*[1 слайд] Представление*

Уважаемый председатель и члены государственной экзаменационной комиссии, вашему вниманию представляется защита выпускной квалификационной работы студента Миронова Дмитрия Сергеевича группы ИКБО-20-19 по направлению подготовки Программная инженерия на тему «Интерактивное веб-приложение для анализа и визуализации данных с использованием алгоритмов машинного обучения». Руководитель работы кандидат технических наук, доцент Плотников Сергей Борисович, консультант по экономической части старший преподаватель Белоусова Ирина Викторовна.

*[2 слайд] Цели и задачи*

Главной целью работы является реализацией алгоритмов машинного обучения в интерактивном веб-приложении.

Главными задачами является правильный выбор алгоритмов и разработка архитектуры системы.

*[3 слайд] Актуальность и новизна*

Использование алгоритмов машинного обучения может помочь выявить закономерности, тенденции и корреляции, которые в противном случае могли бы остаться незамеченными, и это определяет актуальность работы, а новизна связана с решением задачей интеграции алгоритмов машинного обучения

*[4 слайд] Анализ существующих веб-приложений*

В ходе проведенного анализа веб-приложений было выявлена сложность их использования обычным пользователем, за основу было взято веб-приложение, представленное на рисунке 3, для лучшего восприятия информации, тем самым был проведен анализ методов визуализации (след слайд)

*[5 слайд] Анализ методов визуализации*

В ходе проведенного анализа мы отобрали основные методы визуализации. На рисунке представлена стандартная визуализация 2D/3D-изображений, которая включает в себя создание графиков, столбцов и гистограмм подходящее для поставленной задачи связанной с алгоритмами машинного обучения (след слайд)

*[6 слайд] Анализ алгоритмов машинного обучения*

Проанализировав различные методы были выбраны такие методы как регрессия, представленная на рисунке 5 и классификация, представленная на рисунке 6 и кластеризация.

*[7 слайд] Выбор средств разработки*

При выборе средств разработки язык программирования был выбран Python поскольку для него разработано огромное количество библиотек, фреймворк streamlit поскольку он был специально разработан для алгоритмов машинного обучения и база данных PostgreSQL.

*[8 слайд] Проектирование адаптированной модели жизненного цикла*

Существует несколько преимуществ использования итеративной модели жизненного цикла для разработки веб-приложений:

Гибкость.

Более быстрая обратная связь. Итеративный подход позволяет получать отзывы о проекте намного быстрее, чем другие подходы к разработке.

Улучшение качества. Итеративный подход позволяет проводить непрерывное тестирование и обеспечение качества.

*[9 слайд] Разработка структурной схемы веб-приложения*

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.

*[10 слайд] Разработка архитектуры серверной части веб-приложения*

Для разработки серверной части был выбран сервер Tornado поскольку он является частью фреймворка Streamlit и подерживает возможность использования ВЕБСОКЕТОВ, а NGINX был использован в качестве обратного прокси-сервера позволяя исходному серверу не раскрывать свои IP адреса клиентам и для улучшения безопасности был настроен SSL — криптографический протокол, который подразумевает более безопасную связь.

*[11 слайд] Разработка серверной части веб-приложения*

Для того чтобы использовать веб-сокеты, необходимо настроить Nginx и настроить защиту трафика при помощи SSL. Для этого нужно отредактировать файл конфигурации Nginx (/etc/nginx/nginx.conf). Этот блок определяет, какие запросы нужно перенаправлять на сервер Tornado и какие заголовки нужно добавлять для поддержки веб-сокетов.

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

Определяем класс MainHandler, обрабатывает GET-запросы.

Определяем класс WebSocketHandler, обрабатывает веб-сокеты. Методы вызываются при открытии соединения, при получении сообщения и метод при закрытии соединения.

*[12 слайд] Разработка архитектуры клиентской части веб-приложения*

При разработке архитектуры клиентской части, необходимо было разработать простую логику, интуитивно понятную пользователю.

*[13 слайд] Разработка клиентской части*

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.

*[14 слайд] Регрессия*

Пользователь выбирает столбец, по которому хочет увидеть график предсказаний, в любой момент пользователь сможет изменить выбранный столбец и ему представится новый график.

*[15 слайд] Классификация*

Выбрав классификацию, пользователю на боковой панели предоставляется возможность выбрать подвиды классификации при которых выводятся различные результаты, так же пользователю выводится скорость работы и количество классов и точность модели.

*[16 слайд] Классификация*

Так же видим вывод графиков

*[17 слайд] Кластеризация*

При кластеризации пользователю необходимо выбрать столбцы и количество кластеров, которые будут присутствовать на графике.

*[18 слайд] Расчет сложности*

Сложность алгоритмов зависит от загружаемых пользователем фалов, она зависит от количества строк в массиве данных.