

Проект «Определение оптимально рекомендованного бидда»

Логово программистов

Проблема

Денежный конфликт между участниками рынка такси:

Водитель думает:

Если сделаю цену низкой — заказов будет много, но заработаю мало.

Если сделаю высокую — заработаю много с одной поездки, но заказы будут редко.

Пассажир думает:

Если выберу дешёвый вариант — доеду за небольшие деньги, но машина будет простой, а ждать, возможно, дольше.

Если выберу дорогой вариант — поеду с комфортом и быстро, но заплачу намного больше.

Общая проблема:

Оба ищут свой баланс:

Водитель — между **числом заказов** и **доходом с поездки**.

Пассажир — между **ценой**, **временем ожидания** и **комфортом**.

Идеальная система должна помочь **водителю** предложить такую цену, которая устроит **пассажира**, но при этом будет выгодна и самому водителю.

Сегменты пользователей

Водители:

Это люди, исполняющие услуги транспортной перевозки

Пассажиры:

Это люди, которые используют услуги транспортной перевозки

Стек-технологий

- **Машинная модель:** Python;
- **Мобильное приложение:** Kotlin, Jetpack compose;
- **Прочее:** Git, Docker, Figma.

Решение

Проблема поиска оптимальной цены, которая балансирует между вероятностью получения заказа и доходом с поездки, эффективно решается через внедрение интеллектуальной системы динамического ценообразования

Преимущества решения:

Для водителей: увеличение общего заработка за счет интеллектуального баланса между ценой и спросом

Для пассажиров: справедливая цена, соответствующая рыночной ситуации

Для платформы: повышение эффективности matching'a водителей и пассажиров

Альтернативные решения

- **Uber** : Тестирует или внедряет опцию "**Назначь свою цену**", где водители могут повышать базовый тариф на определенный процент, чтобы компенсировать низкий спрос или неудобный маршрут.
- **Аналитические приложения (например, «Такси-Аналитика»)** : Показывают статистику по часам, районам, доходности. Они могут давать рекомендации: "В 8 утра в этом районе можно поднять цену на 10% и все равно получать заказы", или "Сейчас в вашем районе избыток машин, лучше снизить цену или переехать в соседний".

Схема работы прототипа

Сбор данных

- Система постоянно получает информацию

Анализ и расчет

- Модель машинного обучения обрабатывает данные

Генерация рекомендации

- Система предлагает водителю 1-3 варианта цен

Обучение системы

Интерфейс водителя

- Простой экран с кнопками и текущая рекомендованная цена

Демонстрация

The image shows a VS Code editor window with a Python script named `model_trainer.py` open. The script is a Jupyter Notebook-style file with code cells and output. The code is in Russian and implements a machine learning model training and evaluation process. It includes imports for `sklearn.metrics`, `precision_recall_curve`, `roc_auc_score`, `accuracy_score`, and `classification_report`. The script calculates the F1 score, ROC-AUC, and accuracy, and saves the model and meta-information using `joblib`.

Below the editor, there are two File Explorer windows. The top window shows the `data` directory, which contains three CSV files: `dataset.csv`, `sample_output.csv`, and `test_data.csv`. The bottom window shows the `models` directory, which contains a single file: `bid_model1710.joblib`.

The terminal at the bottom of the editor shows the command `(.venv) PS F:\Drivee\AI>` and the output of the script, which includes the F1 score, ROC-AUC, and accuracy.

Команда и участники

Сергей Афиногенов

Менеджер



@linveq

Душа команды

Матвей Зайцев

Дизайнер



@Ded_domsosed

Создатель прекрасного

Команда и участники

Анна Туйкова

Backend-разработчик



@k_kerra

Призер регионального конкурса
«Профессионалы 2025».

Никита Матигоров

Backend-разработчик



@LuckyMatigorov

Full-stack разработчик

Андрей Горелкин

ML-разработчик



@moiceo

Разработчик моделей нейросетей

