных сетей, которые помогут вам напрямую поделиться и встраивать свои диаграммы;
- упрощенный интерфейс для удобства использования;
- совершенно бесплатен;
- поддержка алгоритмов машинного обучения для различных диаграмм.

Благодаря Streamlit [4] – это фреймворк Python с открытым исходным кодом, используемый для развертывания моделей машинного обучения в красивых веб-приложениях всего в несколько строк кода.

1.2 Анализ методов визуализации данных

Успех визуализации прямо находится в зависимости от верности ее использования, а именно от выбора и грамотного использования типов графиков и диаграмм, которые мы выделим далее [5].

1.2.1 Стандартные 2D/3D-образы

Графики, гистограммы, диаграммы, и т.п. — самые простые методы визуализации [6]. Пример представлен на рисунке 1.2.1.1.

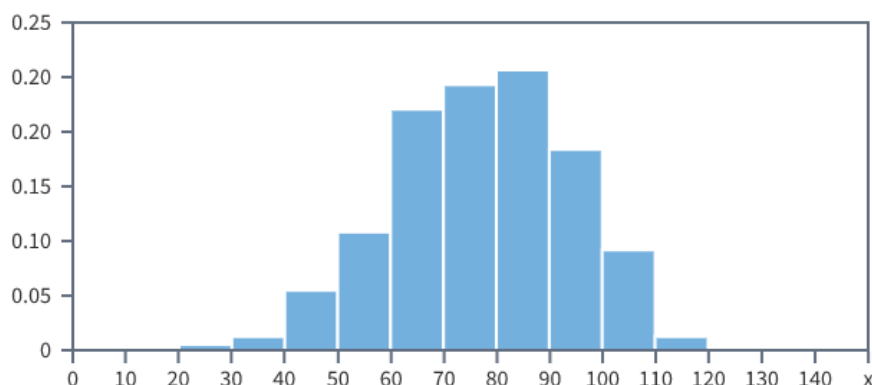


Рисунок 1.2.1.1 – Пример стандартного 2D образа [Разработано автором]

Основной недостаток этого метода — невозможность легко воспринимаемой визуализации сложных данных и данных в большом количестве. Но прост в реализации при добавлении их в приложение.

1.2.2 Отображение иконок

Еще один вид визуализации данных являются отображения иконок. Их главной задачей представляется собой отображение значений элементов многомерных данных в свойства образов которые могут представлять собой: человеческие лица, стрелки, звезды и т.д. Пример представлен на рисунке 1.2.2.1.

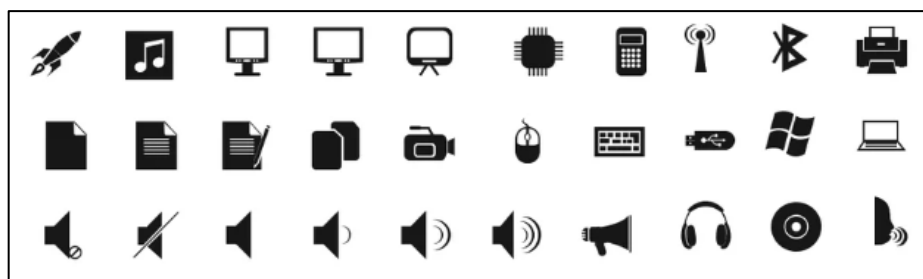


Рисунок 1.2.2.1 – Пример отображения иконок [Разработано автором]

Генерация визуализации включает в себя связывание характеристик компонентов данных с качествами изображений. Эти изображения могут быть

классифицированы для облегчения комплексного изучения данных. Как следствие, окончательное изображение представляет собой дизайн текстуры, который демонстрирует вариации, соответствующие свойствам данных.

1.2.3 Методы ориентированные на пиксели

Главной идеей методов, которые ориентируются на пиксели [8], представляет собой отображение любого измерения значения в цветной пиксель и из группировки по принадлежности к измерению. Пример представлен на рисунке 1.2.3.1.

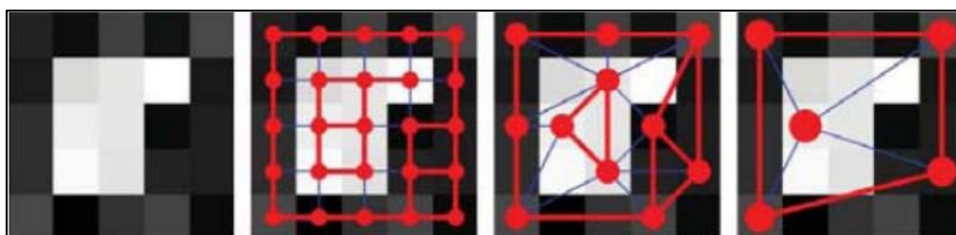


Рисунок 1.2.3.1 – Пример метода, ориентированного на пиксели [Разработано автором]

Этот метод использования одного пикселя для представления одной точки данных позволяет визуализировать значительный объем данных, превышающий один миллион значений. Он специально разработан для целей визуализации в компьютерном зрении.

1.3 Анализ методов машинного обучения

Искусство создания алгоритмов и статистических моделей, которые позволяют компьютерам выполнять задачи без прямых команд, полагаясь на шаблоны и логические выводы, известно как машинное обучение.

1.3.1 Регрессия

Регрессия применяется во множества сферах: экономика, компьютерные и социальные науки, прочее. Её значимость повышается с доступностью больших данных [9].

Регрессия разглядывает определенное явление и ряд наблюдений. Каждое наблюдение имеет две и более переменных. Предполагая, что одна переменная находится в зависимости от прочих, вы пытаетесь сформировать отношения между ними.

Прочими словами, вам нужно выявить функцию, которая отображает взаимосвязь одних переменных или данных от прочих. Зависимые данные именуются зависимыми переменными, выходами или ответами. Пример линейной регрессии представлен на рисунке 1.3.1.1.

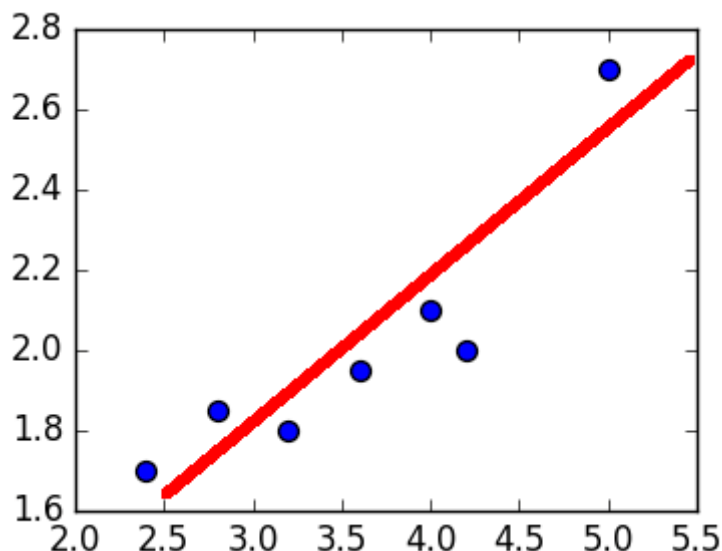


Рисунок 1.3.1.1. – Пример линейной регрессии [Разработано автором]

Самостоятельные данные именуются независимыми переменными, входами или предсказателями. Традиционно в регрессии имеется одна бесперебойная и неограниченная зависимая переменная. Входные переменные могут быть неограниченными, дискретными или категорическими данными, такими как пол, национальность, бренд [10].

1.3.2 Классификация

Процесс классификации включает организацию данных в установленные категории [11]. Несколько объектов сортируются в отдельные классы, а ограниченная группа объектов идентифицируется со знанием соответствующих классов, называемых обучающим набором. Рисунок 1.3.2.1 иллюстрирует классификацию.

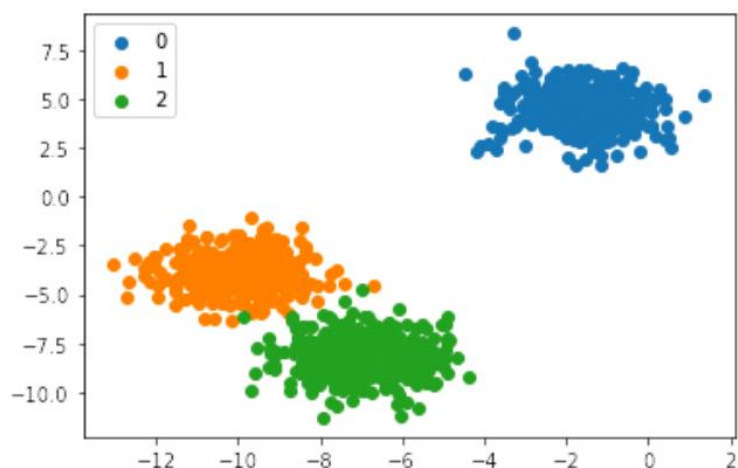


Рисунок 1.3.2.1. – Пример классификации [Разработано автором]

Предположим, вы стремитесь провести различие между клиентами, которые обычно совершают покупки, приносящие высокий доход, и теми, кто, скорее всего, переключится на другие бренды. Тщательно изучив предыдущие данные и определив ключевые показатели, характерные для каждой категории клиентов, вы можете использовать эту информацию для оценки текущих клиентов и прогнозирования их членства в группе. Впоследствии эти знания позволят вам усовершенствовать свою маркетинговую стратегию и потенциально превратить потенциальную потерю клиентов в базу лояльных клиентов. Такой пример иллюстрирует контролируемое машинное обучение.

1.3.3 Кластеризация

Кластеризация – это метод машинного обучения, который используется для группировки похожих экземпляров [12]. Этот метод используется в задачах машинного обучения без учителя, когда набор данных не помечен, и ваша задача состоит в том, чтобы сгруппировать похожие экземпляры. Пример кластеризации представлен на рисунке 1.3.3.1.

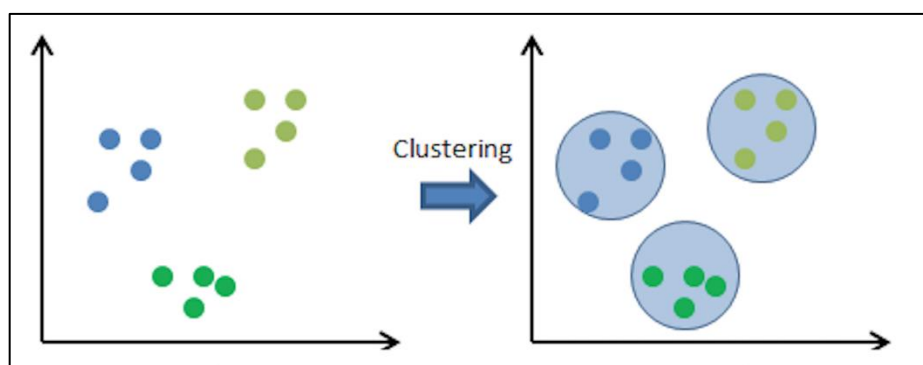


Рисунок 1.3.3.1 – Пример кластеризации [Разработано автором]

Алгоритмы кластеризации в основном используются в маркетинговых кампаниях, когда бизнес хочет найти самую прибыльную группу клиентов из своей базы данных всех клиентов [13].

1.4 Выбор средств разработки веб-приложения

1.4.1 Выбор языка разработки

Для реализации клиентской и серверной частей приложения был выбран язык программирования Python. Также он поддерживает работу с таким инструментом как Streamlit, что напрямую сокращает разработку веб-приложения.

Streamlit - библиотека Python с открытым кодом. Она позволяет с легкостью создавать разные веб-приложения для инженеров машинного обучения. Всего за несколько минут и пару строк кода можно создать приложения.

1.4.2 Выбор алгоритмов машинного обучения

Из рассмотренных методов машинного обучения в веб-приложении будут реализованы регрессия, классификация и кластеризация.

1.4.3 Выбор СУБД

В качестве СУБД для управления данными будет использоваться СУБД PostgreSQL, предоставляющая удобные средства для создания баз данных.

В качестве ORM-фреймворка был выбран Peewee, который позволяет работать с базами данных, не делая ручных запросов, и помогает сократить много времени за счет использования моделей.

1.5 Постановка задачи к разработке и исследованию веб-приложения

Для реализации веб-приложения необходимо определить какими характеристиками оно должно обладать. Наиболее важными качествами, которыми должно обладать веб-приложение, является время обработки данных, простота в использовании и несколько вариантов использования различных алгоритмов машинного обучения для различных массивов данных.

Так как это данное веб-приложение основано из рассматриваемых веб-приложений, BuLiAn является подходящим примером.

В качестве клиента используется Streamlit, на чем и основано приведенное выше веб-приложение. Streamlit — это платформа приложений с открытым исходным кодом для групп машинного обучения и обработки данных. Он использует веб-сервер Tornado под капотом. Tornado — это однопоточный веб-сервер, не основанный на WSGI.

Для реализации сервера нам понадобится использовать Nginx в качестве обратного прокси-сервера перед сервером Streamlit и включить SSL в Nginx. NGINX — это веб-сервер, который также можно использовать в качестве обратного прокси-сервера, балансировщика нагрузки, почтового прокси-сервера и кэша HTTP. Nginx — это бесплатное программное обеспечение с открытым исходным кодом.

Таким образом при реализации данного клиент-сервера на основе Streamlit на стороне клиента загружается массив данных и обрабатывается на сервере. После при выбранной вкладке будет отображенная визуализация.

1.6 Вывод по первой главе

В данной главе выбраны решения для улучшения веб-приложения BuLiAn с использованием различных методов машинного обучения и добавлением веб-сервера. По результатам проведенного исследования была выбран фреймворк Streamlit и веб-сервер Nginx используя методы машинного обучения регрессию, классификацию и кластеризацию.

2 Проектный раздел

2.1 Проектирование адаптированной модели жизненного цикла разработки и исследования веб-приложения

Рекомендованной методологией к разработке программных модулей является итерационная модель жизненного цикла.

В итерационной модели разработка начинается с простой реализации небольшого набора требований к программному обеспечению и итеративно улучшает развивающиеся версии до тех пор, пока вся система не будет реализована и готова к развертыванию [14]. На рисунке 2.1.1 представлена схема итерационной модели жизненного цикла.

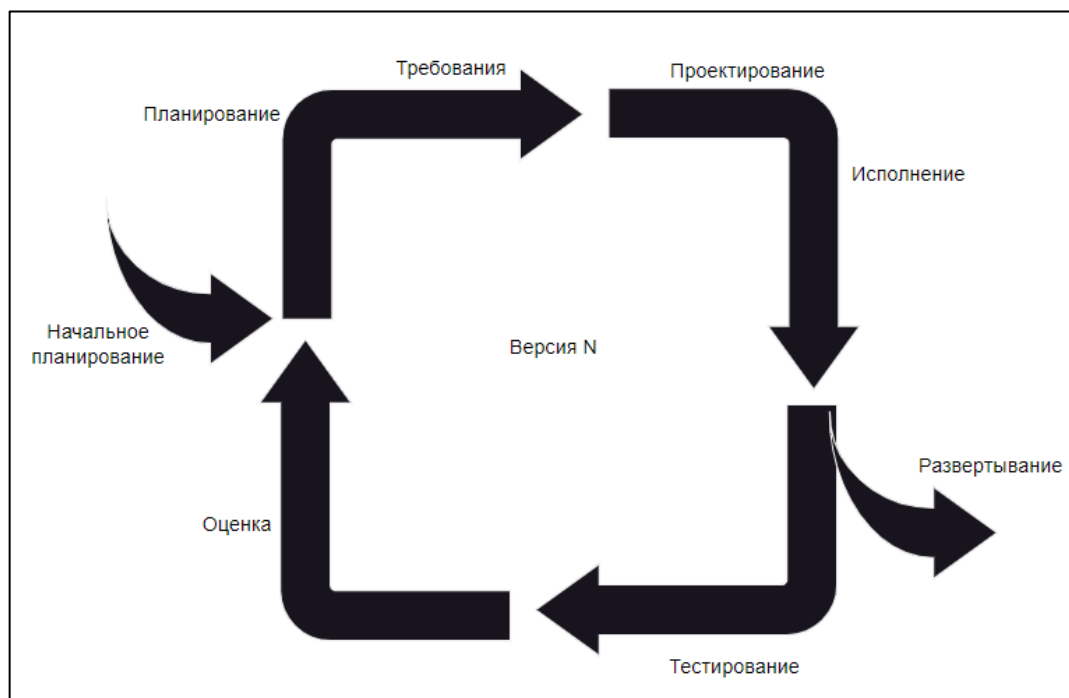


Рисунок 2.1.1 – Итерационная модель жизненного цикла [Разработано автором]

Преимущество данной модели заключается в том, что существует рабочая модель системы на очень ранней стадии разработки, что облегчает поиск функциональных или конструктивных проблем. Обнаружение проблем на ранней стадии разработки позволяет принимать корректирующие меры в условиях ограниченного бюджета.

2.2 Проектирование базы данных

На рисунке 2.2.1 изображена диаграмма базы данных веб-приложения. В базе данных содержится одна основная таблица users. Она нужна для

реализации блока с авторизацией и регистрацией пользователя, поскольку перед входом в веб-приложение, пользователю необходимо авторизоваться.

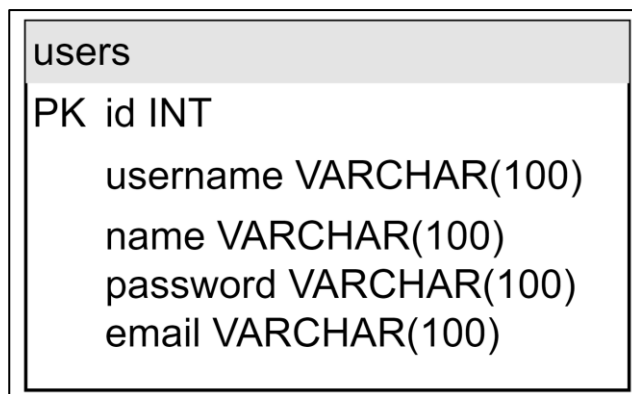


Рисунок 2.2.1 – Диаграмма базы данных [Разработано автором]

При необходимости база данных может быть расширена в зависимости от специфики выполняемых поисковых запросов. Добавление таблиц в базу данных не изменяет основной функционал приложения.

2.3 Разработка архитектуры, структурной и функциональной схемы веб-приложения

2.3.1 Разработка структурной схемы веб-приложения

Для разработки веб-приложения была выбрана клиент-серверная архитектура, так как она позволяет разделить логику и интерфейс программы, что значительно снизит нагрузку [15]. Структурная схема приложения представлена на рисунке 2.2.1.

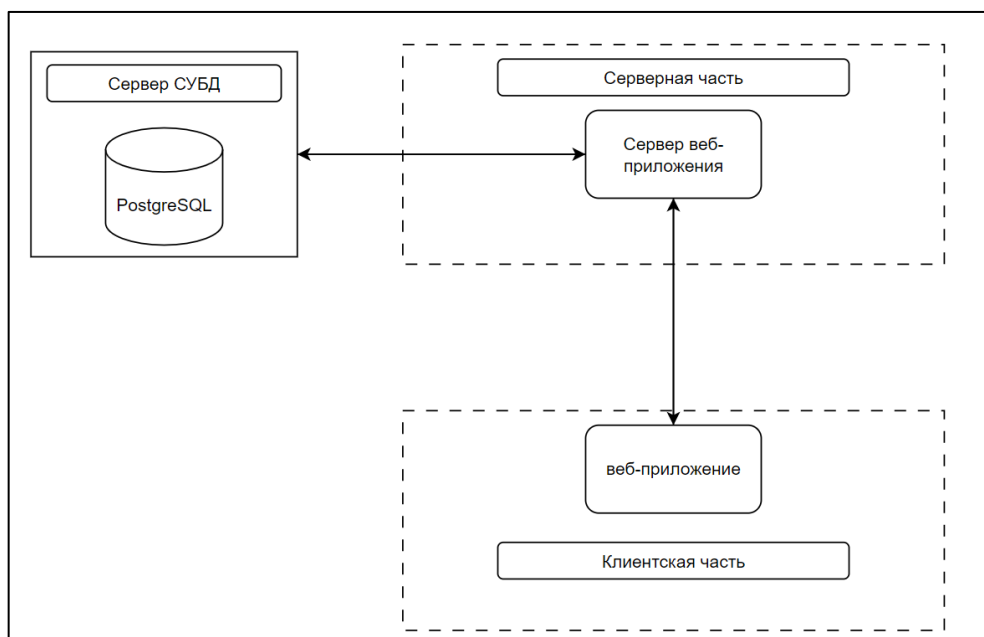


Рисунок 2.2.1 – Структурная схема приложения [Разработано автором]

Структура клиент-серверного приложения состоит из двух частей: серверной и клиентской.

Сервер приложения отвечает за приём запросов от клиентской части по протоколу WebSocket. Сервер может не только отвечать на запрос клиента, но и самостоятельно передавать новую информацию по мере ее поступления. Обмен данными происходит в рамках одного установленного соединения в режиме реального времени.

Сервер PostgreSQL отвечает за хранение данных в базе данных, за возможность обращения к ним через SQL-синтаксис, а также за выполнение других операций.

2.3.2 Разработка функциональной схемы веб-приложения

Была спроектирована контекстная диаграмма в нотации IDEF0. Схема представлена на рисунке 2.2.2.1.

Потоками входной информации были выбраны:

- данные от пользователя;
- массив данных.

На вход по управлению выбраны:

- закон о персональных данных;
- руководство пользователя.

Механизмами информационной системы являются:

- пользователь;
- модель машинного обучения.

В качестве выхода получен следующий информационный элемент:

- визуализированные данные.

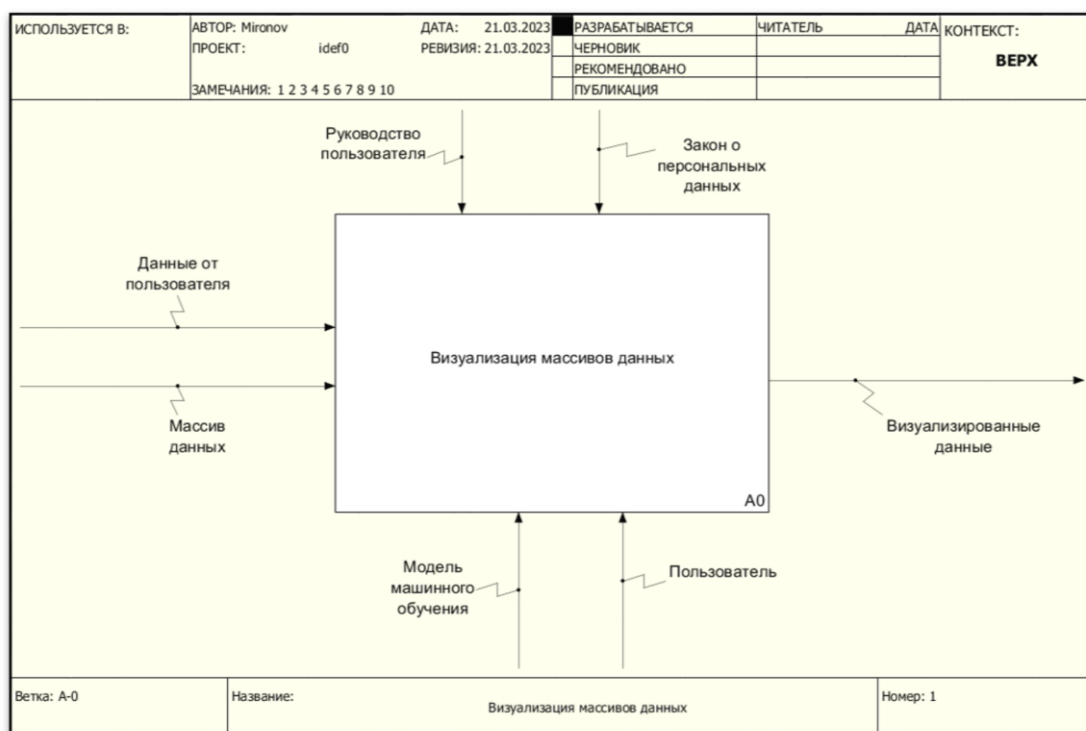


Рисунок 2.2.2.1 – Контекстная диаграмма системы [Разработано автором]

На диаграмме уровня A0 декомпозиции функционального блока «Визуализация массивов данных» обозначены процессы и функциональные блоки, выполняемые в рамках процедуры:

- регистрация (A1);
- загрузка данных (A2);
- визуализация (A3).

Данные процессы изображены на рисунке 2.2.2.2. Действия начинаются с регистрации пользователя и выбором нужных ему параметров где будет происходить визуализация загруженного массива данных.

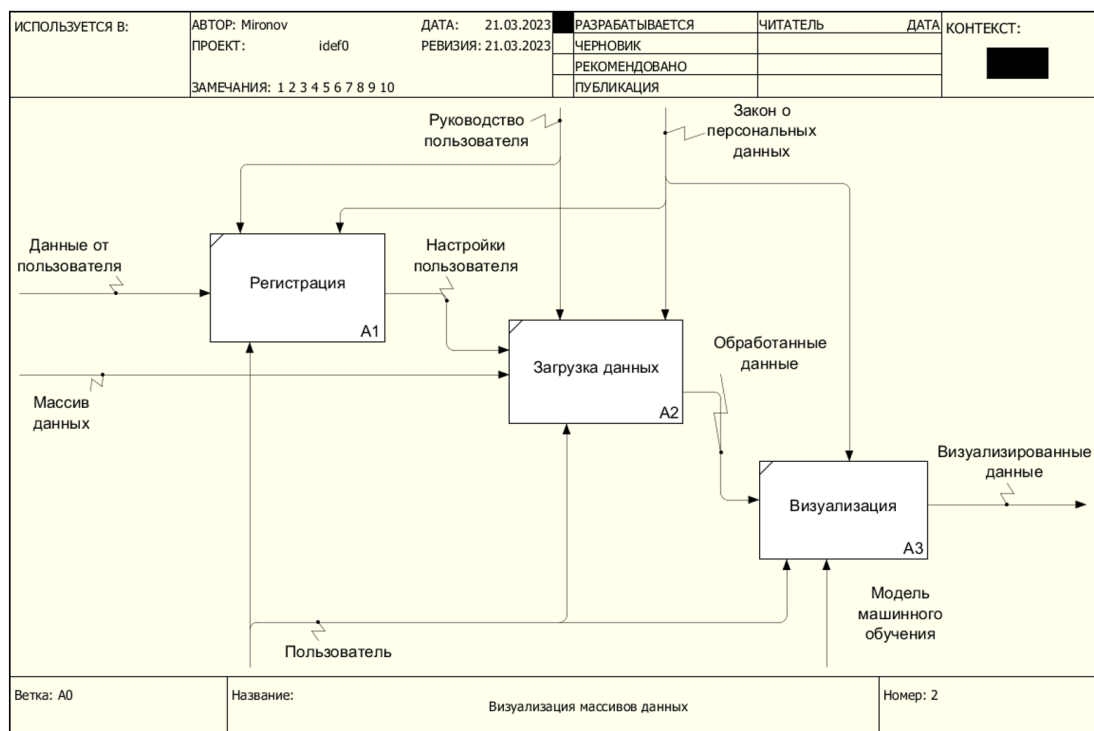


Рисунок 2.2.2.2 – Диаграмма декомпозиции блока «Визуализация данных» в нотации IDEF0 [Разработано автором]

После пользователь загружает свой массив данных и ждем пока программа обработает запрос и выведет на веб-странице его запрос на визуализацию данных.

2.4 Разработка архитектуры клиентской части веб-приложения

Клиентская часть приложения содержит три основных страницы: авторизация\регистрация, главная страница с описанием и загрузкой массива данных, страница с визуализацией [16]. Автором предложена схема клиентской части веб-приложения, представлена на рисунке 2.3.1.

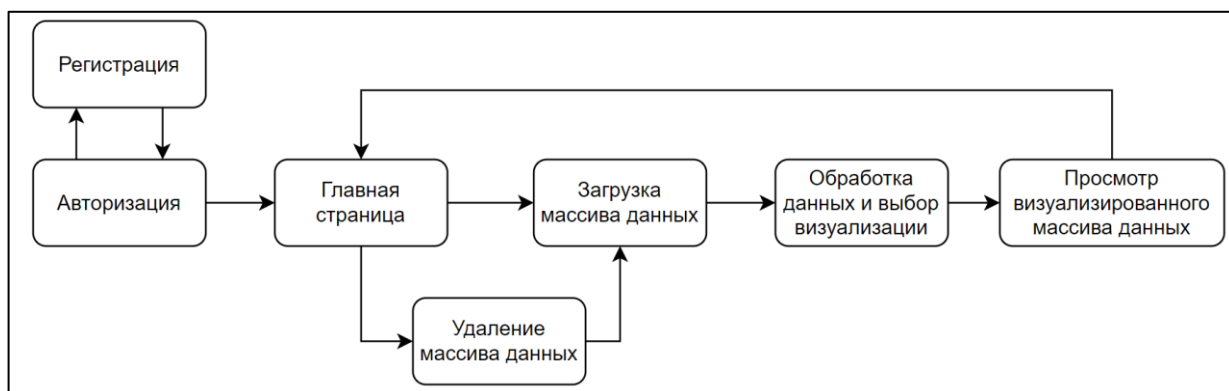


Рисунок 2.3.1 – Схема клиентской части приложения [Разработано автором]

Благодаря авторизации, настройки пользователя сохраняются и при следующем использовании веб-приложения настройка остается такой же, как и в последнее посещение пользователя.

На главной странице пользователю предоставлено описание и возможность загрузить свой массив данных.

После загрузки массива данных происходит обработка и пользователь переходит по вкладкам веб-приложения чтобы просмотреть визуализированные данные. На основе загруженных данных формируются различные графики, после чего отображаются во вкладках. При нажатии на одну из вкладок открывается страница, в котором отображаются график с использованием машинного обучения.

Если пользователю необходимо загрузить новый массива данных, то он переходит на главный экран и удаляет уже загруженные им ранее данные.

2.5 Разработка архитектуры серверной части веб-приложения

Функциональность веб-сокетов обеспечивается комбинацией двух компонентов, а именно Nginx и Tornado. Nginx — это веб-сервер, который может выступать в качестве обратного прокси-сервера, а Tornado, веб-фреймворк Python, имеет встроенную поддержку веб-сокетов. Серверная часть веб-приложения представлена на рис. 3.2.1.

Чтобы включить поддержку веб-сокетов на сервере с использованием Nginx и Tornado, необходимо выполнить следующие шаги: установить Nginx и Tornado на сервер, настроить Nginx в качестве обратного прокси-сервера для пересылки запросов в Tornado, настроить Tornado для обработки запросов с использованием веб-сокетов и разрешить трафик веб-сокетов, открыв необходимые порты в брандмауэре сервера.

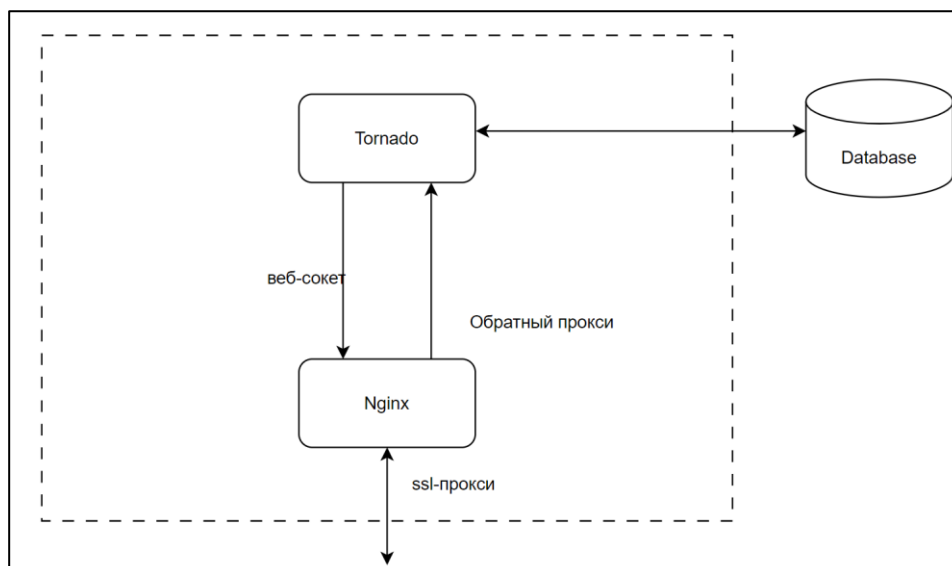


Рисунок 3.2.1 – Схема серверной части веб-приложения [Разработано автором]

Сервер Tornado служит серверным компонентом приложения, обрабатывая запросы клиентов и передавая обновления в реальном времени через веб-сокеты. Архитектура Tornado, управляемая событиями, позволяет эффективно обрабатывать многочисленные соединения. Использование инфраструктуры Tornado для архитектуры веб-приложений с веб-сокетами дает масштабируемое и эффективное решение для создания высокопроизводительных веб-приложений с малой задержкой в реальном времени.

2.6 Вывод по второй главе

В заключение, итеративная модель жизненного цикла была выбрана как наиболее подходящий подход к разработке веб-приложения. Структурные и функциональные схемы также были разработаны для обеспечения четкого понимания компонентов и функций системы. Кроме того, архитектура клиентской и серверной частей была разработана для обеспечения эффективной связи и бесперебойной координации между ними. Успешное завершение этого этапа закладывает основу для успешной разработки надежного веб-приложения, отвечающего потребностям конечных пользователей.

3 Технологический раздел

3.1 Разработка клиентской части приложения

Для пользователя итогового продукта важны быстродействие и простота его использования. При проектировании понятного и удобного интерфейса снижается риск ошибки пользователей, а также упрощается процесс понимания целей и функций.

Пользователь должен зарегистрироваться и после чего войти в приложение, если он уже был ранее на данном сайте, то просто ввести свой логин и пароль [19]. На рисунке 3.1.1 и 3.1.2 представлены регистрация и авторизация в веб-приложение.

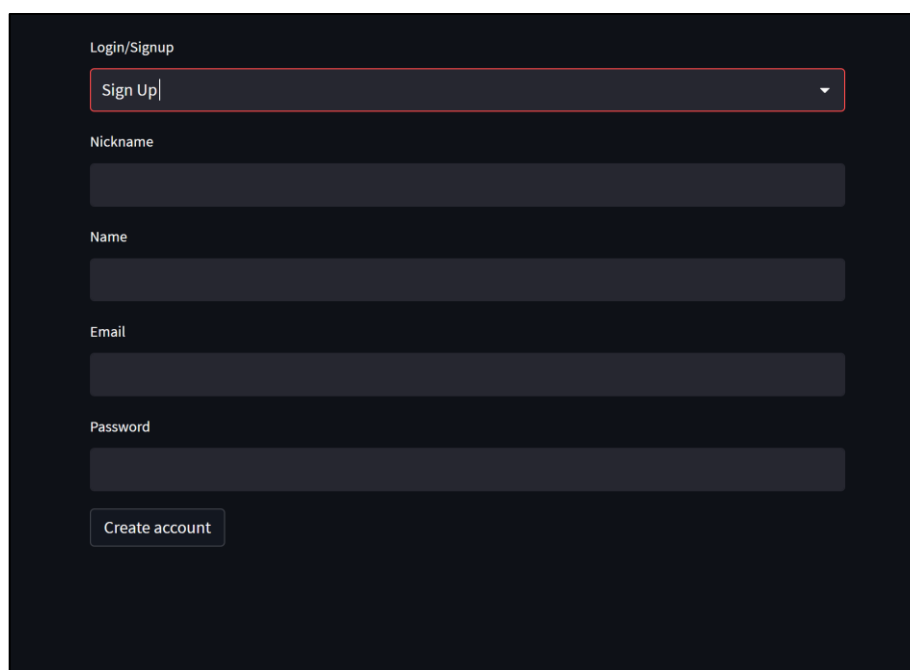
The image shows a dark-themed web interface for user registration. At the top, there is a 'Login/Signup' header with a dropdown menu currently set to 'Sign Up'. Below this are four input fields labeled 'Nickname', 'Name', 'Email', and 'Password'. Each field is represented by a dark gray rectangular box. At the bottom of the form is a button labeled 'Create account'.

Рисунок 3.1.1 – Регистрация в веб-приложение [Разработано автором]

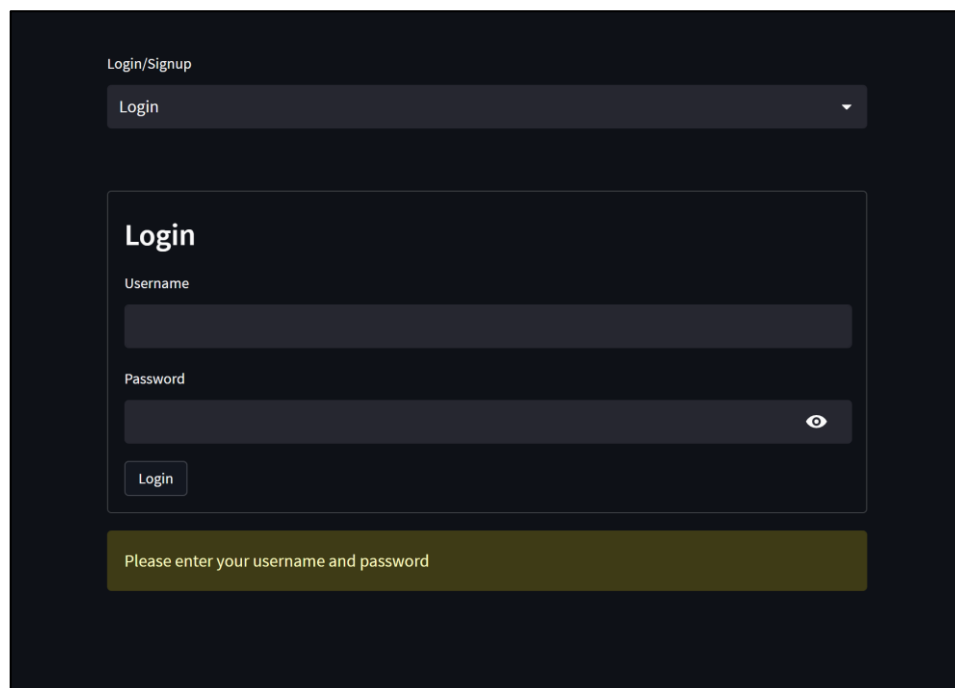


Рисунок 3.1.2 – Авторизация в веб-приложение [Разработано автором]

Фреймворк Streamlit каждый раз запускает код повторно, то после успешного входа код заново отработывает и проверяет что изменилась переменная входа, после чего отображает другой блок условия, в котором находится стартовая страница. Код представлен на листинге 3.1.1.

Листинг 3.1.1 – Код проверка и отображение регистрации, авторизации

```
if choi == "Login": # аутентификация
    # --- USER AUTHENTICATION ---
    auto_account = Authentif_account()
    credentials_in_class = auto_account.convertate_to_dict()
    authenticator = stauth.Authenticate(
        credentials_in_class, "app_home", "auth",
        cookie_expiry_days=30) # запоминает куки
    name, authentication_status, username =
    authenticator.login(
        "Login", "main")
else: # регстрация
    authentication_class = Create_account() # класс для
    регистрации
    sumbit = st.button('Create account')
    if sumbit:
```

Продолжение листинга 3.1.1

```
authentication_class.request_in_class_function_insert_data()  
    chois = "Login"
```

После успешного входа пользователь переходит на главную страницу где может загрузить свой массив данных.

Загрузка данных происходит на главной странице, главная страница с загруженными данными представлена на рисунке 3.1.3.

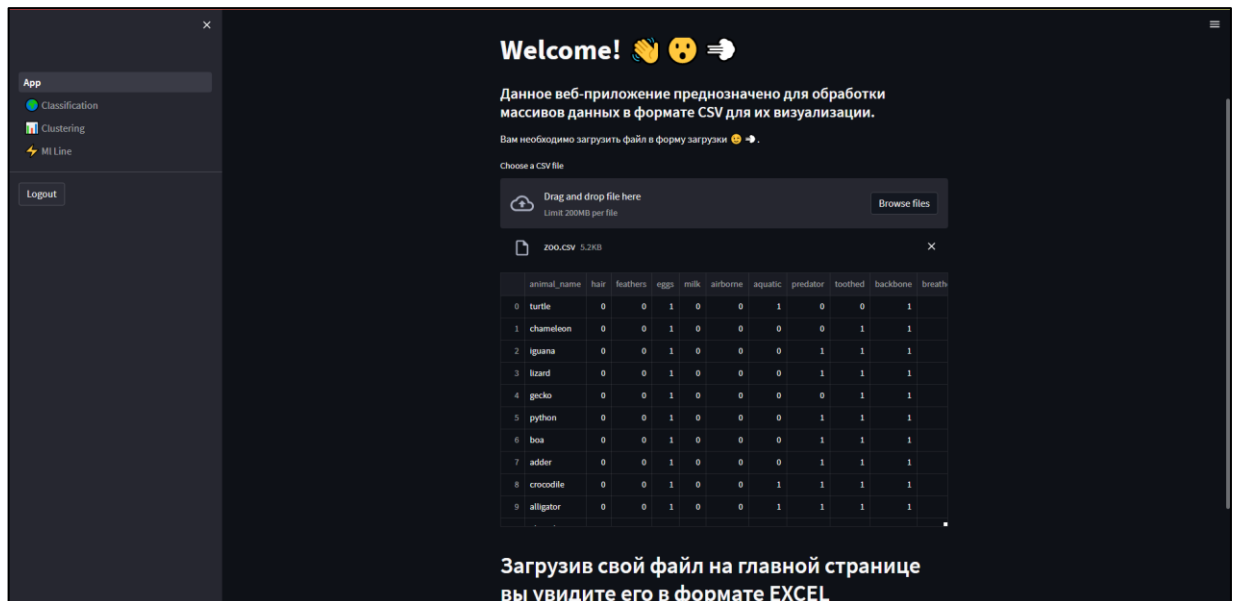


Рисунок 3.1.3 – Главная страница с загруженным массивом данных
[Разработано автором]

Благодаря фреймворку Streamlit расположение вкладок на боковой панели зависит от расположения и названия папок в структуре проекта. Для того чтобы вкладки находились на боковой панели необходимо в структуре проекта их разместить в отдельную папку под определенным названием (рисунок 3.1.4)

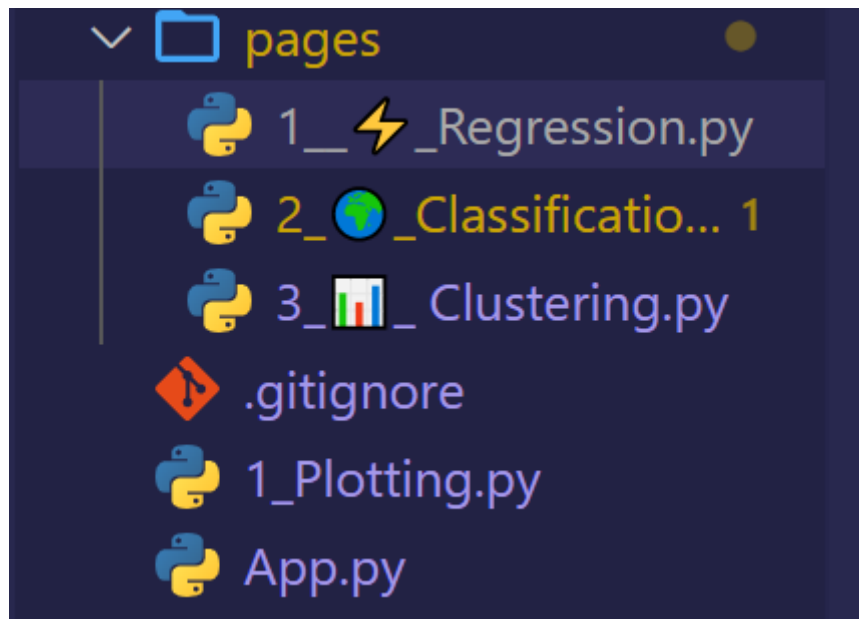


Рисунок 3.1.4 – Структура проекта для отображения вкладок [Разработано автором]

Как видно из рисунка выше, три файла находятся в папке «pages» и они отображаются на боковой панели, а файл «1_Plotting.py» не отображается на боковой панели

Рассмотрим вкладки, расположенные на боковой панели. Первая вкладка отвечает за отображения регрессии, отображения регрессии показано на рисунке 3.1.5. Код представлен в приложении А.

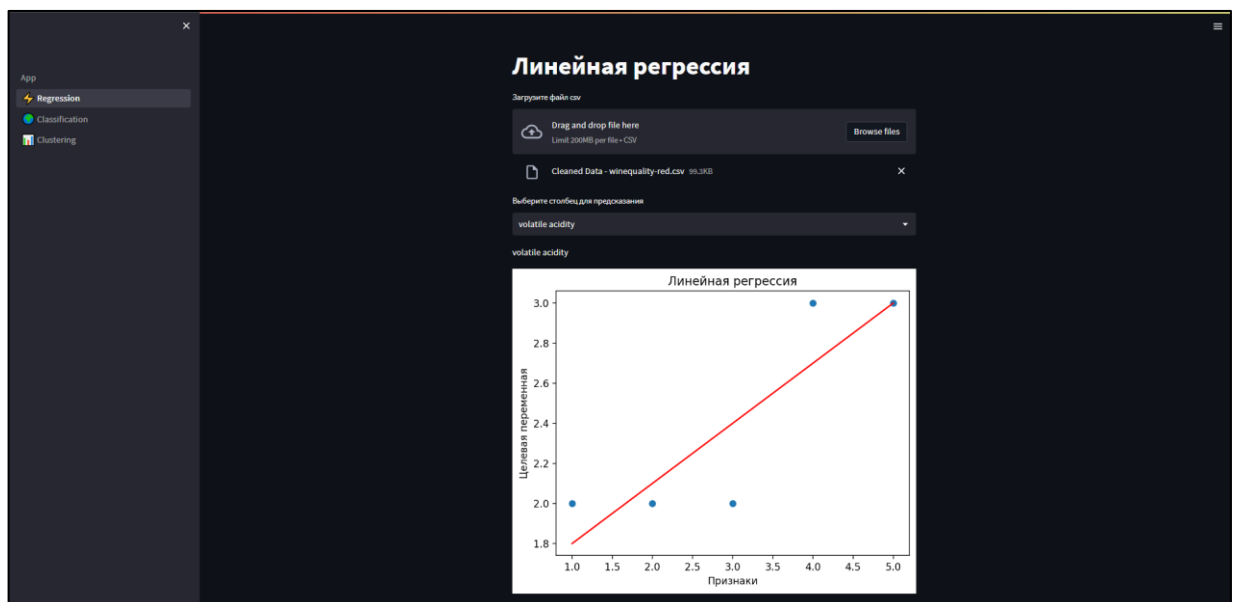


Рисунок 3.1.5 – Визуализированный массив данных при помощи линейной регрессии [Разработано автором]

Следующая вкладка, предназначенная для визуализации массива данных при помощи классификации. Пользователь так же может выбрать на боковой панели различные варианты алгоритма классификации (рисунки 3.1.6 – 3.1.7). Код представлен в приложении В.

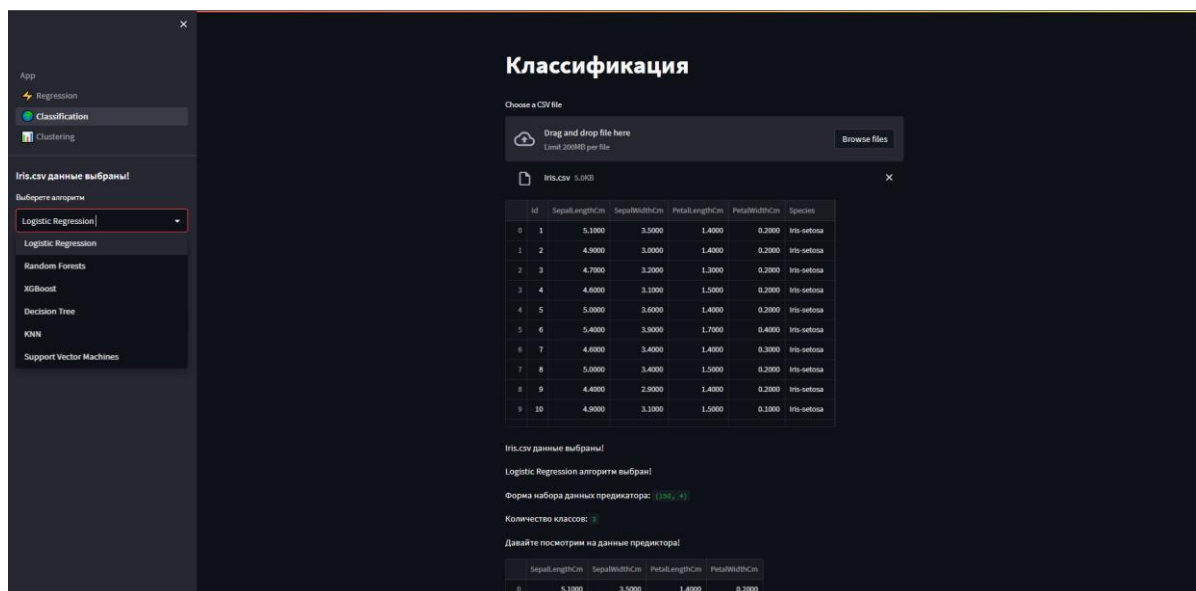


Рисунок 3.1.6 – Вкладка с алгоритмом классификации [Разработано автором]

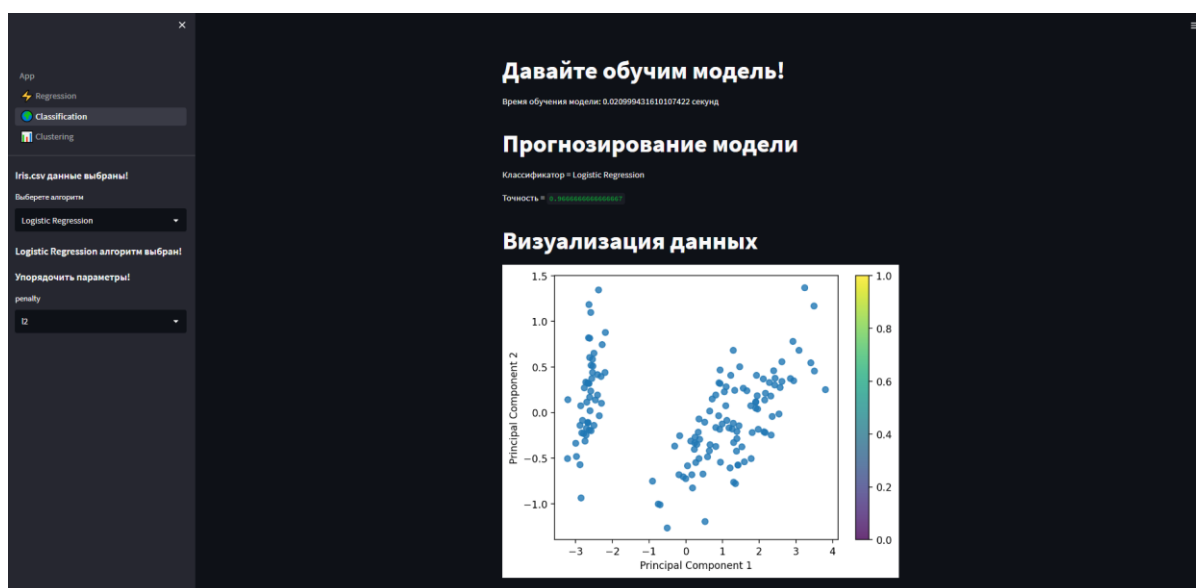


Рисунок 3.1.7 – Вкладка с алгоритмом классификации [Разработано автором]

Последняя вкладка, предназначенная для визуализации при помощи кластеризации. Выбрав столбцы из выпадающего списка и определив количество кластеров при помощи слайдера чисел, пользователю в реальном времени будет представлена визуализация загруженного массива данных (рисунок 3.1.8). Код представлен в приложении С.

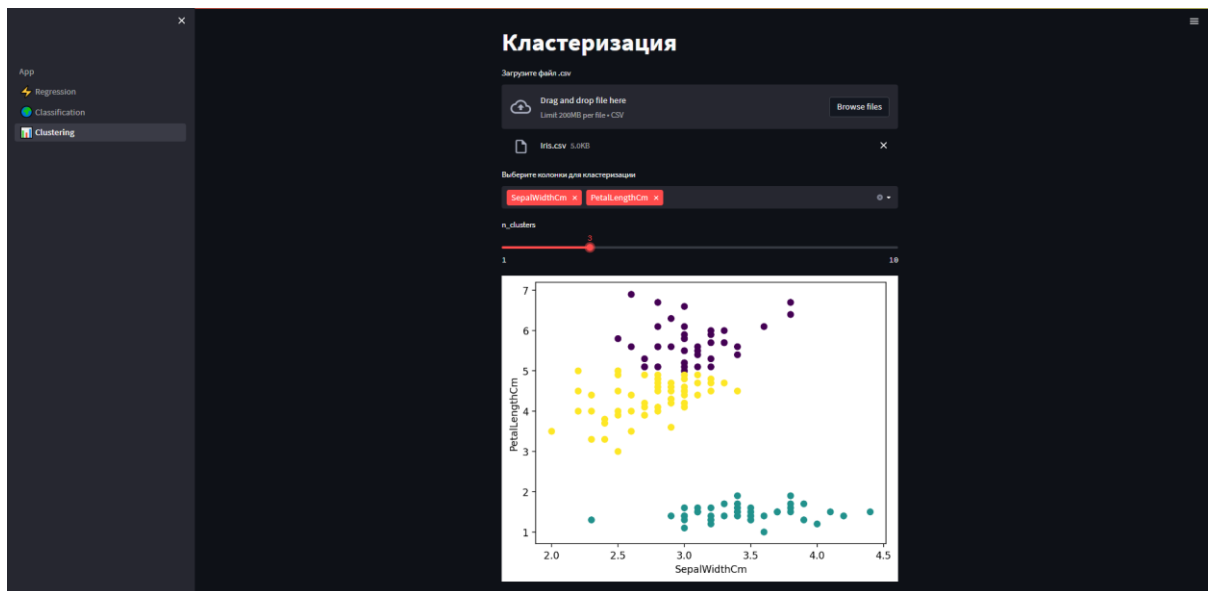


Рисунок 3.1.8 – Вкладка с алгоритмом кластеризации [Разработано автором]

3.2 Разработка серверной части веб-приложения

Так как при разработке архитектуры серверной части веб-приложения была использована архитектура с использованием фреймворка Tornado и Nginx необходимо их настроить [20].

Для того чтобы использовать веб-сокеты, представлено на листинге 3.2.1, необходимо настроить Nginx и настроить защиту трафика при помощи SSL. Для этого нужно отредактировать файл конфигурации Nginx (/etc/nginx/nginx.conf).

Листинг 3.2.1 – Использование веб-сокетов и защита трафика SSL

```
server {

    listen 443 ssl;
    server_name example.com;

    ssl_certificate
/etc/letsencrypt/live/example.com/fullchain.pem;
    ssl_certificate_key
/etc/letsencrypt/live/example.com/privkey.pem;
    location / {
        proxy_pass http://localhost:8888;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
```

Продолжение листинга 3.2.1.

```
        proxy_set_header Connection $connection_upgrade;
        proxy_set_header Host $host;
    }}
```

Этот блок определяет, какие запросы нужно перенаправлять на сервер Tornado и какие заголовки нужно добавлять для поддержки веб-сокетов, представлено на листинге 3.2.2.

Листинг 3.2.2 – Обработчик запросов

```
import tornado.ioloop
import tornado.web
import tornado.websocket
class MainHandler(tornado.web.RequestHandler):
    def get(self):
        self.get_apps()
class WebSocketHandler(tornado.websocket.WebSocketHandler):
    def open(self):
        self.open_apps()
    def on_message(self, message):
        self.on_mess(message)
    def on_close(self):
        self.close_apps()
def make_app():
    return tornado.web.Application([
        (r"/", MainHandler),
        (r"/ws", WebSocketHandler),
    ])
if __name__ == "__main__":
    app = make_app()
    app.listen(8888)
    tornado.ioloop.IOLoop.current().start()
```

Этот код создает два обработчика запросов: MainHandler для обработки обычных HTTP-запросов и WebSocketHandler для обработки веб-сокетов.

Определяем класс MainHandler, который наследуется от tornado.web.RequestHandler. Этот класс обрабатывает GET-запросы.

Определяем класс `WebSocketHandler`, который наследуется от `tornado.websocket.WebSocketHandler`. Этот класс обрабатывает веб-сокеты. Метод `open` вызывается при открытии соединения, метод `on_message` вызывается при получении сообщения, а метод `on_close` вызывается при закрытии соединения.

Определяем функцию `make_app`, которая создает экземпляр `tornado.web.Application` и определяет маршруты для обработчиков запросов.

Запускаем сервер, если этот файл был запущен напрямую. Создаем экземпляр `tornado.web.Application` и запускаем его на порту 8888. Затем запускаем цикл событий Tornado.

Для работы базы данных с сервером необходимо добавить файл с кодом предоставленном на листинге 3.2.3.

Листинг 3.2.3 – Подключение базы данных

```
class Db:
    def __init__(self):
        self.connection = psycopg2.connect(
            host=host,
            user=user,
            password=password,
            database=db_name
        )
        self.connection.autocommit = True # автоматическое
        # сохранение БД
        self.cursor = self.connection.cursor()

    def request_m(self, request, values = None):
        self.cursor.execute(request, values)
        res = self.cursor.fetchall()
        self.connection.commit()
        return res
```

Сервер сможет обрабатывать подключения через веб-сокеты и будет использовать базу данных PostgreSQL для хранения и извлечения данных.

3.3 Расчет вычислительной и емкостной сложности

Для выделения ресурсозатратных алгоритмов созданного модуля необходимо рассчитать их временную и емкостную сложность в худшем случае. При расчете будут рассмотрены только основные функции, которые используют алгоритмы машинного обучения. Количество данных, с которыми работают модули, будем принимать за n . Результаты расчета сложности алгоритмов представлены в таблице 3.3.1.

Таблица 3.3.1 – расчет сложности алгоритмов

Название алгоритма	Описание	T
Регрессия	Связи между переменными	$O(n)$
Классификация	Отнесение объекта к одной из категорий на основании его признаков	$O(m * n)$
Кластеризация	Разбиение множества объектов на подмножества, называемые кластерами	$O(k * n * t)$

Как видно при использовании данных средних размеров не является критичным для работы программы.

3.4 Тестирование приложения

Для проведения тестирования разработанного приложения был составлен чек-лист и соответствующие тест-кейсы.

3.5 Чек-лист

Покрытие тестированием:

Для каждой тестируемой особенности приложения, необходимо написать, как минимум 1 тест-кейс.

Чек-лист:

- 1) Проверить успешное выполнение регистрации
- 2) Проверить успешное выполнение авторизации
- 3) Проверить загрузку массива данных
- 4) Проверить визуализацию данных с использованием машинного обучения

3.6 Тест-кейсы

В таблицах 3.4.2.1 – 3.4.2.4 описаны тест-кейсы покрытия требований к разрабатываемому продукту.

Таблица 3.4.2.1 – Тест-кейс «регистрация»

ID / Priority: 591ddb4d872f-e189-9a94-838c-e5eb2367 / 1
IDEA: Пользователь зарегистрирован. ADDITIONAL INFO: Email – «mironovds@gmail.com» Имя – «dima» Никнейм – «dmirs» Пароль – «79262001»
REVISION HISTORY: – Created // 08.05.2023 // Миронов Д.С. –
PROCEDURE: 1. Открыть веб-приложение. 2. Выбрать вкладку зарегистрироваться. 3. Ввести данные. 4. Нажать на кнопку зарегистрироваться.
EXPECTED RESULT: Успешно зарегистрирован пользователь.

Таблица 3.4.2.2 – Тест-кейс «авторизация»

ID / Priority: 591ddb4d872f-e189-9a94-838c-e5eb2368 / 1
IDEA: Пользователь авторизован. ADDITIONAL INFO: Никнейм – «dmirs» Пароль – «79262001»
REVISION HISTORY: – Created // 08.05.2023 // Миронов Д.С. –
PROCEDURE: 1. Открыть веб-приложение. 2. Выбрать вкладку войти. 3. Ввести данные. 4. Нажать на кнопку войти.
EXPECTED RESULT: Успешно авторизован пользователь и переход на стартовую страницу.

Таблица 3.4.2.3 – Тест-кейс «загрузка массива данных»

ID / Priority: 591ddb4d872f-e189-9a94-838c-e5eb2369 / 1
--

Продолжение таблицы 3.4.2.3

IDEA: Массив данных успешно загружен. ADDITIONAL INFO: Название файла – 123.csv
REVISION HISTORY: – Created // 08.05.2023 // Миронов Д.С. –
PROCEDURE: 1. Нажать на кнопку загрузки данных. 2. Выбрать в проводнике файл с данными. 3. Нажать на кнопку.
EXPECTED RESULT: Все манипуляции проведены успешно без возникновения ошибок.

Таблица 3.4.2.4 – Тест-кейс «визуализацию данных»

ID / Priority: 591ddb4d872f-e189-9a94-838c-e5eb2317 / 1
IDEA: Визуализация данных с использованием алгоритмов машинного обучения
REVISION HISTORY: – Created // 08.05.2023 // Миронов Д.С. –
PROCEDURE: 1. Выбрать вкладку. 2. Дождаться окончания формирования визуализации. 3. Взаимодействие с визуализированными данными.
EXPECTED RESULT: Успешное визуализированные данных и взаимодействие с ними.

3.6.1 Анализ проведенного тестирования

Согласно тестовому плану, после разработки тест-кейсов программный продукт считается покрытым тестированием.

В результате проведенных тестов ошибок не было обнаружено. Из этого следует, что программный продукт отвечает всем приведенным требованиям.

3.7 Вывод к разделу 3

В результате выполнения работ, описанных в третьей главе, были созданы клиентская и серверная часть программного модуля, отличающиеся быстродействием и эффективностью.

Было проведено тестирование модуля для проверки и дальнейшего исправления ошибок, а также для выделения показателей временных затрат.

Тестирование показало высокую скорость работы и корректность выполненных действий.

Веб-приложение может позволить компаниям и частным лицам получать ценную информацию из своих данных выявить закономерности, тенденции и корреляции, которые в противном случае могли бы остаться незамеченными, позволяя пользователям принимать обоснованные решения и предпринимать активные шаги для повышения своей производительности.

4 Экономический раздел

4.1 Организация и планирование работ по теме

В составе работы задействовано 3 человека:

1) руководитель (руководитель выпускной квалификационной работы Плотников Сергей Борисович, к.т.н., доцент, кафедра ИиППО) – отвечает за грамотную постановку задачи, контролирует отдельные этапы работы, вносит необходимые коррективы и оценивает выполненную работу в целом;

2) консультант (консультант по экономической части ВКР, должность кафедра сокр.) – отвечает за консультирование экономической части выпускной квалификационной работы;

3) разработчик (студент 4-го курса Миронов Дмитрий Сергеевич, группа ИКБО-20-19) – реализация всех поставленных задач, в том числе проведение тестирования готового продукта и подготовка проектной документации.

Состав задействованных в работе участников представлен на рисунке 4.1.1.



Рисунок 4.1.1 - Взаимодействие специалистов в проекте [Разработано автором]

4.2 Организация работ

На разработку отводится 90 рабочих дней.

Этапы разработки представлены в таблице 4.2.1.

Таблица 4.2.1. – Этапы разработки

	Название этапа	Исполнитель	Трудоемкость, чел/дни	Продолжительность работ, дни
1	Разработка и утверждение технического задания	Руководитель	1	5
		Разработчик	5	
2	Исследовательский раздел:		26	25
2.1	Анализ существующих веб-приложений для анализа и визуализации данных	Разработчик	8	8
2.2	Анализ методов визуализации данных	Разработчик	3	3
2.3	Анализ методов машинного обучения	Разработчик	7	7
2.4	Выбор средств разработки веб-приложения	Разработчик	3	3
2.5	Постановка задачи к разработке и исследованию веб-приложения	Разработчик	4	4
		Руководитель	1	
3	Проектный раздел:		35	35
3.1	Проектирование адаптированной модели жизненного цикла разработки и исследования веб-приложения	Разработчик	9	9
3.2	Разработка архитектуры, структурной и функциональной схемы веб-приложения	Разработчик	6	6
3.3	Разработка архитектуры клиентской части веб-приложения	Разработчик	9	9
3.4	Разработка архитектуры серверной части веб-приложения	Разработчик	11	11

Продолжение таблицы 4.2.1

4	Технологический этап:		27	25
4.1	Разработка интерфейса приложения	Разработчик	12	12
4.2	Расчет вычислительной и емкостной сложности	Разработчик	8	8
		Руководитель	1	
4.3	Тестирование приложения	Разработчик	5	5
		Руководитель	1	
5	Экономический этап:		7	5
5.1	Организация и планирование работ по теме	Разработчик	3	3
		Консультант	1	
5.2	Расчет стоимости проведения работ по теме	Разработчик	2	2
		Консультант	1	
6	Сдача готового продукта	Разработчик	1	1
		Руководитель	1	
		Консультант	1	
Итого			104	96

4.3 График проведения работ

Календарный график исполнения работы представлен на рисунке 4.3.1.

Из рисунка 4.3.1 так же видно, что общий срок разработки составит 96 день.

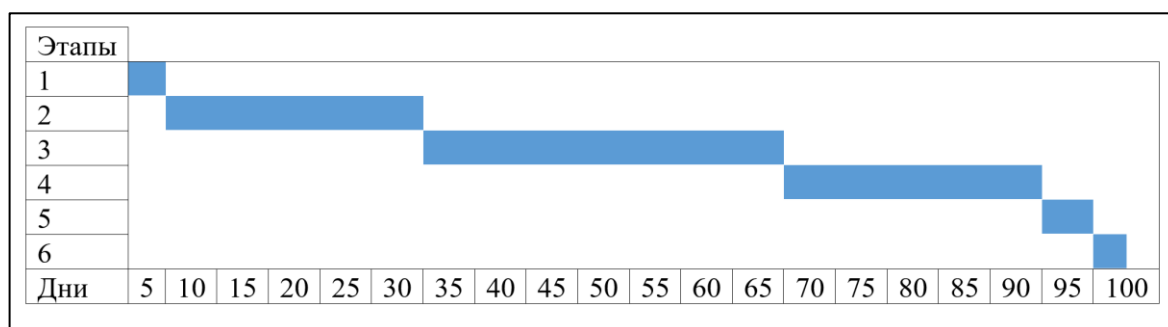


Рисунок 4.3.1 – График проведения работ [Разработано автором]

4.4 Расчёт стоимости проведения работ

с	1 статья «Материалы, покупные изделия и полуфабрикаты + ТЗР (15%) от Σ
е	итого по материалам
б	2 статья «Специальное оборудование» - затрат нет
е	3 статья «Основная заработная плата»
с	4 статья «Дополнительная заработная плата» 20-30% от основной заработной
т	платы
о	5 статья «Страховые отчисления» - 30% от ФОТ
и	6 статья «Командировочные расходы» - затрат нет
м	7 статья «Контрагентские услуги» - затрат нет
о	8 статья «Накладные расходы» - 250% от основной заработной платы
с	9 статья «Прочие расходы» - затрат нет
т	
ь	

В выпускной квалификационной работе объем затрат на НИР и ОКР был проведен методом калькулирования.

1. Статья «Материалы, покупные изделия и полуфабрикаты» (таблица 4.4.1).

Таблица 4.4.1 – Расчет затрат на материалы, покупные изделия и полуфабрикаты

№ пп	Наименование материалов	Единицы измерения	Количество	Цена за единицу (руб)	Стоимость (руб)
1	2	3	4	5	6
1	Флешка 64Гб	шт	1	455	455
2	Бумага А 4	пачка	1	331	331
3	Картридж для принтера	шт	1	1456	1456
4	Ручка	шт	10	18	54
Итого материалов					2296
Транспортно-заготовительные расходы					655
Итого					2951

2. Статья «Специальное оборудование»

Расходы на специальное оборудование отсутствуют.

3. Статья «Основная заработная плата»

Расчет основной заработной платы представлен в таблице 4.4.2.

Таблица 4.4.2 – Расчет основной заработной платы

№ пп	Наименование этапа	Исполнитель (должность)	Мес. оклад (руб)	Трудоемкость (чел/дни)	Оплата за день (руб)	Оплат а за этап (руб)
1	2	3	4	5	6	7
2	ТЗ	Руководитель	51000	1	2318	2318
		Разработчик	40000	5	1818	9090
3	Исследовательск ий этап	Руководитель	51000	1	2318	2318
		Разработчик	40000	25	1818	45454
4	Проектный этап	Разработчик	40000	35	1818	63636
5	Технологический этап	Разработчик	40000	25	1818	45454
		Руководитель	51000	2	2318	4636
6	Экономический этап	Консультант	37500	2	1704	3409
		Разработчик	40000	5	1818	9090
7	Сдача готового продукта	Руководитель	51000	1	2318	2318
		Консультант	37500	1	1704	1704
		Разработчик	40000	1	1818	1818
Итого						191245

4. Статья «Дополнительная заработная плата»

Дополнительная заработная плата (ДЗП) в среднем составляет 20-30% от суммы основной заработной платы.». Расчет дополнительной заработной платы (ДЗП) приведен в формуле (4.1)

$$\text{ДЗП} = 191250 * 0,25 = 47\,811,25 \text{ руб.} \quad (4.1)$$

Дополнительная заработная плата научного и производственного персонала составляет по проекту 47 812,5 руб.

5. Статья «Страховые отчисления»

Отчисления на социальные нужды составляют 30% от фонда оплаты труда (ФОТ), который состоит из основной и дополнительной заработной платы. Расчёт приведён в формулах (4.2) и (4.3)

$$\text{ФОТ} = \text{ОЗП} + \text{ДЗП} = 191245 + 47\,811,25 = 239\,056,25 \text{ руб.} \quad (4.2)$$

$$\text{СВ} = \text{ФОТ} * 30\% = 239\,056,25 * 0,3 = 71\,716,875 \text{ руб.} \quad (4.3)$$

6. Статья «Командировочные расходы»

Командировочные расходы отсутствуют.

7. Статья «Контрагентские услуги»

В процессе разработки данного проекта услуги сторонних организаций не использовались.

8. Статья «Накладные расходы»

К накладным расходам относятся расходы на содержание и ремонт зданий, сооружений, оборудования, инвентаря. Это затраты, сопутствующие основному производству, но не связанные с ним напрямую. Расчёт накладных расходов представлен формулой (4.4)

$$НР = ОЗП * 200\% = 191\,245 * 2 = 382\,490 \text{ руб.} \quad (4.4)$$

9. Статья «Прочие расходы»

По статье «прочие расходы» затрат нет.

Полная себестоимость проекта приведена в таблице 4.4.3

Таблица 4.4.3 – Полная себестоимость проекта

№ пп	Номенклатура статей расходов	Затраты (руб)
1	2	3
1	Материалы, покупные изделия и полуфабрикаты (за вычетом отходов)	2951
2	Специальное оборудование для научных (экспериментальных) работ	–
3	Основная заработная плата научного и производственного персонала	191 245
4	Дополнительная заработная плата научного и производственного персонала	47 811,25
5	Страховые взносы в социальные фонды	7116,875
6	Расходы на научные и производственные командировки	–
7	Оплата работ, выполненных сторонними организациями и предприятиями	–
8	Накладные расходы	382 490
9	Прочие прямые расходы	–
Итого		696 214,125

Дальше необходимо рассчитать договорную цену. Расчёт представлен формулой (4.5).

$$\text{Цена договорная} = \text{себестоимость} + \text{прибыль} + \text{НДС} \quad (4.5)$$

Норма прибыли составляет 20-30% от стоимости разработки. Расчёт прибыли представлен формулой (4.6).

$$П = 696\,214,125 * 0,3 = 208\,864,238 \text{ руб.} \quad (4.6)$$

Разработка ведется для коммерческой организации, поэтому данный вид работы облагается налогом на добавленную стоимость (НДС) в размере 20%: Расчёт НДС осуществлён формулой (4.7).

$$\text{НДС} = (C + П) * 20\% = (696\,214,125 + 208\,864,238) * 0,2 = 905\,078,363 \text{ руб.} \quad (4.7)$$

Таким образом, договорная цена будет рассчитана по формуле (4.8):

$$\text{ДЦ} = C + П + \text{НДС} = 696\,214,125 + 208\,864,238 + 905\,078,363 = 1\,810\,156,726 \quad (4.8)$$

4.5 Выводы по четвертой главе

В четвертой главе разработаны основные разделы бизнес-плана проекта, проработан вопрос организации и планирования работ, а также определена договорная цена в 1 810 156,726 руб.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы проведено исследование конкурентных решений, используемых для визуализации данных.

Спроектирована архитектура веб-приложения, а также его основные компоненты и база данных. Созданная архитектура позволяет дальнейшее добавлять различные дополнения к системе. Также выбрана модель жизненного цикла разработки программного модуля.

Реализован основной функционал веб-приложения, налажено взаимодействие между его частями. В результате было создано веб-приложение, благодаря удобному интерфейсу и расширенным инструментам аналитики это приложение может позволить компаниям и частным лицам получать ценную информацию из своих данных. Использование алгоритмов машинного обучения может помочь выявить закономерности, тенденции и корреляции, которые в противном случае могли бы остаться незамеченными, позволяя пользователям принимать обоснованные решения и предпринимать активные шаги для повышения своей производительности. Учитывая растущее значение данных в современном мире, разработка этого веб-приложения является своевременным и ценным вкладом в область анализа данных.

Тестирование разработанной программы показало, что наибольшая временная и ёмкостная сложность алгоритмов является квадратичной, что является оптимальным при использовании небольшого числа данных.

Разработаны основные разделы бизнес-плана проекта, проработан вопрос организации и планирования работ, определена договорная цена.

In the course of the final qualification work, a study of competitive solutions used for data visualization was conducted.

The architecture of the web application is designed, as well as its main components and database. The created architecture allows you to further add various

additions to the system. The model of the software module development life cycle is also selected.

The main functionality of the web application has been implemented, interaction between its parts has been established. As a result, a web application was created, thanks to a user-friendly interface and advanced analytics tools, this application can allow companies and individuals to get valuable information from their data. Using machine learning algorithms can help identify patterns, trends, and correlations that might otherwise go unnoticed, allowing users to make informed decisions and take proactive steps to improve their productivity. Given the growing importance of data in the modern world, the development of this web application is a timely and valuable contribution to the field of data analysis.

Testing of the developed program showed that the greatest time and capacity complexity of the algorithms is quadratic, which is optimal when using a small number of data.

The main sections of the business plan of the project have been developed, the issue of organization and planning of work has been worked out, the contractual price has been determined.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Ember Charts Build Status: [Электронный ресурс] — URL: <https://www.npmjs.com/package/ember-charts> (дата обращения: 14.11.2022).
2. Create beautiful custom charts: [Электронный ресурс] — URL: <https://www.chartblocks.io/> (дата обращения: 20.12.2022).
3. BuliAn: [Электронный ресурс] — URL: <https://tdenzl-bulian-bulian-ifeiih.streamlit.app/> (дата обращения: 25.12.2022).
4. Streamlit documentation: [Электронный ресурс] — URL: <https://docs.streamlit.io/> (дата обращения: 10.01.2023).
5. В чем заключается визуализация данных: [Электронный ресурс] — URL: <https://aws.amazon.com/ru/what-is/data-visualization/> (дата обращения: 15.01.2023).
6. Что такое визуализация данных: [Электронный ресурс] — URL: <https://www.oracle.com/cis/business-analytics/what-is-data-visualization/> (дата обращения: 16.01.2023).
7. Что такое визуализация данных: какая она бывает и не бывает [Электронный ресурс] — URL: <https://revealthedata.com/blog/all/chto-takoe-vizualizaciya-dannyh-kakaya-ona-byvaet-i-ne-byvaet/> (дата обращения: 21.01.2023).
8. Сегментация изображений: [Электронный ресурс] — URL: <http://neerc.ifmo.ru/wiki/index.php> (дата обращения: 03.02.2023).
9. Базовые принципы машинного обучения на примере линейной регрессии: [Электронный ресурс] — URL: <https://habr.com/ru/companies/ods/articles/322076/> (дата обращения: 14.02.2023).
10. Что такое линейная регрессия?: [Электронный ресурс] — URL: <https://sysblok.ru/glossary/chto-takoe-linejnaja-regressija/> (дата обращения: 23.02.2023).
11. Классическое машинное обучение: задачи классификации, обобщения, кластеризации данных: [Электронный ресурс] — URL:

<https://evergreens.com.ua/ru/articles/classical-machine-learning.html> (дата обращения: 03.03.2023).

12. Обучение без учителя: 4 метода кластеризации данных на Python: [Электронный ресурс] — URL: <https://proglab.io/p/unsupervised-ml-with-python> (дата обращения: 05.03.2023).

13. machine learning: [Электронный ресурс] — URL: <https://www.techtarget.com/searchenterpriseai/definition/machine-learning-ML> (дата обращения: 16.03.2023).

14. Модели жизненного цикла, принципы и методологии разработки программного обеспечения (ПО): [Электронный ресурс] — URL: <https://evergreens.com.ua/ru/articles/software-development-metodologies.html> (дата обращения: 21.03.2023).

15. Функциональная архитектура цифровых продуктов: [Электронный ресурс] — URL: <https://sherer.pro/blog/funkcionalnaja-arhitektura-cifrovyyh-produktov-chast-2/> (дата обращения: 29.03.2023).

16. What is Client-Server Architecture? Everything You Should Know: [Электронный ресурс] — URL: <https://www.simplilearn.com/what-is-client-server-architecture-article> (дата обращения: 06.04.2023).

17. Tornado: [Электронный ресурс] — URL: <https://www.tornadoweb.org/en/stable/> (дата обращения: 10.04.2023).

18. Nginx: [Электронный ресурс] — URL: <https://nginx.org/ru/> (дата обращения: 18.04.2023).

19. Руководство по применению Streamlit с помощью Python: [Электронный ресурс] — URL: <https://biconsult.ru/products/rukovodstvo-po-primeneniyu-streamlit-s-pomoshchyu-python> (дата обращения: 20.04.2023).

20. Deploy Streamlit app on Nginx: [Электронный ресурс] — URL: <https://ngbala6.medium.com/deploy-streamlit-app-on-nginx-cfa327106050> (дата обращения: 25.04.2023).

21. PostgreSQL: [Электронный ресурс] — URL: <https://www.postgresql.org/> (дата обращения: 28.04.2023).

22. Методические рекомендации по выполнению организационно-экономической части выпускных квалификационных работ [Электронный ресурс]: метод. указания / Т. Ю. Гавриленко, О. В. Григоренко, Е. К. Ткаченко. — М.: РТУ МИРЭА, 2019. — Электрон. опт. диск (ISO)
23. Экономика предприятия [Электронный ресурс]: учебно-методическое пособие / И.А. Назарова, А.С. Вихрова. – М.: РТУ МИРЭА, 2021. – Электрон. опт. диск (ISO). – 71 с.
24. Григоренко О.В., Садовнича И.О., Мыльникова А. Экономика предприятия и управление организацией М.: РУСАЙНС, 2017-235с.
25. ГОСТ 7.32-2017 МЕЖГОСУДАРСТВЕННЫЙ СТАНДАРТ, Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления
26. ГОСТ Р ИСО 15704-2008 Требования к стандартным архитектурам и методологиям предприятия
27. Порядок проведения государственной итоговой аттестации по образовательным программам высшего образования – программам бакалавриата, программам специалитета и программам магистратуры СМКО МИРЭА 7.5.1/03.П.30-19
28. Положение о выпускной квалификационной работе студентов, обучающихся по образовательным программам подготовки бакалавров СМКО МИРЭА 7.5.1/03.П.67-19
29. Федеральный государственный образовательный стандарт высшего образования - бакалавриат по направлению подготовки 09.03.04 Программная инженерия (ФГОС ВО 3++)
30. Методические указания по подготовке выпускных квалификационных работ по направлениям и специальности профессиональной подготовки высшего образования 09.03.04 (Программная инженерия), 09.04.04 (Программная инженерия)

ПРИЛОЖЕНИЕ А

Листинг А.1 – Файл «1_Regression.py»

```
import streamlit as st
import pandas as pd
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt

def linear_regression(file):
    # Загрузка данных из файла csv
    data = pd.read_csv(file)

    feature = st.selectbox('Выберите столбец для предсказания', data.columns)
    st.write(feature)
    data=pd.DataFrame({'x':[1,2,3,4,5], 'y':[2,2,2,3,3]})
    # Разделение данных на признаки и целевую переменную
    X = data.iloc[:, :-1].values
    y = data.iloc[:, -1].values

    # Обучение модели линейной регрессии
    model = LinearRegression()
    model.fit(X, y)

    # Визуализация результатов линейной регрессии
    fig, ax = plt.subplots()
    ax.scatter(X, y)
    ax.plot(X, model.predict(X), color='red')
    ax.set_xlabel('Признаки')
    ax.set_ylabel('Целевая переменная')
    ax.set_title('Линейная регрессия')
    st.pyplot(fig)

# Заголовок приложения
st.title('Линейная регрессия')

# Загрузка файла csv
file = st.file_uploader('Загрузите файл csv', type='csv')

# Проверка наличия файла
if file is not None:
    # Вызов функции linear_regression
    linear_regression(file)
```

ПРИЛОЖЕНИЕ В

Листинг В.1 – Файл «2_Clsassification.py»

```
import streamlit as st
import pandas as pd
from matplotlib import pyplot as plt
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from sklearn.linear_model import LogisticRegression, LogisticRegressionCV
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier, BaseEnsemble, GradientBoostingClassifier
from sklearn.svm import SVC, LinearSVC
import time
from xgboost import XGBClassifier
from sklearn.model_selection import train_test_split
import sys
@st.cache
def get_dataset(dataset_name):
    df = dataset_name.iloc[:,1:]
    X=df.iloc[:, :-1] #drop(['Species'], axis=1)
    y= df.iloc[:, -1] #df['Species']
    return X,y
def parameter_changes(clf_name):
    params = dict()
    if clf_name=="KNN":
        K = st.sidebar.slider("K",1,15)
        params["K"]=K
    elif clf_name=="Logistic Regression":
        penalty = st.sidebar.selectbox("penalty", ("l2", "none"))
        params["penalty"]=penalty
    elif clf_name=="XGBoost":
        learning_rate = st.sidebar.slider("learning_rate",0.01,10.0)
        params["learning_rate"]=learning_rate
    elif clf_name=="Random Forests":
        n_estimators = st.sidebar.slider("n_estimators",0,1500,100)
        params["n_estimators"]=n_estimators
    elif clf_name=="Decision Tree":
        criterion=st.sidebar.selectbox("criterion", ("gini", "entropy"))
        max_depth = st.sidebar.slider("max_depth",1,15)
        params["max_depth"]=max_depth
        params["criterion"]=criterion
```

Продолжение листинга В.1

```

elif clf_name=="Support Vector Machines":
    C = st.sidebar.slider("C",1,15)
    kernel=st.sidebar.selectbox("kernel", ("linear",
"poly", "rbf", "sigmoid", "precomputed"))
    params["C"]=C
    params["kernel"]=kernel
    return params
def get_algorithm(algorithm,params):
    if algorithm=="KNN":
        alg =
KNeighborsClassifier(n_neighbors=params["K"])
    elif algorithm=="Logistic Regression":
        alg =
LogisticRegression(penalty=params["penalty"])
    elif algorithm=="XGBoost":
        alg =
XGBClassifier(learning_rate=params["learning_rate"])
    elif algorithm=="Random Forests":
        alg =
RandomForestClassifier(n_estimators=params["n_estimators"])
    elif algorithm=="Decision Tree":
        alg =
DecisionTreeClassifier(max_depth=params["max_depth"],criterion
=params["criterion"])
    elif algorithm=="Support Vector Machines":
        alg = SVC(C=params["C"],kernel=params["kernel"])
    return alg

titles = st.container()
dataset = st.container()
training = st.container()
prediction = st.container()
visualisation = st.container()

with titles:
    st.title("Классификация")
    # image = Image.open('sklearn.png')
    # st.image(image, caption='Sklearn Logo')
    st.write("""

""")
with dataset:
    # st.title("Dataset Information")
    #dataset_name      =      st.sidebar.selectbox("Select
Dataset", ("Iris", "Breast Cancer", "Wine Dataset"))
    #Check out all available dataset: https://scikit-learn.org/stable/datasets/toy\_dataset.html
    uploaded_files = st.file_uploader(
        "Choose a CSV file")

    if uploaded_files !=None:
        dataset_name=pd.read_csv(uploaded_files)

```

Продолжение листинга В.1

```

        st.write(dataset_name)

        st.sidebar.subheader("{0}                               данные
выбраны!".format(uploaded_files.name))
        st.write("{0}                               данные
выбраны!".format(uploaded_files.name))
        algorithm_name = st.sidebar.selectbox("Выберете
алгоритм", ("Logistic                               Regression", "Random
Forests", "XGBoost", "Decision Tree", "KNN", "Support Vector
Machines"))
        st.sidebar.subheader("{0}                               алгоритм
выбран!".format(algorithm_name))
        st.write("{0}                               алгоритм
выбран!".format(algorithm_name))

        X,y = get_dataset(dataset_name)

        st.sidebar.subheader("Упорядочить параметры!")

        params = parameter_changes(algorithm_name)

        st.write("Форма набора данных предикатора:
",X.shape)
        st.write("Количество классов: ",len(np.unique(y)))
        st.write("Давайте посмотрим на данные
предиктора!")
        st.write(X[:5])

        st.write("Давайте посмотрим на данные, которые мы
будем прогнозировать!")
        st.write(y[:5])

    with training:
        st.title("Давайте обучим модель!")
        alg_name= get_algorithm(algorithm_name,params)
        X_train, X_test, y_train, y_test = train_test_split(X,
y, test_size=0.2, random_state=100)

        start_time = time.time()
        alg_name.fit(X_train,y_train)
        elapsed_time = time.time() - start_time
        st.write("Время обучения модели:      {}
секунд".format(elapsed_time))

    with prediction:
        st.title("Прогнозирование модели")
        y_pred= alg_name.predict(X_test)
        acc = accuracy_score(y_test, y_pred)

        st.write(f'Классификатор = {algorithm_name}')
        st.write(f'Точность =', acc)
try:

```


Продолжение листинга В.1

```
with visualisation:
    st.title("Визуализация данных")
    pca = PCA(2)
    X_projected = pca.fit_transform(X)

    x1 = X_projected[:, 0]
    x2 = X_projected[:, 1]

    fig = plt.figure()
    plt.scatter(x1, x2,
                alpha=0.8)

    plt.xlabel('Principal Component 1')
    plt.ylabel('Principal Component 2')
    plt.colorbar()
    st.pyplot(fig)
except Exception:
    e = sys.exc_info()[1]
    st.write(e.args[0])
    # print(e.args[0])
```

ПРИЛОЖЕНИЕ С

Листинг С.1 – Файл «3_Clustering.py»

```
import streamlit as st
import pandas as pd
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt

st.title('Кластеризация')

# Загрузка файла .csv
file = st.file_uploader('Загрузите файл .csv', type='csv')

if file is not None:
    # Чтение данных из файла
    data = pd.read_csv(file)

    # Выбор колонок для кластеризации
    features = st.multiselect('Выберите колонки для кластеризации', data.columns)

    # Создание модели кластеризации
    X = data[features]
    n=st.slider('n_clusters', 1, 10, 3)
    if n==0:
        n=1
    model = KMeans(n_clusters=n)
    model.fit(X)

    # Вывод графика кластеризации
    fig, ax = plt.subplots()
    ax.scatter(X.iloc[:, 0], X.iloc[:, 1], c=model.labels_)
    ax.set_xlabel(features[0])
    ax.set_ylabel(features[1])
    st.pyplot(fig)

    # Вывод информации о кластерах
    st.write('Информация о кластерах:')
    for i in range(model.n_clusters):
        cluster_data = data[model.labels_ == i]
        st.write(f'Кластер {i}:')
        st.write(cluster_data.describe())
```

ПРИЛОЖЕНИЕ D



Рисунок D.1 – Сертификат о публикации статьи