# Андрей Мишенков

---

+7 (926) 377-25-81 | miand_ram@mail.ru | GitHub

## О себе

Математик-прикладник (МЭИ) + MBA по логистике. 20+ лет в аналитике крупных торговых компаний и производств с оборотом от 1 млрд+ руб, из них 10+ лет – управление отделами аналитики и планирования.

## Текущий фокус

Переход в Data Science и Machine Learning. Завершаю комплексную программу Skillbox «Профессия Machine Learning Engineer» (5 курсов, сертификаты). Разработал 20+ ML-проектов, в т.ч. end-to-end, включая продакшн-упаковку моделей (REST API, Docker).

Ищу позицию, где смогу применять современные методы ML для решения бизнес-задач и развиваться в направлении MLOps и продакшн-внедрения моделей.

## Большой опыт в аналитике и планировании

Масштабная автоматизация, внедрение ERP/BI систем, работа с бизнес-заказчиком, перевод требований на язык математики и кода.

---

## Профессиональные навыки

---

### Data Science & Machine Learning

- Классические ML-алгоритмы (регрессия, классификация, кластеризация)
- Python (Jupyter, pandas, numpy, sklearn, LightGBM, XGBoost, CatBoost, Optuna) – продвинутый
- Deep Learning: PyTorch, CNN, RNN/LSTM/GRU, трансформеры (BERT, GPT-2) – базовый
- Рекомендательные системы: ALS, гибридные – базовый
- Временные ряды: Prophet, SARIMAX, TBATS, ETNA – средний
- MLOps: Git, Docker, FastAPI – средний; MLflow, Evidently, Airflow – базовый

### Data Engineering & Business Intelligence (BI)

- SQL, Базы данных: DWH, ETL, хранимые процедуры, UDF-функции, MS SQL, MySQL, ClickHouse – продвинутый
- BI: Power BI, DAX, OLAP, VBA, Power Pivot, Power Query – продвинутый
- ERP: 1С (УПП, ERP, УТ), MS Dynamics NAV – продвинутый

## Business Analysis & Supply Chain Management

- Управление цепями поставок (SCM, DRP, MRP)
- Оптимизация товарных запасов, ABC/XYZ-анализ, оборачиваемость, уровень сервиса, управление рисками
- Планирование и бюджетирование (S&OP, Cash-Flow, P&L, ROI)
- Бизнес-анализ, ценообразование и юнит-экономика
- Внедрение ERP/BI-систем (1С, Power BI, MS SQL), автоматизация бизнес-процессов

---

# ПРОЕКТЫ / ПОРТФОЛИО / СВОДНАЯ ТАБЛИЦА ПРОЕКТОВ

---

| № | Направление | Проект | Ключевая метрика | Результат |
|---|---|---|---|---|
| 1 | Классификация | СберАвтоподписка (REST API) | ROC-AUC | 0.7274 |
| 2 | Классификация | Кредитный риск | ROC-AUC | 0.7652 |
| 3 | Классификация | Отток клиентов телеком | ROC-AUC | 0.85+ |
| 4 | Регрессия | House Prices (LightAutoML) | MAE | 13 427 |
| 5 | CV | EMNIST (классические ML) | Accuracy | 84.28% |
| 6 | CV | EMNIST (CNN) | Accuracy | 90.88% |
| 7 | NLP | Фейковые новости (RNN/LSTM/GRU) | Accuracy/F1 | 87.68%/0.692 |
| 8 | NLP | Спам (BERT/GPT-2) | Accuracy/F1 | 99.30%/0.9945 |
| 9 | RecSys | Книги (ALS) | mAP@10 | 0.057371 |

| № | Направление | Проект | Ключевая метрика | Результат |
|---|---|---|---|---|
| 10 | RecSys | Книги (гибридная ALS+CatBoost) | mAP@10 | 0.050410 |
| 11 | RecSys | Рейтинг книги (CatBoost) | precision_micro | 0.4875 |
| 12 | Time Series | Прогноз спроса (линейная + сезонность) | $R^2$/MAPE | 0.649/15.19% |
| 13 | Time Series | ETNA (5 моделей) | SMAPE | 11.65% |
| 14 | Time Series | TBATS/Prophet/SARIMAX | MAPE | 19.53% |
| 15 | MLOps | MLflow трекинг (Kinopoisk) | Accuracy/F1 | 0.8239/0.7917 |
| 16 | MLOps | Airflow пайплайн (цены авто) | Accuracy | 0.7388 |
| 17 | Kaggle | Классификация (Skillbox) | Место | 1 |
| 18 | Kaggle | Регрессия (Skillbox) | Место | 2 |
| 19 | Kaggle | House Prices (Kaggle) | Топ % | 13% (~500 / 4000) |

---

# ПРОГНОЗИРОВАНИЕ И КЛАССИФИКАЦИЯ

---

## Прогнозирование целевых действий пользователя (СберАвтоподписка)

---

Разработана ML-модель для предсказания конверсии (целевых действий) пользователей на сайте «СберАвтоподписка», упакована в production-сервис REST API.

**Что сделано:**

- Проведен анализ поведения пользователей на основе данных о сессиях и событиях (~2 млн сессий, ~16 млн событий).
- Разработан пайплайн предобработки с заполнением пропусков, генерацией новых признаков (временные, географические, устройства).
- Проведена работа с дисбалансом классов.
- Сравнено 6 моделей (XGBoost, LightGBM, CatBoost, RandomForest, LogisticRegression, MLP), выбрана LightGBM.

- Модель упакована в REST API на FastAPI с временем ответа < 1 секунды.
- Созданы скрипты для периодического переобучения и тестирования.

**Результаты:**

- ROC-AUC: **0.7274** на полной выборке, на тесте – 0.7051, что на 5.5% выше целевого значения 0.65.
- Recall для класса 1: **74.16%** (модель выделяет ~74% потенциальных конверсий).
- Время ответа API: **< 1 секунды**.

**Стек:** Python, pandas, numpy, scikit-learn, LightGBM, XGBoost, CatBoost, FastAPI, Pydantic, dill, GridSearchCV, RandomizedSearchCV, TargetEncoder, OneHotEncoder, StandardScaler, JSON, REST API, uvicorn.

**GitHub:** https://github.com/Andrey-Mishenkov/SberAutopodpiska_prediction_model

---

## Оценка кредитного риска (Credit Risk Assessment)

---

Разработана ML-модель для предсказания дефолта клиента на основе кредитной истории и данных о просрочках.

**Что сделано:**

- Обработано 12 parquet-файлов с кредитными данными, 24 млн записей, чтение и обработка чанками, создана агрегированная выборка по клиентам.
- Feature engineering: количество кредитов, просрочек, финансовые коэффициенты, доли.
- Проведен отбор моделей в 2 этапа: 12 моделей → 4 лучших → 2 финальных (XGBoost, LightGBM).
- Применена кросс-валидация (5 фолдов), подбор гиперпараметров через Optuna и RandomizedSearch.
- Собран ансамбль VotingClassifier из 3 моделей (XGBoost + LightGBM + CatBoost).

**Результаты:**

- ROC-AUC на тесте: **0.7652** (ансамбль), **0.7631** (XGBoost), **0.7624** (LightGBM).
- Целевое значение: ROC-AUC 0.75, превышено на 1.5%.
- Время обучения LightGBM: **30 сек**, XGBoost: **59 сек**.

**Стек:** Python, pandas, numpy, scikit-learn, XGBoost, LightGBM, CatBoost, HistGradientBoostingClassifier, Optuna, RandomizedSearch, VotingClassifier, SelectKBest, OneHotEncoder, undersampling, pickle, parquet.

**GitHub:** https://github.com/Andrey-Mishenkov/credit_risk_management_prediction_model

---

## Прогнозирование цены дома (LightAutoML)

Разработана регрессионная модель для предсказания стоимости жилых домов на основе датасета Kaggle House Prices.

**Что сделано:**

- Проведен полный цикл EDA: анализ пропусков, выбросов, корреляций.
- Создано ~30 новых признаков (комбинации площадей помещений, возрастные, бинарные, временные, взаимодействия).
- Обучено 2 конфигурации LightAutoML (базовая и улучшенная) с разными таймаутами и параметрами.
- Выполнен анализ важности признаков в режимах fast и accurate.
- Применена стратификация по целевой переменной (8 интервалов).

**Результаты:**

- MAE тест: **13 427** (улучшенная модель), **13 621** (базовая).
- Целевое значение: MAE < 15 324, превышение на **~12%**.
- Время обучения: 125 сек (базовая), 1 312 сек (улучшенная).

**Стек:** Python, pandas, numpy, matplotlib, seaborn, scipy, LightAutoML, TabularAutoML, LightGBM, CatBoost, XGBoost, linear_l2.

**GitHub:** https://github.com/Andrey-Mishenkov/house_prices_prediction_kaggle/tree/main

---

# КОМПЬЮТЕРНОЕ ЗРЕНИЕ

---

## Распознавание рукописных символов EMNIST (классические ML модели)

---

Разработана ML-модель для классификации рукописных букв и цифр на основе датасета EMNIST. Готовая модель упакована в FastAPI-сервис с веб-интерфейсом и запуском в Docker-контейнере.

**Что сделано:**

- Загружен датасет EMNIST (~113 тыс обучающих, ~19 тыс тестовых изображений, 47 классов).
- Реализована предобработка: нормализация пикселей, HOG-признаки, статистические и морфологические признаки.
- Создан пайплайн: MinMaxScaler → HOG → VarianceThreshold → SelectKBest → классификатор.
- Проведено сравнение и отбор из ~10 моделей (линейные, ансамбли), подобраны гиперпараметры через Optuna.
- Лучшая модель: ExtraTreesClassifier, сохранена в сжатом формате .pkl.gz с метаданными.

**Результаты:**

- Accuracy на тесте: **84.28%**, целевой порог 0.68 превышен на **~16%** (п.п.).
- Accuracy кросс-валидация (3-fold): **83.75%**.

**Стек:** Python, scikit-learn, XGBoost, LightGBM, CatBoost, emnist, skimage, HOG, MinMaxScaler, SelectKBest, VarianceThreshold, Optuna, cloudpickle, gzip, sklearnex, FastAPI, uvicorn, Docker, HTML, JavaScript.

**GitHub:** https://github.com/Andrey-Mishenkov/cv_handwritten_character_recognition_classical_ml

---

### Распознавание рукописных символов EMNIST (CNN)

---

Разработана сверточная нейронная сеть для классификации рукописных символов на основе датасета EMNIST. Готовая модель упакована в FastAPI-сервис с веб-интерфейсом и запуском в Docker-контейнере.

**Что сделано:**

- Загружен датасет EMNIST (~113 тыс обучающих, ~19 тыс тестовых изображений, 47 классов).
- Применена аугментация данных: padding, RandomCrop, RandomRotation (±8°).
- Сравнено 11 архитектур CNN (LogReg, LeNet-5, VGG-like, ResNet и др.).
- Использован AdamW оптимизатор, CrossEntropyLoss с label_smoothing, Early Stopping, ReduceLROnPlateau.
- Лучшая модель: ImprovedCNN (VGG-подобная архитектура). Дообучена на полном датасете, сохранена в формате .ckpt.

**Результаты:**

- Accuracy на валидации: **90.88%**, превышение целевого значения 0.87 на **~4%** (п.п.).
- Loss валидация: **0.9252**.

**Стек:** Python, PyTorch, torchvision, numpy, matplotlib, torchinfo, AdamW, CrossEntropyLoss, label_smoothing, Early Stopping, ReduceLROnPlateau, BatchNorm2d, Dropout, FastAPI, uvicorn, Docker, HTML, JavaScript.

**GitHub:** https://github.com/Andrey-Mishenkov/cv_handwritten_character_recognition_neural_networks

---

## NLP И АНАЛИЗ ТЕКСТА

---

### Классификация фейковых новостей (RNN / LSTM / GRU)

Разработана модель бинарной классификации новостных статей на реальные и фейковые с использованием рекуррентных нейронных сетей.

**Что сделано:**

- Использован датасет FakeNewsNet (~23 тыс записей, после очистки ~22 тыс).
- Создано текстовое поле из конкатенации домена, заголовка и индикатора URL.
- Обучены Word2Vec эмбеддинги (размерность 100) на тренировочных данных.
- Реализовано 3 архитектуры: RNN, LSTM, GRU (однослойные, hidden_size=64, dropout=0.6).
- Применено Early Stopping, ручное уменьшение learning rate, сохранение лучших весов.
- Лучшая модель: LSTM.

**Результаты:**

- LSTM: Accuracy **87.68%**, F1-Score (fake) **0.6920**, ROC-AUC **0.8985**.
- GRU: Accuracy **86.89%**, F1-Score (fake) **0.6667**, ROC-AUC **0.8908**.
- RNN: Accuracy **86.25%**, F1-Score (fake) **0.6303**, ROC-AUC **0.8794**.
- Целевые значения по всем метрикам превышены на **1–5%** (п.п.).

**Стек:** Python, pandas, numpy, matplotlib, PyTorch, torchtext, gensim, scikit-learn, spaCy, Word2Vec, RNN, LSTM, GRU, nn.Embedding, BCEWithLogitsLoss, AdamW, BucketIterator.

**GitHub:** https://github.com/Andrey-Mishenkov/fake_news_classification_using_rnn

---

## Классификация спам-сообщений (BERT/GPT-2)

---

Разработана модель бинарной классификации электронных писем на спам и не спам с использованием предобученных трансформеров BERT и GPT-2.

**Что сделано:**

- Использован датасет Trec 2006 (10 000 писем для экспериментов).
- Разработана функция text_cleaner: лемматизация, удаление стоп-слов, расширение сокращений.
- Получены эмбеддинги через BERT (sentence-transformers) и GPT-2 (transformers), размерность 768.
- Обучен простой классификатор (2 полносвязных слоя + Dropout) поверх эмбеддингов.
- Сравнены метрики и время генерации эмбеддингов для двух подходов.

**Результаты:**

- BERT: Accuracy **99.30%**, F1-Score (spam) **0.9945**, ROC-AUC **0.9998**, время генерации **2.5 мин**.
- GPT-2: Accuracy **99.20%**, F1-Score (spam) **0.9937**, ROC-AUC **0.9998**, время генерации **6 мин**.

- BERT быстрее в 2.6 раза при сопоставимом качестве.
- Целевые метрики Accuracy 73–75% превышены на **24–26%** (п.п.), F1-Score 0.84–0.85 на **14–15%** (п.п.).

**Стек:** Python, PyTorch, transformers, sentence-transformers, nltk, pandas, numpy, matplotlib, seaborn, scikit-learn, BERT (bert-base-uncased), GPT-2, BCELoss, AdamW, WordNetLemmatizer, DataLoader, TensorDataset.

**GitHub:** https://github.com/Andrey-Mishenkov/spam_classification_models_bert_and_gpt-2

---

# РЕКОМЕНДАТЕЛЬНЫЕ СИСТЕМЫ

---

## Рекомендательная система книг (ALS – матричная факторизация)

---

Разработана рекомендательная система книг с использованием метода матричной факторизации (ALS) на основе implicit-оценок.

**Что сделано:**

- Загружены данные: книги (10 000), рейтинги (~6 млн оценок), жанры/теги.
- Бинаризация рейтингов (≥4 → 1, иначе → 0), разбиение train/test 70/30.
- Построена user-item матрица (53 тыс × 10 тыс).
- Проведен полный перебор гиперпараметров (108 комбинаций).
- Рассчитаны бейзлайны: случайный и популярные книги.

**Результаты:**

- ALS (лучший): mAP@10 **0.057371** (+105% к популярным книгам).
- ALS (базовый): mAP@10 **0.047436** (+70% к популярным книгам).
- Популярные книги: mAP@10 **0.027912**.
- Случайный бейзлайн: mAP@10 **0.000535**.
- Целевой порог (популярные книги) mAP@10 = 0.0279 превышен в **~2 раза**, **+0.03** (п.п.).

**Стек:** Python, pandas, numpy, implicit, ALS, matrix factorization, ap_at_k.

**GitHub:** https://github.com/Andrey-Mishenkov/recommender_system_using_matrix_factorization_als

---

## Рекомендательная система книг (гибридная ALS + CatBoost)

---

Разработана гибридная рекомендательная система, сочетающая коллаборативную фильтрацию (ALS) и контентные признаки (CatBoost).

**Что сделано:**

- Загружены данные: книги, рейтинги, жанры (goodreads_book_genres_initial.json).
- Бинаризация рейтингов, разбиение train/test 70/30 на основе порядка прочтения.
- Обучена базовая модель ALS (factors=50, iterations=15).
- Собраны контентные признаки пользователей и книг (статистика оценок, жанровые предпочтения, средний рейтинг).
- Обучен CatBoostClassifier, объединены рекомендации ALS и CatBoost с оптимальным весом.

**Результаты:**

- Гибридная модель: mAP@10 **0.050410**.
- ALS (базовый): mAP@10 **0.047412**.
- Улучшение: **+6.3%** (абсолютное +0.003 п.п.).
- Оптимальный вес: ALS = 0.5, CatBoost = 0.5.
- Время работы: ALS **0.5 сек**, гибрид **8.7 сек**.

**Стек:** Python, pandas, numpy, implicit, ALS, CatBoost, CatBoostClassifier, OneHotEncoder, ap_at_k, Jupyter Notebook.

**GitHub:** https://github.com/Andrey-Mishenkov/recommender_system_hybrid_using_matrix_factorization_als_and_gradient_boosting_catboost

---

## Предсказание рейтинга книги (мультиклассовая классификация)

---

Создан прототип рекомендательной системы для предсказания рейтинга книги (целые значения 1–5) на основе истории оценок, жанров, эмбеддингов заголовков и описаний.

**Что сделано:**

- Использован датасет goodbooks-10k (~6 млн оценок, 10 000 книг).
- Обработаны жанры: фильтрация ~34 тыс тегов до 16 основных категорий.
- Сгенерированы эмбеддинги заголовков и описаний через Sentence-BERT (all-mpnet-base-v2), уменьшена размерность через PCA (768 → 100).
- Протестировано 5 конфигураций признаков (жанры+статистика, только эмбеддинги, все вместе).
- Применена работа с дисбалансом классов (сбалансированные подвыборки, усиление миноритарных классов).

**Результаты:**

- Лучшая модель: precision_micro **0.4875**.
- Превышение целевого порога precision_micro 0.3 в **~1.6 раза, +0.18** (п.п.).
- Рейтинг 5: precision **0.6065**, recall **0.6309**.
- Рейтинг 1: precision **0.1811**, recall **0.4034**.

**Стек:** Python, pandas, numpy, CatBoost, CatBoostClassifier, Sentence-BERT, all-mpnet-base-v2, PCA, scikit-learn, OneHotEncoder, Jupyter Notebook.

**GitHub:** https://github.com/Andrey-Mishenkov/recommender_system_prototype_using_content-based_filtering

---

# ПРОГНОЗИРОВАНИЕ ВРЕМЕННЫХ РЯДОВ

---

## Прогнозирование спроса (линейная регрессия + сезонность)

---

Проведен анализ продаж компании и сформирован прогноз на следующий год с использованием линейной регрессии и коэффициентов сезонности.

**Что сделано:**

- Объединены данные о продажах, остатках, сезонных индексах и календаре периодов (2 года, ~6 тыс записей, 270+ товаров).
- Проведена очистка: корректировка продаж при отсутствии товара, замена выбросов на медиану, очистка от сезонности (Яндекс Wordstat).
- Выполнен ABC-анализ: группа A (5 групп) – 49% оборота, B (17 групп) – 40%, C (34 группы) – 11%.
- Построена линейная регрессия для каждого товара на очищенных данных с применением сезонных индексов.
- Проведен статистический анализ: тесты Шапиро-Уилка и Пирсона, F-критерий, t-статистики, доверительные интервалы.

**Результаты:**

- $R^2$ = **0.649** (модель объясняет 65% дисперсии).
- Средняя ошибка аппроксимации: **15.19%**.
- F-критерий: **40.63** > F-теор **4.30** (модель статистически значима).
- Прогноз оборота: **1 197.3 млн руб** (+60% к прошлому году).

**Стек:** Python, pandas, numpy, matplotlib, seaborn, scipy, scikit-learn, LinearRegression, statsmodels.

**GitHub:** https://github.com/Andrey-Mishenkov/demand_forecasting_using_linear_regression_and_seasonality

---

## Прогнозирование временных рядов (TBATS, Prophet, SARIMAX, AutoARIMA)

---

Оптимизация и сравнение 4-х моделей временных рядов для прогнозирования спроса на товар, выход на целевую метрику.

**Что сделано:**

- Использован датасет Kaggle Demand Forecasting (один товар, один магазин, данные 5 лет до дня).
- Разделение: train 1461 день (2013–2016), test 365 дней (2017).
- Созданы экзогенные переменные: тригонометрические, one-hot (день недели, месяц, квартал, выходной).
- Проведен многоэтапный подбор гиперпараметров для каждой модели (GridSearch, stepwise, полный перебор).
- Рассчитаны метрики MSE, MAE, MAPE, зафиксировано время обучения.

**Результаты:**

- TBATS Tuned: MAPE **19.53%**, время обучения **155 сек**.
- SARIMAX Tuned: MAPE **19.88%**, время обучения **39.7 сек**.
- Prophet Tuned: MAPE **19.94%**, время обучения **0.12 сек**.
- AutoARIMA: MAPE **28–29%** (не достиг целевого значения).
- Целевого значения MAPE < 20% достигли 3 модели.

**Стек:** Python, pandas, numpy, matplotlib, scipy, tbats, Prophet, pmdarima, auto_arima, statsmodels, SARIMAX, GridSearch, Box-Cox, one-hot encoding, trigonometric features.

**GitHub:** https://github.com/Andrey-Mishenkov/time_series_models_TBATS_PROPHET_ARIMA/tree/main

---

## Прогнозирование временных рядов (ETNA: Naive, Elastic, CatBoost, Prophet, SARIMAX)

---

Настройка и оптимизация 5-ти моделей временных рядов из библиотеки ETNA, выход на целевую метрику. Сформирован прогноз продаж для 150 сегментов (50 товаров × 3 магазина) на горизонт 90 дней.

**Что сделано:**

- Использован датасет Kaggle Demand Forecasting (2013–2017), отобраны 3 магазина (150 сегментов).
- Созданы экзогенные признаки: календарные, бинарные флаги, тригонометрические (годовая и недельная сезонность).
- Проведен EDA: визуализация, ACF-анализ, STL-декомпозиция, матрица корреляций.
- Настроены и обучены 5 моделей: Naive, ElasticNet, CatBoost, Prophet, SARIMAX с подбором гиперпараметров через Optuna.
- Выполнен бэктестинг с расширяющимся окном (3–5 фолдов) на всех сегментах.

**Результаты:**

- SARIMAX: SMAPE **11.65%** (бэктест 11.49%), время обучения **1.7 мин**.
- Prophet: SMAPE **11.65%** (бэктест 11.60%), время обучения **0.9 мин**.
- CatBoost: SMAPE **11.66%** (бэктест 11.37%), время обучения **15.5 мин**.

- ElasticNet: SMAPE **11.87%** (бэктест 10.94%), время обучения **3.3 мин**.
- Naive: SMAPE **16.88%**.
- Целевого значения SMAPE < 12% достигли 4 модели.

**Стек:** Python, pandas, numpy, matplotlib, seaborn, etna, TSDataset, NaiveModel, ElasticPerSegmentModel, CatBoostPerSegmentModel, ProphetModel, SARIMAXModel, Optuna, AutoRegressivePipeline, LogTransform, DeseasonalityTransform, LinearTrendTransform, DateFlagsTransform, FilterFeaturesTransform, STL, ACF.

**GitHub:** https://github.com/Andrey-Mishenkov/time_series_library_ETNA_models_ELASTIC_CATBOOST_PROPHET_SARIMAX/tree/main

---

# MLOps, ИНФРАСТРУКТУРА, ДЕПЛОЙ

---

## Трекинг экспериментов (MLflow) – классификация тональности отзывов Kinopoisk

---

Проведена реорганизация скрипта классификации отзывов с внедрением системы трекинга экспериментов MLflow.

**Что сделано:**

- Интегрирован MLflow Tracking в существующий скрипт классификации отзывов.
- Настроено логирование параметров, метрик и артефактов моделей.
- Сохранена baseline-модель (MultinomialNB + CountVectorizer).
- Разработан альтернативный подход: TfidfVectorizer + LogisticRegression (10 000 признаков).
- Проведено сравнительное тестирование обеих моделей на идентичных данных.

**Результаты:**

- Baseline (Naive Bayes + BOW): accuracy **0.8076**, F1-score **0.7553**.
- Alternative (LogReg + TF-IDF): accuracy **0.8239**, F1-score **0.7917**.
- Улучшение accuracy: **+1.6%**, F1-score: **+3.6%**.
- Все эксперименты зафиксированы в MLflow UI для воспроизводимости.

**Стек:** Python, scikit-learn, MLflow, MultinomialNB, LogisticRegression, CountVectorizer, TfidfVectorizer, pickle, Git.

**GitHub:** https://github.com/Andrey-Mishenkov/mlflow_experiment_tracking_review_sentiment_classification

---

## Автоматизация ML-пайплайна (Apache Airflow) – предсказание цены автомобиля

Разработан автоматизированный пайплайн машинного обучения для классификации автомобилей по ценовой категории с использованием Apache Airflow.

**Что сделано:**

- Разработан DAG с двумя PythonOperator: обучение модели и пакетное предсказание.
- Создан пайплайн предобработки: удаление неинформативных колонок, обработка выбросов (IQR), генерация признаков.
- Сравнено 3 модели классификации (LogisticRegression, RandomForest, SVC) с кросс-валидацией (4 фолда).
- Настроено расписание: ежедневно в 15:00, логирование через модуль logging.
- Реализовано сохранение модели с временной меткой, автоматический выбор последнего файла для предсказания.

**Результаты:**

- Лучшая модель: RandomForestClassifier, accuracy **0.7388** (кросс-валидация).
- LogisticRegression: accuracy **0.7290**, SVC: accuracy **0.7375**.
- DAG выполняется ежедневно, результаты сохраняются в CSV с временной меткой.

**Стек:** Python, Apache Airflow, DAG, PythonOperator, BashOperator, scikit-learn, LogisticRegression, RandomForestClassifier, SVC, ColumnTransformer, OneHotEncoder, StandardScaler, IQR, dill, logging, JSON, CSV, Git, Docker.

**GitHub:** https://github.com/Andrey-Mishenkov/airflow_car_price_prediction_pipeline_training_model

---

## KAGGLE-СОРЕВНОВАНИЯ / andreimishenkov

---

### Kaggle: Классификация (внутреннее соревнование Skillbox)

---

Предсказание экономических параметров группы людей по их социально-демографическим характеристикам.

**Результат: 1 место**

**Стек:** Python, pandas, scikit-learn, классификация, feature engineering, XGBoost, LightGBM, CatBoost.

**Kaggle:**

- Описание: https://www.kaggle.com/competitions/skillbox-ml-junior-classification-10/leaderboard
- Лидерборд: https://www.kaggle.com/competitions/skillbox-ml-junior-classification-10/leaderboard

## Kaggle: Регрессия (внутреннее соревнование Skillbox)

Задача регрессии. Определение прочности бетона.

**Результат: 2 место**

**Стек:** Python, pandas, scikit-learn, регрессия, feature engineering, XGBoost, LightGBM, CatBoost.

**Kaggle:**

- Описание: https://www.kaggle.com/competitions/skillbox-ml-junior-regression-10/overview
- Лидерборд: https://www.kaggle.com/competitions/skillbox-ml-junior-regression-10/leaderboard

## Kaggle: House Prices (публичное соревнование) Advanced Regression Techniques

Задача регрессии. Прогнозирование цены дома на основе характеристик недвижимости.

**Результаты:**

- **Топ 13%** (~500 место из 4000 участников).
- RSME: 0.12319.

**Стек:** Python, pandas, scikit-learn, LightAutoML, регрессия, feature engineering.

**Kaggle:**

- Ссылка: https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques/overview
- Лидерборд скриншот: https://github.com/Andrey-Mishenkov/house_prices_prediction_kaggle/blob/main/leaderboard.png