## 📂 Структура проекта

final\_project/

│

├─ dags/

│ └─ blendcal\_inference.py ← DAG для Airflow (end-to-end pipeline)

│

├─ project/

│ ├─ modules/

│ │ ├─ extract\_csv.py ← загрузка CSV → parquet

│ │ ├─ prepare.py ← фичепайплайн (EDA → признаки)

│ │ └─ ensemble.py ← загрузка артефактов, инференс BlendCAL

│ │

│ ├─ data/

│ │ ├─ raw/ ← входные CSV (ga\_sessions.csv, ga\_hits.csv)

│ │ ├─ landing/ ← parquet после ingest

│ │ ├─ staging/ ← features.parquet

│ │ └─ predictions/ ← preds.csv + run\_id=...

│ │

│ └─ artifacts/

│ ├─ prep/prep\_params.json, freq\_maps.json

│ ├─ CatBoost\_calibrated\_isotonic.pkl, LGB.booster.txt, XGB.booster.json, ...

│ ├─ BlendCAL\_weights.json, BlendCAL\_thresholds.json

│ └─ (доп. подкаталоги: eval/, split/, tabnet/ и т.д.)

│

├─ api/app/ ← код API (мы брали preprocessor, artifacts\_loader и config оттуда)

│

└─ docker\_airflow/

├─ docker-compose.yml ← Airflow стек (postgres + init + webserver + scheduler)

└─ Dockerfile ← кастомный образ airflow-blendcal:py39

## ⚙️ Docker/Airflow окружение

* Образ: airflow-blendcal:py39 (на базе apache/airflow:2.9.2-python3.9)
* Установленные зависимости:
* numpy==1.21.6
* pandas==1.3.5
* scipy==1.7.3
* joblib==1.4.2
* scikit-learn==0.24.2
* xgboost==2.1.4
* lightgbm==4.6.0
* catboost==1.2.7
* docker-compose.yml монтирует:
  + dags/, project/, api/, project/data/, project/artifacts/
* SECRET\_KEY настроен, логи Airflow читаются в UI.
* PYTHONPATH настроен на project/modules и api.

## ✅ Что работает

* Контейнеры поднимаются (webserver, scheduler, postgres).
* Airflow UI доступен: <http://localhost:8080> (admin/admin).
* DAG blendcal\_inference\_csv\_once виден и запускается.
* Весь пайплайн проходит до конца:
  + ingest\_sessions/heats → parquet
  + prepare\_features → features.parquet (~1.86 млн строк, 41 фича)
  + predict\_ensemble → preds.csv (~107 MB, 1.86 млн строк, доля y\_hat=1 ≈ 5.3%)
  + archive\_outputs → копия preds.csv в подпапку run\_id=...

## 📊 Проверки

* Дубликатов session\_id нет.
* proba адекватно распределён (min=0.0, max≈0.59, mean≈0.023).
* y\_hat сбалансирован: ≈ 98.5k положительных из 1.86M.
* В features.parquet NaN < 7% только по фичам any\_car\_detail/n\_models/n\_brands.

## 🔧 Улучшения, которые внесли

* Логирование прогресса в prepare.py, extract\_csv.py, ensemble.py.
* Timeout’ы в DAG, чтобы шаги не зависали бесконечно.
* Автосоздание папок (landing/staging/predictions).
* Архивирование предсказаний по run\_id, чтобы не затирались результаты.
* Всё это сделано **без новых зависимостей** (только стандартная библиотека).

## 📄 Git & README

* .gitignore (в корне репо) исключает тяжёлые данные и модели, но оставляет prep/prep\_params.json и prep/freq\_maps.json.
* README для Airflow включает:
  + требования
  + структуру проекта
  + команды запуска
  + описание шагов DAG
  + sanity-чек команду для preds.csv
  + типичные ошибки и их фиксы

## 🎯 Итог

* Мы воспроизвели логику ноутбука в виде полного Airflow-пайплайна.
* DAG end-to-end: ingestion → feature engineering → inference ансамбля.
* Архитектура законченная и воспроизводимая.
* Проект готов к публикации на GitHub: достаточно выложить код, DAG, docker-compose, .gitignore и README (данные и модели — отдельно или в LFS/релизах).