РАСЧЕТНО-ГРАФИЧЕСКАЯ РАБОТА ПО ДИСЦИПЛИНЕ «МАШИННОЕ ОБУЧЕНИЕ И БОЛЬШИЕ ДАННЫЕ». Тема: «Разработка Web-приложения (дашборда) для инференса (вывода) моделей ML и анализа данных»

Вывод машинного обучения (Machine learning inference)[1][2] включает в себя применение модели машинного обучения (Machine Learning, ML) к набору данных и создание выходных данных или «прогноза». Обычно модель ML представляет собой программный код, реализующий математический алгоритм. Процесс вывода ML развертывает этот код в производственной среде, что позволяет генерировать прогнозы для входных данных, предоставленных реальными конечными пользователями.

Жизненный цикл ML состоит из двух основных частей:

1. **Фаза обучения** — включает в себя создание модели ML, ее обучение путем запуска модели на примерах помеченных данных, а затем тестирование и проверку модели путем ее запуска на невидимых примерах.
2. **Выводы ML** — включают в себя использование модели на реальных данных для получения действенных результатов. На этом этапе система вывода принимает входные данные от конечных пользователей, обрабатывает данные, передает их в модель ML и передает выходные данные обратно пользователям.

Аналитические информационные панели (дашборд) — это отличный способ для специалистов по данным передавать информацию компаниям, но их создание часто может быть дорогостоящим и трудоемким. Streamlit — это простая в использовании библиотека на основе Python, которая упрощает этот процесс.

Streamlit — это библиотека на основе Python, которая позволяет специалистам по данным легко создавать бесплатные приложения ML. Streamlit позволяет отображать описательный текст и выходные данные модели ML, визуализировать данные и производительность модели ML, а также изменять входные данные модели ML через пользовательский интерфейс с помощью боковых панелей.

ЗАДАНИЕ.

Используя возможности библиотеки Streamlit, создайте веб-приложение (дашборд), основной целью которого является инференс (вывод) моделей машинного обучения и представление результатов анализа данных с различными вариантами их визуализации.

1. Выберите один из двух наборов данных, соответствующих вашему варианту (для регрессии или классификации). Среди всех моделей ML, построенных в ходе выполнения лабораторных работ, выберите шесть лучших моделей на основе значений коэффициента детерминации R^2 для регрессии и F1 для классификации:

[1] Machine Learning Inference. – URL: https://www.gigaspaces.com/data-terms/machine-learning-inference

[2] Machine Learning Inference – All You Need to Know. – URL: https://pareto.ai/blog/machine-learning-inference

Модель ML1: классическая модель обучения с учителем (например, линейная и полиномиальная регрессия с регуляризаторами и без них, деревья принятия решений, kNN, SVM, логистическая регрессия, байесовский классификатор).

Модель ML2: ансамблевая модель (бустинг).

Модель ML3: модель на основе библиотеки продвинутого градиентного бустинга (CatBoost и др.).

Модель ML4: ансамблевая модель (бэггинг).

Модель ML5: ансамблевая модель (стэкинг).

Модель ML6: глубокая полносвязная нейронная сеть.

2. С**ериализуйте (сохраните) выбранные модели ML на жесткий диск**. При *сериализации моделей ML,* использующих TensorFlow, XGBoost и CatBoost, используйте встроенные инструменты. При *сериализации моделей ML из библиотеки Sklearn (или собственных моделей ML)* используйте стандартный модуль **pickle**[3] в Python.

3. С помощью библиотек Streamlit (можно Dash) реализуйте Web-приложение (дашборд).

Структура Web-приложения (дашборд):

✶ **Web-страница 1 с информацией о разработчике моделей ML** (ФИО, номер учебной группы и цветная фотография, тема РГР);

✶ **Web-страница 2 с информацией о наборе данных** (описание предметной области датасета и его признаков, особенности предобработки данных и EDA);

✶ **Web-страница 3 с визуализациями зависимостей** в наборе данных (визуализации Matplotlib, Seaborn, минимум 4 различных вида визуализаций);

✶ **Web-страница 4,** с помощью которой можно получить **предсказание соответствующей модели ML** (см. п. 1). Для этого необходимо разработать интерфейс с использованием статичных и интерактивных элементов управления (виджетов) Streamlit:

– загрузка файла в формате *.csv;

– ввод данных в соответствии с используемым набором данных и их валидацию; при необходимости указать единицу измерения вводимого значения признака;

– вывод результата в понятной интерпретации для конечного пользователя (например, если необходимо получить стоимость драгоценного камня, указать результат в денежном формате с указанием нужной валюты и т.д.).

[3] Модуль **pickle** реализует алгоритм сериализации и десериализации объектов Python.

РЕКОМЕНДАЦИИ К ОФОРМЛЕНИЮ ОТЧЕТА РГР:

1. Выполнить форматирование текста отчета в соответствии с требования к оформлению текста в отчете курсового проекта.

2. В **отчете** описать ход работы над заданием РГР. Ниже представлена структура отчета РГР.

Титульный лист.

Содержание. Создать автоматизированное оглавление отчета РГР.

Введение (1 стр.). Отразить актуальность темы РГР и применяемый для ее реализации стек технологий, конкретные изученный инструменты (библиотеки), (например, Sklearn, TensorFlow и др.) и отметить их роль в решении задач ML. Сформулировать цель и задачи РГР (3 задачи, которые будут решены в ходе выполнении работы).

Автоматизированное оглавление. Разработайте автоматизированное оглавление.

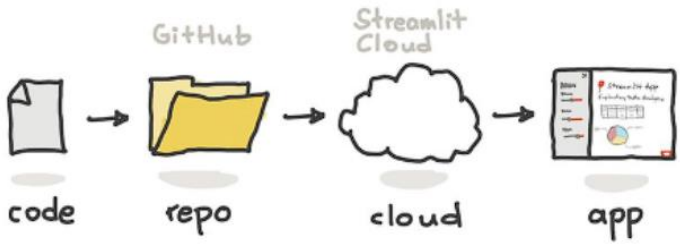
Основная часть. Включает в себя как минимум два пункта, в которых выделены подпункты (1.1, 1.2 и т. д.):

- Пункт 1. Описываются выбранные модели, процесс их сериализации (краткое описание самих моделей и используемых инструментов сериализации);
- Пункт 2. Описывается последовательно процесс разработки дашборда, приводятся небольшие фрагменты исходного кода. В Пункте 2 сформировать ссылку на GitHub, куда предварительно размещены все исходные файлы РГР.

Таким образом, должен быть репозиторий на GitHub, в котором лежит ***readme*** с подробным описанием проекта РГР.

В качестве результатов необходимо указать две ссылки:

- ссылка на Ваш открытый GitHub-репозиторий,
- ссылка на Web-сервис (например, выложенный на Streamlit Cloud, https://streamlit.io/cloud)



Пункт 3.

3.1. Демонстрация прогнозирования на нескольких примерах данных, включая:

- Данные с выбросами: Примеры данных, содержащие редкие случаи или аномалии, которые могут быть интересны для анализа. Предоставьте скриншоты результатов прогнозирования для этих случаев.
- Корректные данные: Примеры данных без искажений или выбросов, которые представляют собой стандартные случаи. Включите скриншоты прогнозов на этих данных.

3.2. Каждый пример должен сопровождаться объяснением: Краткое описание того, почему выбранные данные были использованы и как они влияют на результаты прогнозирования.

Заключение (1 – 1.5 стр.). Описать, что было сделано в рамках РГР (выводы).

Список использованных источников (5 источников). Список использованных ссылок на Web-ресурсы и литературы выполняется в виде нумерованного списка в соответствии с действующим **ГОСТом Р 7.0.100–2018** к библиографическому описанию На каждый источник в тексте отчета РГР должна быть ссылка, например, вида [1] в конце приложения, где 1 – номер источника. Количество ссылок в библиографическом списке должно быть не менее 3.

Приложения:
Приложение А содержит скрины Web-страниц приложения дашбрда в моменте исполнения.
Приложение Б содержит исходный код разработанного Web приложения.

3. Файл отчета сохранить под именем **РГР_Фамилия студента_Группа.docx** (например, **РГР_Иванов_МО-221.docx**).

4. После выполнения и оформления отчета РГР необходимо:
4.1. Распечатать титульный лист.
4.2. В строке **выполнил** студент должен поставить свою подпись.
4.3. В строке **принял** преподаватель должен поставить свою подпись.
4.4. Произвести сканирование или фотографирование титульного листа.
4.5. Заменить титульный лист без подписей на титульный лист с подписями в файле отчета РГР.
4. 6. Файл отчета РГР сохранить в формате PDF.

4.7. Выполненную РГР (файл отчета в формате PDF) разместить в своем личном кабинете на сайте ОмГТУ.

ПОДГОТОВКА РАБОЧЕЙ СРЕДЫ

Streamlit – это библиотека Python с открытым исходным кодом, которая очень популярна среди разработчиков в области ML. Ее преимущество перед другими инструментами в том, что для работы с ней не требуются дополнительные знания в области Web-технологий. Достаточно уверенного владения языком программирования Python.

Подготовка среды для работы (следующие команды выполняем в терминале).

- Создать в выбранной для проекта папке, виртуальное окружение:

```
$ python3 -m venv venv
```

- Активировать окружение:

```
$ source venv/bin/activate
```

- Обновить пакетный менеджер pip:

```
$ pip install --upgrade pip
```

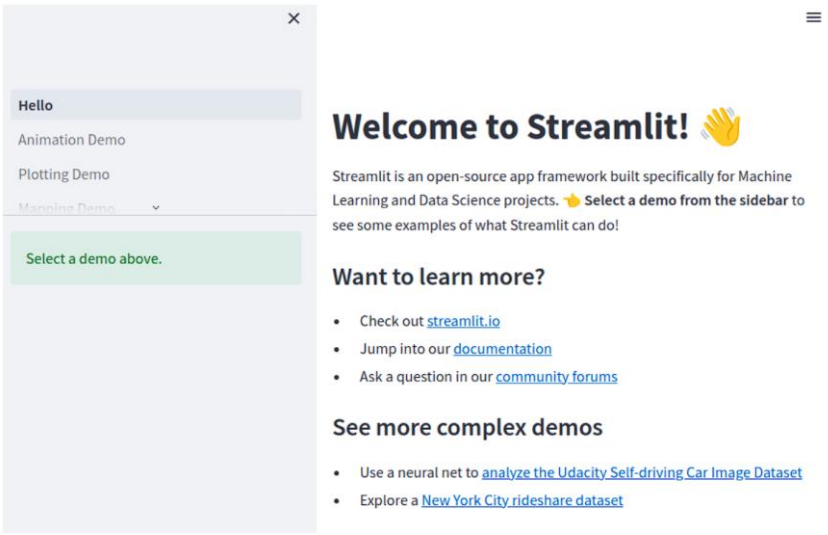
- Установить фреймворк Streamlit:

```
$ pip install streamlit
```

- Чтобы убедиться в корректности установки, запустим демо библиотеки:

```
$ streamlit hello
```

Если все отработало верно, то откроется страница демонстрационного приложения, в котором можно познакомиться с возможностями, предоставляемыми Streamlit:



ПОЛЕЗНЫЕ ИСТОЧНИКИ ПО РГР

Материалы по Streamlit

1. Документация Streamlit. – URL: https://streamlit.io
2. Как создать свое первое приложение на Streamlit. – URL: https://docs.streamlit.io/library/get-started/create-an-app
3. Начало работы с библиотекой Streamlit. Get started. – URL: https://docs.streamlit.io/en/stable/getting_started.html#create-your-firststreamlit-app
4. Описание виджетов библиотеки streamlit. Display interactive widgets. – URL: https://docs.streamlit.io/en/stable/api.html#display-interactive-widgets
5. Описание графических возможностей библиотеки Streamlit. Display charts. – URL: https://docs.streamlit.io/en/stable/api.html#displaycharts
6. Основные функции фреймворка с примерами кода. – URL: https://docs.streamlit.io/library/api-reference
7. Развертывание моделей машинного обучения. Часть первая. Размещаем Web-приложение в облачной платформе Heroku. – URL: https://habr.com/ru/articles/664076/
8. Demo Your Model With Streamlit. – URL: https://towardsdatascience.com/demo-your-model-with-streamlit-a76011467dfb/
9. How to Quickly Deploy Machine Learning Models with Streamlit. – URL: https://machinelearningmastery.com/how-to-quickly-deploy-machine-learning-models-streamlit/
10. Python Serialization. – URL: https://pythongeeks.org/python-serialization/
11. Учебный онлайн курс «Streamlite for Data Science course». – URL: https://datascience-course.streamlit.app/

Streamlit for Data Science COURSE

Примеры по Streamlit

1. Streamlit. Поиск кратчайшего пути. – URL: https://habr.com/ru/articles/568836/
2. Streamlit Tutorial: A Beginner's Guide to Building Machine Learning-Based Web Applications in Python. – URL: https://builtin.com/machine-learning/streamlit-tutorial

3. Web-приложение для предсказания цены на недвижимость на Python и Streamlit. – URL: https://www.youtube.com/watch?v=ZjxPMwL552s&ab_channel=miracl6

4. Галерея приложений реализованных на Streamlit (со ссылками на репозитории проектов, очень полезно!) – URL: https://streamlit.io/gallery

5. Как написать веб-приложение для демонстрации data science-проекта на Python. – URL: https://blog.skillfactory.ru/kak-napisat-veb-prilozhenie-dlya-demonstratsii-data-science-proekta-na-python/

6. Учебный проект на Python: интерфейс в 40 строк кода. Ч. 2. – URL: https://habr.com/ru/company/skillfactory/blog/509340/