МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
Российской Федерации
федеральное государственное автономное
образовательное учреждение высшего образования
«Самарский национальный исследовательский университет
имени академика С.П. Королева»

(Самарский университет)

Институт информатики, математики и электроники
Кафедра технической кибернетики

Отчет

по лабораторной работе № 3

«MLOps с использованием ClearML | недельный прогноз погоды»

по дисциплине

«Инженерия данных»

Обучающийся: Фролова А. С.

6231-010402D

Преподаватель: Парингер Р. А.

Самара 2025

1. ***Архитектура:***

Реализован пайплан для создания прогноза максимальной температуры, который поступает в Telegram в виде сообщения. В задании было представлено описание пайплайна. Описание по шагам:

1) Необходимо было сначала собрать данные с API (разрешено использование Open-meteo из первой лабораторной, https://open-meteo.com/). API не имеет ключа, поэтому подходит нам по требованиям. Выбрать целевую переменную - temp_max. А также географию: Москва. Начальная дата должна быть выбрана так, чтобы промежуток составлял минимум 3 года - 2020-01-01, чтобы рассмотреть больше данных и соответствовать требованиям.
2) Поднять сервер ClearML (2.0.2) и настроить agent. После этого создать датасет из полученных данных и загрузить его.
3) Реализовать первичное обучение, чтобы этот task в дальнейшей использовать для HPO.
4) Выполнять HPO, с не менее чем 10 конфигурациями, после чего зарегистрировать лучшую модель.
5) Создать сервис Fast-API (0.122.0), поднять его для дальнейшего использования в n8n. Загрузить лучшую модель и использовать ее для предсказаний.
6) Создать еженедельную отправку сообщения в n8n (1.121.3), через HTTP Request к predict на Fast-API. По шагам: Schedule Trigger запускает workflow, HTTP Request вызывает FastAPI /predict, формируется итоговое сообщение с температурой, результат отправляется в Telegram Bot.

Инструменты и почему именно они:

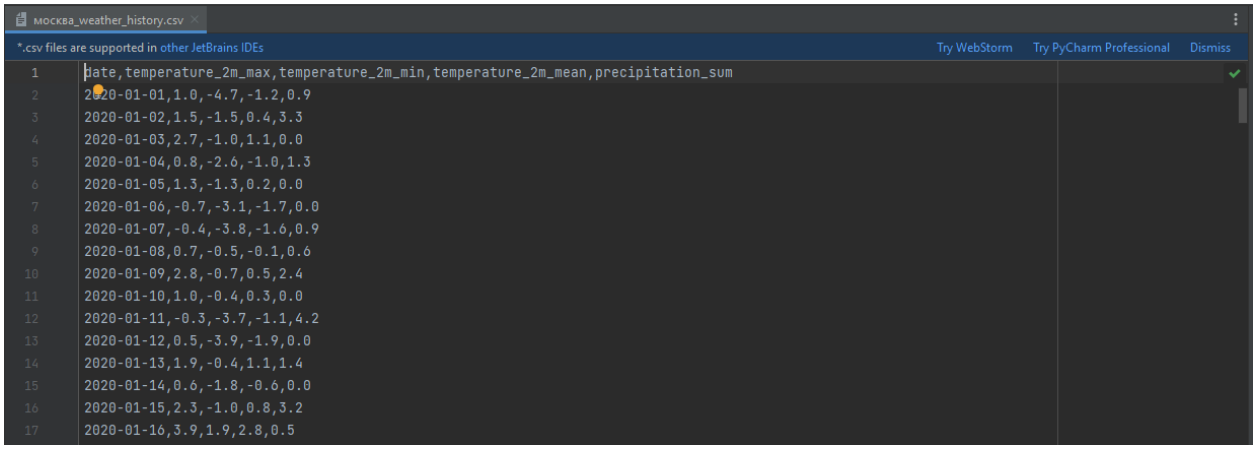
- n8n Workflow (последняя версия, 1.121.3): оркестрация триггеров, работаем с логикой и узлами, а также бинарными данными (https://n8n.io/).
- ClearML (2.0.2): инструмент для проведения экспериментов, обучения моделей, хранения и создания датасетов, а также их использования (https://clear.ml/).
- Telegram Bot API: удобная выдача сообщений, прием и отправка видео.
- Базовые библиотеки для реализации.
- LightGBM (4.6.0) – быстрая работа с признаками и данными.
- libgomp1: необходим для lightbm.

Главная причина – требование для выполнения лабораторной. n8n имеет удобный визуальный стиль и контроль, а также был ранее использован в другой лабораторной. ClearML прост в использовании, особенно если разобраться с активацией агента.

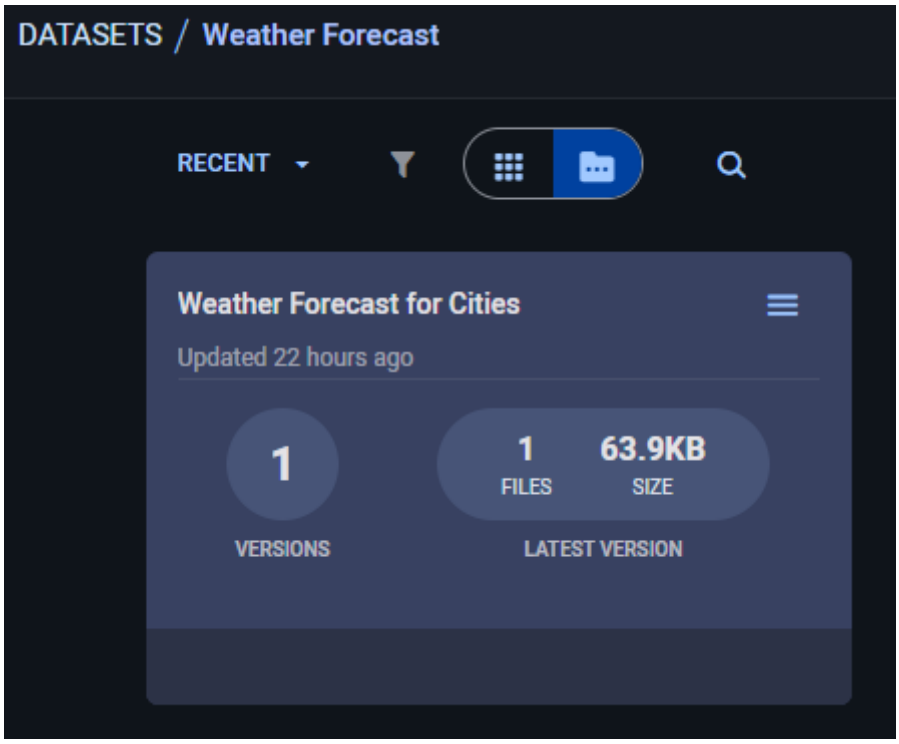
Структура:

- api.py – загружаем данные с Open-Meteo, проверяем, что города есть в списке, сохраняем данные в формате csv;

Скриншот полученного csv:



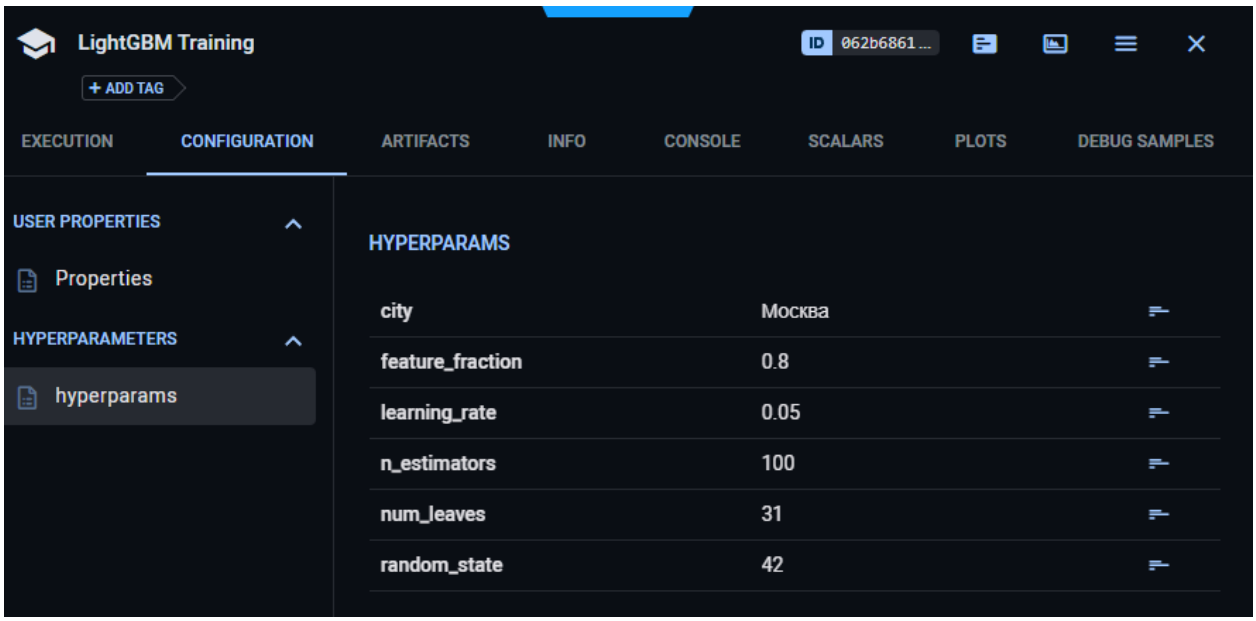
- database.py – добавляем файл в датасет, указав версию, выводим ID, чтобы проще было обращаться в дальнейшем;



Данные с Open-Meteo: https://archive-api.open-meteo.com/v1/archive.

Диапазон дат: 2020-01-01 -:- 2025-11-28.

- train.py – загрузка датасета, генерация признаков (циклические, календарные данные и так далее), обучение модели LightGBM, расчет MAE и RMSE, сохраняем, как .pkl;



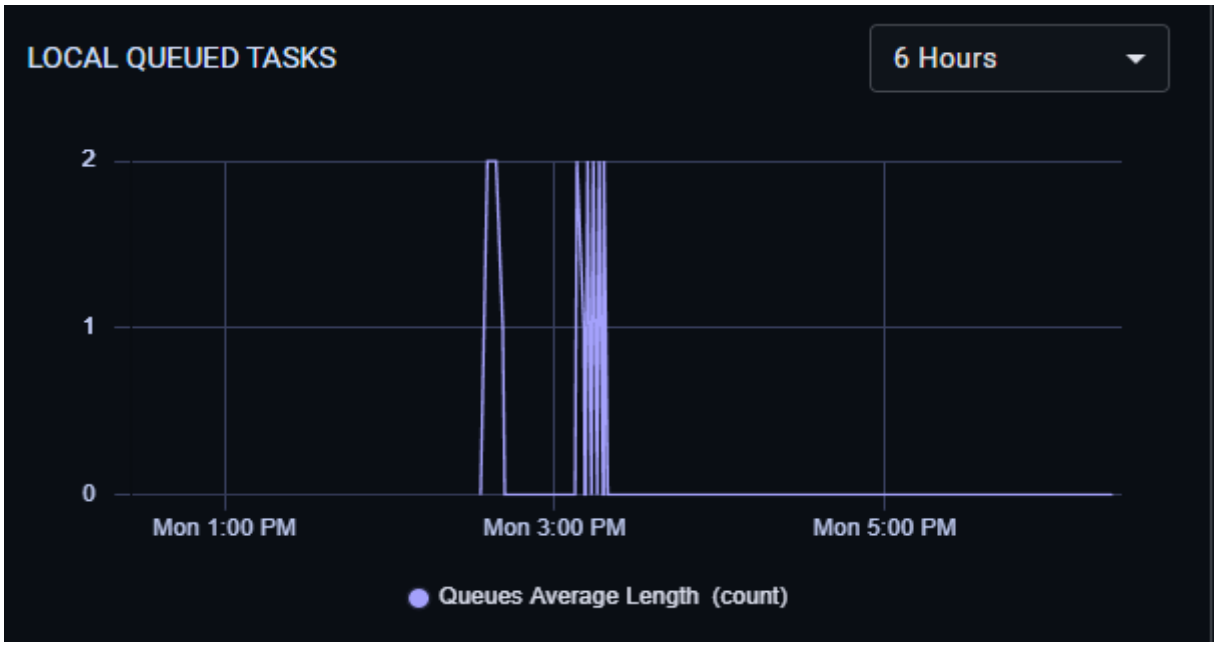
Информация о признаках, такие как: лаги, скользящие средние и так далее, была взята с habr: https://habr.com/ru/companies/skillfactory/articles/842444/. Там есть также ссылки на более подробную информацию и способы применения.

Логи с консоли на ClearML с результатами MAE и RMSE:

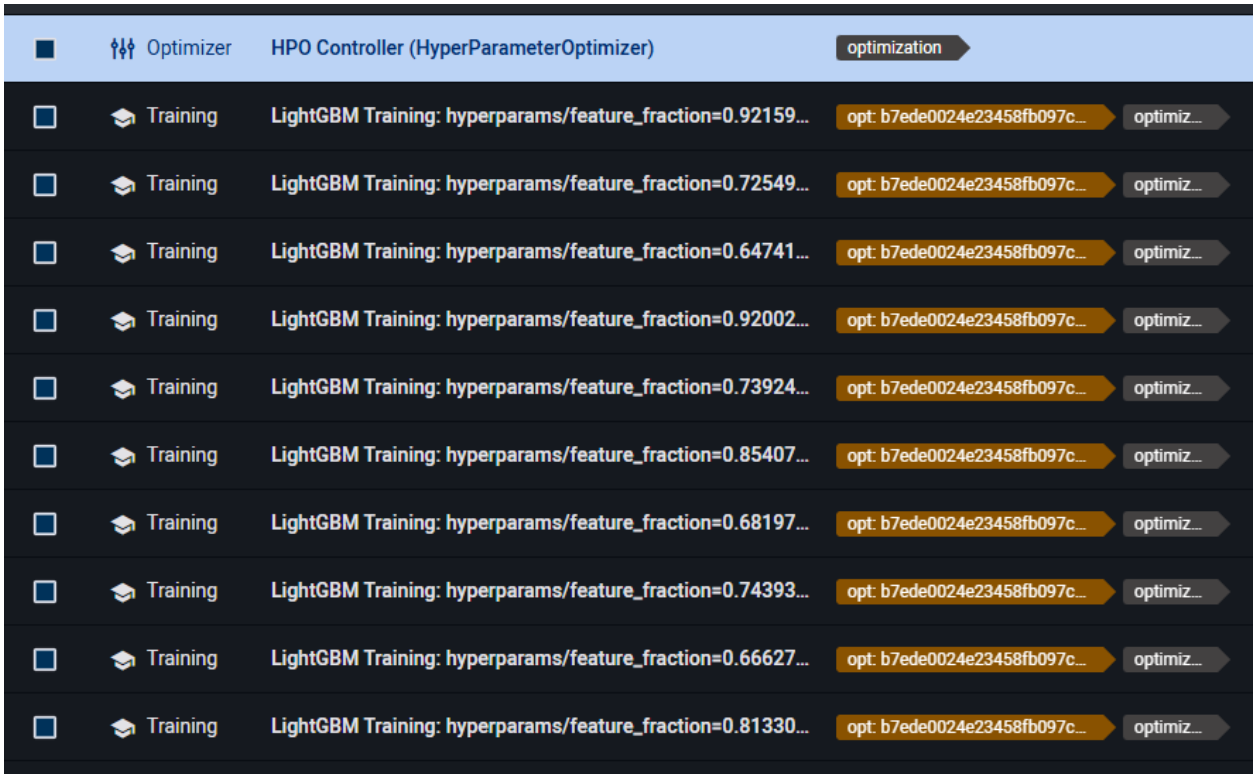
```
[LightGBM] [Info] Total Bins 3175
[LightGBM] [Info] Number of data points in the train set: 1702, number of used features: 16
[LightGBM] [Info] Start training from score 10.883549
MAE: 0.8241, RMSE: 1.0747
Training complete!
```

- hpo.py – проводим поиск лучших гиперпараметров (learning_rate, n_estimators и так далее), запускаем 10 раз train, в конце выдаем ID лучшего эксперимента;

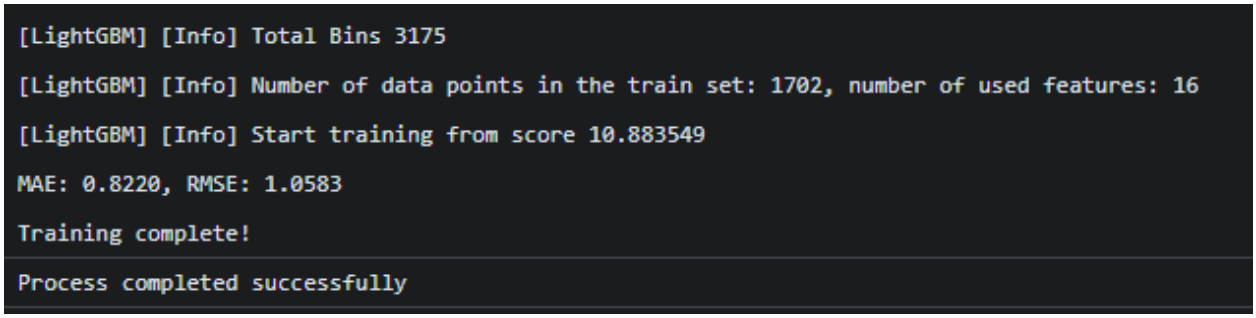
Скриншот созданной очереди для расчета:



Скриншот созданных training и оптимизатора:

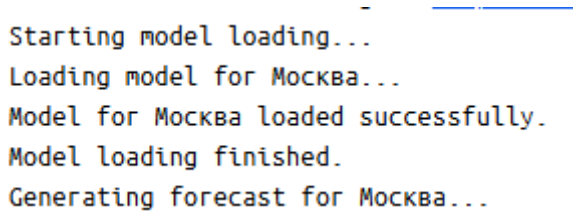


Значения MAE и RMSE для лучшего эксперимента:



- main.py – реализация загрузки лучшей модели, соответствующей лучшему эксперименту, проводим предсказания с учетом модели, создаем возможно для запроса по predict, чтобы обращаться в n8n.

Скриншот с Docker Compose weather-fast-api:



2. ***Качество данных: какие проверки качества данных реализованы и возможные точки сбоя.***

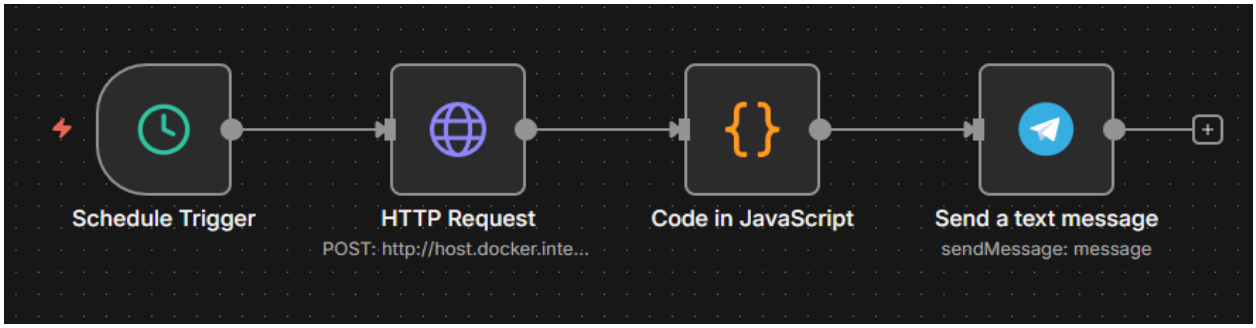
Добавлено несколько проверок данных, обработок точек сбоя и т. д на разных этапах. Приведем их:

- Проверка запроса к Open-Meteo.
- Ошибки в случае сохранения csv.

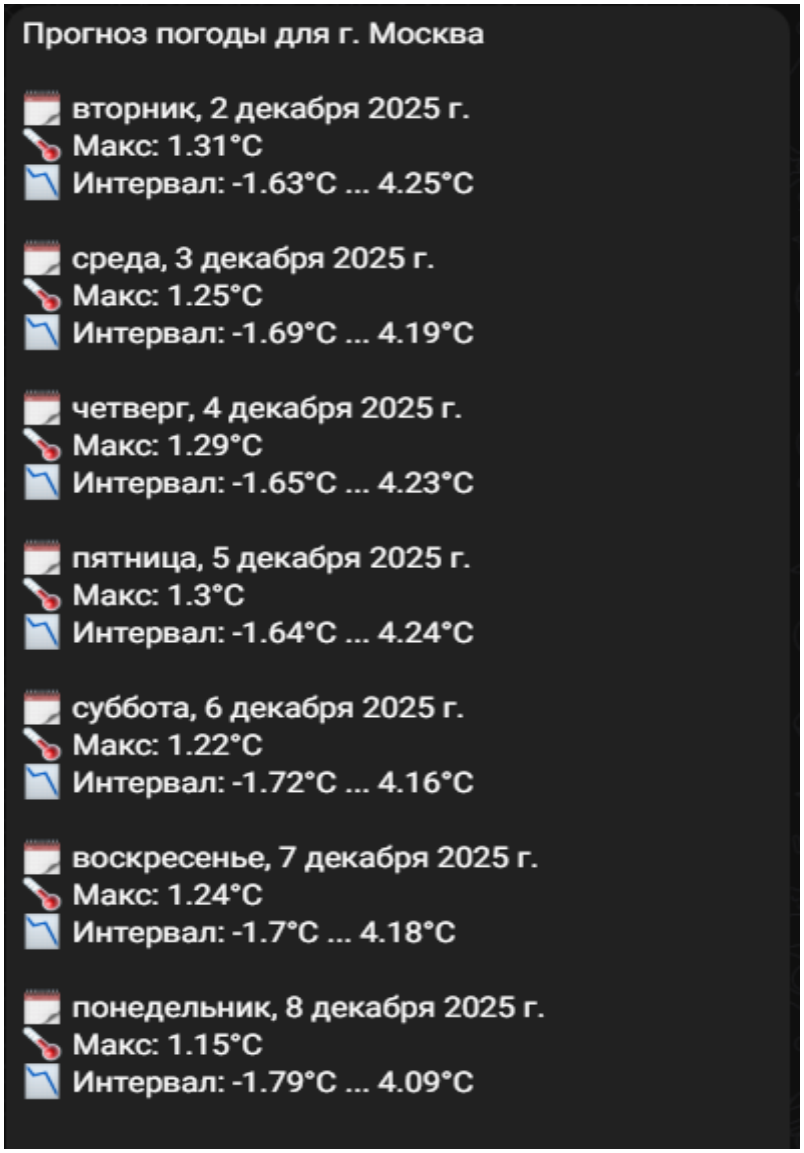
- Проверка на наличие подходящего csv, pkl для городов.
- Проверка доступ и работу сервиса.
- Ошибки в загрузки данных для прогноза города.
- Проверка на наличие достаточного числа данных.
- Проверка на наличие требуемого города в списке возможных.
- Проверка на наличие модели для города, была ли она загружена.

3. Результаты.

Представлен workflow в виде файла с названием Weather,json. Скриншоты с n8n с отображением узлов:



Пример уведомления в Telegram:



Указана на скриншоте дата прогноза, максимальная температура, интервал температур. Значки отличаются от первой лабораторной, чтобы отличать сообщения, так как бот в Telegram использовался один и тот же.

4. Выводы: что было сложным, что бы улучшили.

Сначала была попытка выполнить лабораторную без ClearML agent, но не получилось, поэтому необходимо было дополнительно использовать конфигурации и пакеты для установки.

Также возникли трудности при запуске контейнеров, так как api-server не успевал запуститься до всех остальных. Пришлось добавлять зависимости, их можно увидеть в docker-compose.yml.

Найти датасет, скачать с него данные – не составило особых трудностей, с учетом использования того же API, что и в первой лабораторной. Написать .py для ClearML также не слишком проблемным, кроме, опять же, agent, так как были попытки просто взять его через docker-compose, однако та версия была старой, поэтому было принято решение устанавливать через pip.

Построение workflow в n8n было значительно проще, чем в лабораторной два, трудности были лишь с форматами файлов, однако это все равно решилось значительно быстрее.

По улучшениям:

- Добавить дополнительные признаки.
- Добавить Pydantic моделия для внутренних структур, чтобы валидация происходила и внутри сервиса.
- Выполнить обучение для иных признаков, кроме максимальной температуры.
- Добавить поддержку большего числа моделей городов.
- Провести тестирование альтернатив LightGBM, так как возможно они будут работать лучше и стоит использовать именно их.
- Добавить больший анализ ошибок, кроме MAE и RMSE.
- Автоматизировать обновление датасета для ClearML.
- Добавить графики температур и иных признаков.
- Добавить LLM для генерации более естественного текста, объяснения прогноза пользователю.

Лабораторная работа оказалась относительно простой. Для лучшей модели удалось достичь MAE в 0.822, а RMSE – 1.06. В общем можно сказать, что был построен рабочий пайплайн ML, Fast-API сервис и реализована соответствующая интеграция.