Assignment MLOPS

Deskripsi Assignment

Assignment ini berfokus pada implementasi praktik MLOps end-to-end menggunakan dataset House Prices Advanced Regression Techniques dari Kaggle. Student akan mengembangkan pipeline machine learning yang siap produksi dengan mengikuti best practice industri.
Link : dataset

Soal 1: Produksi Modular Code

```
assigment_deployment/
|── artifacts/
|   |── train.csv
├── src/
|   |──run_pipeline.py
|   ├── data/
|   |   ├── __init__.py
|   |   ├── data_loader.py      # Data loading utilities
|   |   └── data_processor.py   # Data preprocessing pipeline
|   ├── models/
|   |   ├── __init__.py
|   |   ├── model.py           # Model architecture definitions
|   |   └── trainer.py         # Training logic
|   ├── utils/
|   |   ├── __init__.py
|   |   ├── logger.py          # Logging configuration
|   |   └── config.py          # Configuration management
|   └── api/
|       ├── __init__.py
|       ├── main.py            # FastAPI application
|       └── schemas.py         # Pydantic models
├── notebooks/
|   └── research.ipynb
├── config/
|   └── config.yaml
├── requirements.txt
└── README.md
|__
```

- older **notebooks** digunakan untuk menyimpan file `.ipynb` sebagai tempat riset awal sebelum proses modularisasi kode.
- Folder **artifacts** digunakan untuk menyimpan data seperti file `.csv` yang akan digunakan sebagai data latih.
- Folder **src** adalah folder inti kode, dengan sub-folder sebagai berikut:

    - **data**: Berisi script untuk load data dan preprocessing data.
    - **model**: Berisi script untuk training model dan prediksi model.
    - **utils**: Berisi script untuk logger dan konfigurasi.
    - **api**: Berisi script untuk kebutuhan API.
- Folder **config** digunakan untuk menyimpan file konfigurasi seperti `config.yaml`.
- File **requirements.txt** berisi daftar library yang digunakan dalam proyek.
- File **README.md** berisi penjelasan detail mengenai proyek.

Berikut langkah serta penjelasan dari setiap proses

1. Riset Awal dan Pembuatan Notebook (`research.ipynb`)

    Langkah pertama adalah melakukan riset data yang mencakup:

    - Pengumpulan Data: Mengambil data dari berbagai sumber yang relevan, seperti file CSV, database, atau API.
    - Pra-pemrosesan Data: Menangani missing values, encoding untuk variabel kategorikal, normalisasi fitur, dan lain-lain.
    - Modeling: Menggunakan berbagai algoritma machine learning untuk melatih model dan melakukan evaluasi. Metrik yang umum digunakan antara lain RMSE, MAE, dan R^2 untuk regresi..

2. Membuat Folder notebooks dan Menyimpan File `research.ipynb`

    Setelah eksperimen selesai di notebook, file research.ipynb disimpan di folder notebooks. Folder ini bertujuan untuk menyimpan file notebook yang berhubungan dengan riset dan eksperimen, sehingga proyek dapat terorganisir dengan baik.

3. Membuat Folder artifacts dan Menyimpan Data

    Folder artifacts dibuat untuk menyimpan dataset yang akan digunakan dalam pipeline machine learning.

4. Membuat File config.yaml

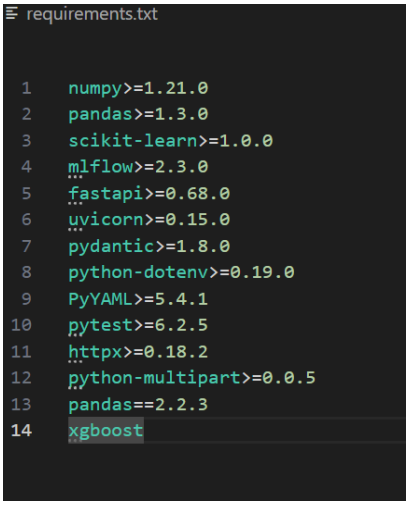
    File config.yaml digunakan untuk menyimpan berbagai konfigurasi yang diperlukan dalam proyek, seperti:

    - Konfigurasi Umum: Lokasi penyimpanan file dan pengaturan lain terkait.
    - Konfigurasi MLflow: Pengaturan untuk logging eksperimen dan model menggunakan MLflow.
    - Informasi Model dan Parameter: Detail tentang model yang digunakan dan parameter terkait yang digunakan dalam eksperimen.

    Dengan config.yaml, parameter dan pengaturan proyek menjadi lebih terstruktur dan mudah diubah.

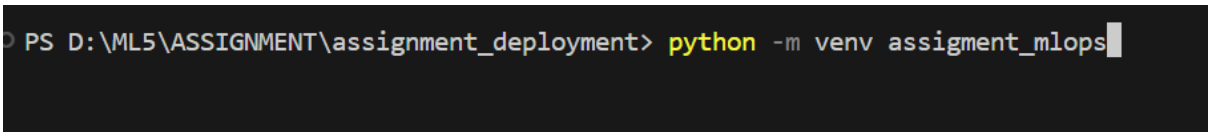
5. Membuat File requirements.txt

File requirements.txt berisi daftar pustaka (libraries) yang diperlukan oleh proyek. File ini dihasilkan berdasarkan dependensi yang digunakan dalam notebook research.ipynb. Tujuannya adalah untuk memudahkan penginstalan pustaka yang dibutuhkan di lingkungan lain atau bagi kolaborator lain yang bekerja pada proyek ini.

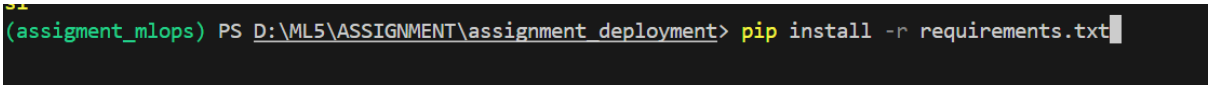


6. Membuat Virtual Environment untuk Pipeline

Virtual environment ini hanya akan memiliki pustaka yang dibutuhkan untuk proyek ini.



activated env yang sudah dibuat kemudian install library yang dibutuhkan.



7. Modularisasi Kode Berdasarkan Notebook `research.ipynb`

Setelah eksperimen selesai dan model terbaik ditemukan, kode dari notebook dipindahkan ke dalam file Python terpisah agar lebih terstruktur. Ini bertujuan untuk menjadikan kode lebih reusable dan memudahkan pemeliharaan di masa depan.

- File `logger.py`: File ini bertanggung jawab untuk mencatat informasi penting dan menangani error dalam pipeline. Dengan menggunakan library seperti `logging`, file ini dapat mencatat error, peringatan, dan informasi penting lainnya yang terjadi selama eksekusi pipeline.
- File `data_loader.py`: File ini bertujuan untuk memuat data, baik dari file CSV, database, atau sumber lain. Data yang dimuat akan diproses lebih lanjut di langkah berikutnya.
- File `data_processor.py`: Di sini dilakukan pra-pemrosesan data. Proses ini meliputi pembersihan data, menangani missing values, encoding variabel

kategorikal, normalisasi, dan pembagian data menjadi set pelatihan dan pengujian. File ini memastikan bahwa data siap untuk dimasukkan ke dalam model.

- File `model.py` : Di file ini, kita mendefinisikan model machine learning yang akan digunakan dalam pipeline. Misalnya, model regresi linier, regresi pohon keputusan, atau model lain sesuai dengan kebutuhan proyek.
- File `trainer.py` : File `trainer.py` bertugas untuk melatih model yang telah dibuat dengan menggunakan data pelatihan yang telah diproses. Di sini, kita juga bisa melacak metrik seperti RMSE, MAE, dan R^2 untuk mengevaluasi kinerja model. Konfigurasi MLflow experiment tracking juga dibuat di sini.

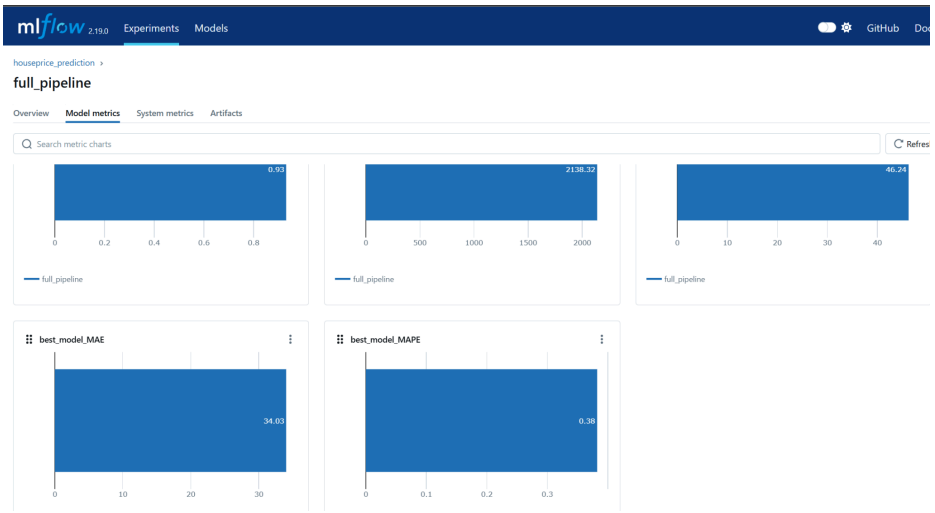
8. Membuat file `run_pipeline.py`, berfungsi untuk menjalankan seluruh urutan pipeline secara otomatis. File ini akan mengatur alur eksekusi, mulai dari memuat data, pra-pemrosesan, pembuatan model, pelatihan model, hingga evaluasi model.

9. Setelah mengonfigurasi MLflow, langkah berikutnya adalah untuk memonitor eksperimen yang dijalankan, yaitu dengan melacak parameter, metrik, dan model yang dihasilkan.

untuk mengakses hasil eksperimen di mlflow

```
mlflow ui --backend-store-uri sqlite:///mlflow.db
```

Akses UI pada http://localhost:5000 untuk melihat parameter, metrik, dan artifact yang telah dilacak.



10. Membuat API

- Membuat file schema.py

  schema.py digunakan untuk mendefinisikan struktur data yang digunakan oleh API, seperti request body dan response body.

- Membuat file main.py

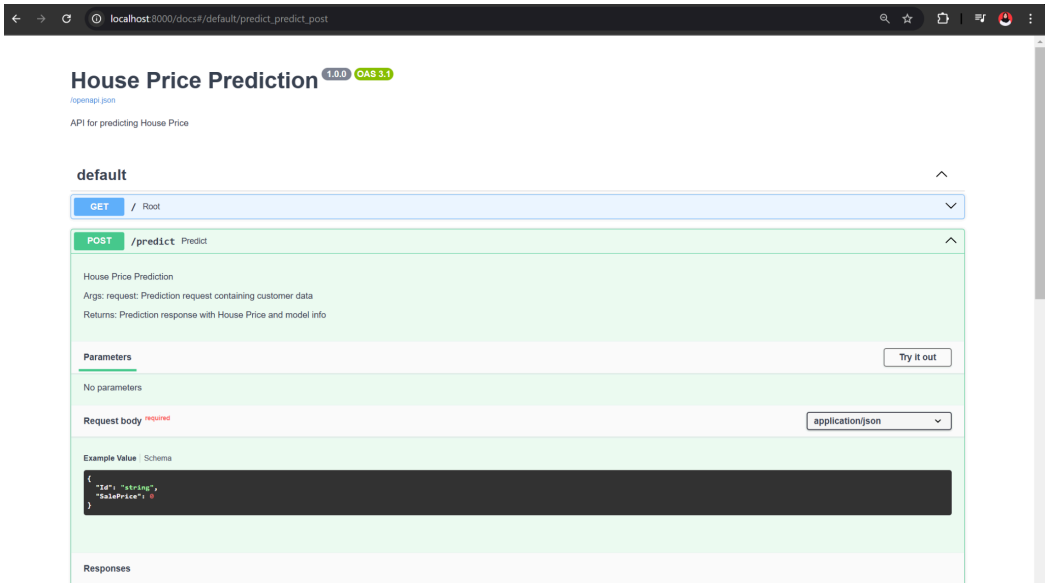
main.py adalah file yang berfungsi untuk mengonfigurasi FastAPI dan menentukan routing untuk API. Ini akan mencakup endpoints untuk prediksi rumah berdasarkan input yang diterima.

- Start API Server

  Untuk menjalankan server FastAPI menggunakan uvicorn, yang merupakan ASGI server untuk menjalankan aplikasi FastAPI. Jalankan perintah berikut di terminal untuk memulai server:

  - uvicorn src.api.main:app --reload
  - src.api.main:app adalah lokasi file main.py dan objek FastAPI yang ada di dalamnya.
  - --reload digunakan untuk me-reload aplikasi saat ada perubahan di kode
  - Setelah server berjalan, dapat mengakses dokumentasi API secara otomatis di :
  - http://localhost:8000/docs

  Tampilan doc API



- Testing API
  input yang digunakan:
  {
  "Id": 0,
  "MSSubClass": 0,
  "MSZoning": "string",
  "LotFrontage": 0,
  "LotArea": 0,
  "Street": "string",
  "LotShape": "string",
  "LandContour": "string",
  "Utilities": "string",

```
"LotConfig": "string",
"LandSlope": "string",
"Neighborhood": "string",
"Condition1": "string",
"Condition2": "string",
"BldgType": "string",
"HouseStyle": "string",
"OverallQual": 0,
"OverallCond": 0,
"YearBuilt": 0,
"YearRemodAdd": 0,
"RoofStyle": "string",
"RoofMatl": "string",
"Exterior1st": "string",
"Exterior2nd": "string",
"MasVnrArea": 0,
"ExterQual": "string",
"ExterCond": "string",
"Foundation": "string",
"BsmtQual": "string",
"BsmtCond": "string",
"BsmtExposure": "string",
"BsmtFinType1": "string",
"BsmtFinSF1": 0,
"BsmtFinType2": "string",
"BsmtFinSF2": 0,
"BsmtUnfSF": 0,
"TotalBsmtSF": 0,
"Heating": "string",
"HeatingQC": "string",
"CentralAir": "string",
"Electrical": "string",
"onestFlrSF": 0,
"twondFlrSF": 0,
"LowQualFinSF": 0,
"GrLivArea": 0,
"BsmtFullBath": 0,
"BsmtHalfBath": 0,
"FullBath": 0,
"HalfBath": 0,
"BedroomAbvGr": 0,
"KitchenAbvGr": 0,
"KitchenQual": "string",
"TotRmsAbvGrd": 0,
"Functional": "string",
"Fireplaces": 0,
"GarageType": "string",
"GarageYrBlt": 0,
```

```
  "GarageFinish": "string",
  "GarageCars": 0,
  "GarageArea": 0,
  "GarageQual": "string",
  "GarageCond": "string",
  "PavedDrive": "string",
  "WoodDeckSF": 0,
  "OpenPorchSF": 0,
  "EnclosedPorch": 0,
  "threeSsnPorch": 0,
  "ScreenPorch": 0,
  "PoolArea": 0,
  "MiscVal": 0,
  "MoSold": 0,
  "YrSold": 0,
  "SaleType": "string",
  "SaleCondition": "string"
}
```

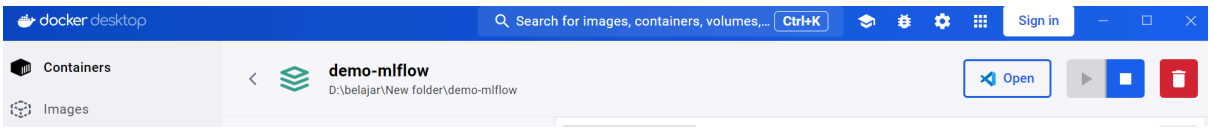

12. Deployment Docker

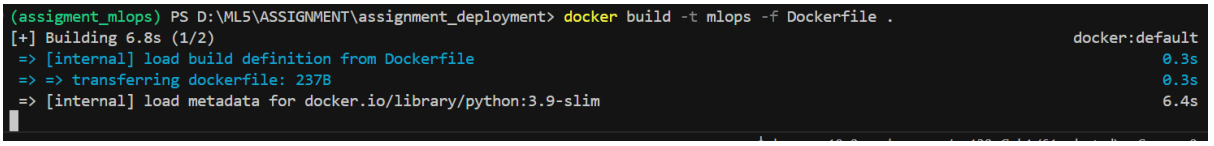
- Dockerfile untuk FastAPI dan Machine Learning Model

  ini akan digunakan untuk membangun image Docker yang berisi aplikasi FastAPI untuk memanggil model Machine Learning.

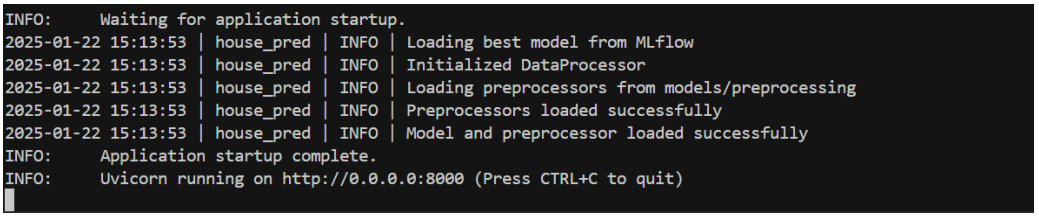
- docker-compose.yml untuk FastAPI dan Monitoring

  File docker-compose.yml digunakan untuk mengonfigurasi dan mengelola beberapa layanan yang berjalan di dalam container.

- Pastikan Docker Desktop sudah terinstal. Buka aplikasi Docker Desktop untuk memverifikasi bahwa Docker berjalan dengan baik di sistem Anda.
- 

- Setelah Docker Desktop berjalan, buka terminal dan build image. Proses build akan mengunduh image dasar (base image), menginstal dependensi, dan menyalin file aplikasi ke dalam image, menghasilkan image yang dapat dijalankan sebagai container.
- 

- Jalankan image Docker yang telah dibangun


INFO: Waiting for application startup.
2025-01-22 15:13:53 | house_pred | INFO | Loading best model from MLflow
2025-01-22 15:13:53 | house_pred | INFO | Initialized DataProcessor
2025-01-22 15:13:53 | house_pred | INFO | Loading preprocessors from models/preprocessing
2025-01-22 15:13:53 | house_pred | INFO | Preprocessors loaded successfully
2025-01-22 15:13:53 | house_pred | INFO | Model and preprocessor loaded successfully
INFO: Application startup complete.
INFO: Uvicorn running on http://0.0.0.0:8000 (Press CTRL+C to quit)