Nama: Andhika Putra Bagaskara

MLOps House Price Prediction Pipeline using Streamlit

House Price Prediction Pipeline merupakan pipeline yang bertujuan untuk memprediksi suatu harga rumah yang akan dijual dan kemudian akan ditampilkan prediksi tersebut dengan menggunakan Streamlit. Pipeline ini terbagi menjadi beberapa bagian yaitu:

1. dataLoader.py

Pada implementasi kode untuk *MLOps* prediksi harga rumah, dikembangkanlah suatu class bernama *DataLoader* yang bertujuan untuk proses pengelolaan data pelatihan dan pengujian. class ini dirancang untuk memuat data dari file CSV dan melakukan validasi awal guna memastikan data memenuhi syarat untuk digunakan dalam model. Konstruksi class DataLoader mendukung fleksibilitas dengan memungkinkan pengguna memberikan jalur data (*path*) secara manual atau menggunakan path yang diatur dalam konfigurasi default yang didefinisikan pada modul config.

Fungsi utama dalam class ini adalah *load_data,* yang bertanggung jawab untuk memuat data dari path yang ditentukan. Proses ini mencakup pembacaan dua file: data pelatihan dan data pengujian. Setelah data berhasil dimuat, fungsi ini mencatat informasi terkait dimensi data menggunakan logge. Jika terjadi kesalahan selama proses pemuatan, logger akan mencatat pesan kesalahan untuk memudahkan proses debugging.

Fungsi lainnya adalah *validate_data*, yang berfungsi untuk memvalidasi data yang telah dimuat. Validasi mencakup pemeriksaan terhadap keberadaan kolom yang diperlukan. Selain itu, fungsi ini juga melakukan pemeriksaan terhadap nilai *null* dan memberikan peringatan jika ditemukan data yang kosong.

2. data_processor.py

Pada bagian ini dikembangkanlah class *DataProcessor* yang dirancang untuk menangani proses preprocessing data, termasuk pengisian nilai kosong, pemilihan fitur penting, encoding data kategorikal, dan normalisasi fitur numerik. Fungsi utama *fit_transform* bertujuan untuk melakukan fitting dan transformasi data secara bersamaan. Proses ini melibatkan penghapusan kolom yang tidak relevan atau memiliki nilai kosong dalam jumlah besar, pengisian nilai kosong dengan teknik imputasi median untuk kolom numerik dan modus untuk kolom kategorikal, serta pemilihan fitur penting berdasarkan korelasi dengan variabel target SalePrice. Setelah pemilihan fitur, kolom kategorikal dienkode menggunakan *one-hot encoding*, sementara fitur numerik dinormalisasi menggunakan *MinMaxScaler*. Fungsi ini juga memisahkan data menjadi fitur (*features*) dan target sebelum menyimpan objek praproses ke dalam direktori yang telah disiapkan.

Fungsi *transform* dirancang untuk menerapkan proses praproses yang sama pada data baru menggunakan objek praproses yang telah dilatih sebelumnya. Fungsi ini memastikan bahwa data baru diproses dengan cara yang konsisten, termasuk proses penanganan nilai kosong, *one-hot encoding*, dan normalisasi.

Class ini juga dilengkapi dengan fungsi *save_preprocessors* dan *load_preprocessors* untuk menyimpan dan memuat objek praproses seperti scaler yang digunakan dalam proses transformasi data. Hal ini memungkinkan keberlanjutan proses tanpa perlu melakukan fitting ulang setiap kali pipeline digunakan.

3. model.py

Bagian ini dirancang untuk mengelola proses pembuatan dan pengelolaan model prediksi harga rumah. Implementasi ini mencakup dua kelas utama, yaitu ModelFactory dan HousePriceModel, yang berfungsi sebagai kerangka modular untuk memilih, membangun, dan mengelola model pembelajaran mesin.

Class *ModelFactory* bertanggung jawab sebagai pabrik pembuatan model (*model factory*), yang mendukung beberapa jenis model seperti LinearRegression, RandomForestRegressor, dan GradientBoostingRegressor. Fungsi *get_model_config* menyediakan konfigurasi setiap model, termasuk kelas model dan parameter yang

relevan, yang diambil dari file konfigurasi (*config*). Fungsi *create_model* memungkinkan pembuatan instance model berdasarkan tipe model.

Class *HousePriceModel* berfungsi sebagai kerangka kerja inti untuk membangun dan mengelola model prediksi harga rumah. Saat diinisialisasi, class ini menerima tipe model yang diinginkan sebagai parameter dan menggunakan *ModelFactory* untuk membuat instance model yang sesuai melalui fungsi build. Selain itu, kelas ini menyediakan fungsi *get_params* untuk mendapatkan parameter model. Mekanisme logging yang terintegrasi mencatat setiap langkah penting, termasuk kesalahan yang terjadi, untuk mendukung transparansi dan kemudahan debugging.

4. train.py

Pada bagian ini terdapat class *ModelTrainer* yang dirancang untuk melatih, mengevaluasi, dan mengelola model pembelajaran mesin dalam prediksi harga rumah. Class ini berperan penting dalam menangani seluruh siklus pelatihan model, termasuk logging, pemantauan, dan penyimpanan menggunakan MLflow sebagai kerangka manajemen eksperimen.

Class *ModelTrainer* diawali dengan inisialisasi yang mencakup pengaturan nama eksperimen MLflow dan konfigurasi pelacakan (*tracking*) MLflow melalui fungsi setup_mlflow. Dalam proses ini, *ModelTrainer* memastikan bahwa setiap eksperimen tercatat dengan baik di MLflow, sehingga memungkinkan analisis historis dan pengelolaan versi model.

Fungsi utama dalam kelas ini adalah *train_model*, yang bertugas melatih model berdasarkan tipe yang diberikan. Fungsi ini mencakup pembuatan model melalui *ModelFactory*, pelatihan model menggunakan data pelatihan, evaluasi kinerja menggunakan metrik seperti RMSE, MAE, dan R^2, serta logging parameter, metrik, dan model ke MLflow.

Fungsi *train_all_models* memungkinkan pelatihan seluruh model yang telah dikonfigurasi. Setelah semua model dilatih, fungsi *_select_best_model* memilih model terbaik berdasarkan nilai R^2. Mekanisme ini memberikan efisiensi dalam memilih model yang paling cocok untuk diterapkan.

Selain itu, kelas ini menyediakan fungsi *get_best_model* untuk mengambil informasi model terbaik dan *get_all_metrics* untuk mendapatkan metrik dari semua model yang telah dilatih. Dengan demikian, pengguna dapat dengan mudah membandingkan kinerja model yang telah dibangun.

5. run_pipeline.py

Bagian ini berfungsi untuk menjalankan seluruh alur lengkap pipeline prediksi harga rumah. Implementasi ini mengintegrasikan berbagai kompone, mulai dari memuat data, preprocessing, pelatihan model, hingga pelacakan eksperimen menggunakan MLflow. Kode ini dirancang untuk memastikan proses berjalan secara otomatis, terstruktur, dan terdokumentasi dengan baik.

Langkah pertama dalam pipeline adalah pengaturan MLflow melalui fungsi *setup_mlflow*. Fungsi ini memastikan bahwa URI pelacakan (*tracking URI*) dan eksperimen utama (*experiment*) dikonfigurasi dengan benar. Eksperimen yang dijalankan akan dicatat ke dalam database MLflow, memungkinkan pengguna untuk memantau proses dan hasil setiap eksekusi.

Selanjutnya, pipeline dimulai dengan memuat data menggunakan class *DataLoader*. Data yang dimuat divalidasi untuk memastikan kualitasnya memenuhi standar yang telah ditentukan. Informasi terkait data, seperti bentuk dan kolom, dicatat menggunakan MLflow untuk keperluan dokumentasi.

Pada tahap praproses, class *DataProcessor* digunakan untuk memproses data, termasuk pemilihan fitur penting, imputasi nilai kosong, encoding data kategorikal, dan normalisasi fitur numerik. Data hasil praproses kemudian dibagi menjadi data pelatihan dan validasi menggunakan fungsi *train_test_split*, memastikan bahwa model dilatih dan dievaluasi secara terpisah. Objek praproses disimpan untuk mendukung reusabilitas dalam alur kerja berikutnya.

Fungsi selanjutnya yaitu pelatihan dan evaluasi model menggunakan class *ModelTrainer*. Pipeline ini melatih beberapa model yang telah dikonfigurasi secara otomatis dan mengevaluasi performanya berdasarkan metrik seperti RMSE, MAE, dan R^2. Setiap model dicatat sebagai run terpisah dalam eksperimen MLflow, memungkinkan

analisis mendalam terhadap parameter dan kinerja model. Setelah semua model dilatih, model terbaik dipilih berdasarkan nilai R^2 dan dicatat sebagai model produksi dalam MLflow.

6. main.py (untuk FastAPI)

Bagian ini merupakan file yang digunakan untuk implementasi API berbasis FastAPI yang dirancang untuk menyediakan layanan prediksi harga rumah secara real-time. API ini mengintegrasikan model pembelajaran mesin yang telah dilatih sebelumnya dengan dukungan MLflow untuk pelacakan dan pengelolaan model. Pada saat API dijalankan, fungsi startup_event memastikan model terbaik dimuat berdasarkan metrik kinerja, seperti R^2, dari eksperimen di MLflow. Selain itu, objek praproses (*preprocessor*) yang digunakan selama pelatihan juga dimuat untuk menjaga konsistensi dalam pemrosesan data masukan.

API ini menyediakan beberapa endpoint utama. Endpoint root (/) memberikan pesan pengantar untuk memverifikasi bahwa API berjalan dengan benar. Endpoint prediksi (/predict) menerima data fitur rumah dalam format JSON, memproses data tersebut menggunakan preprocessor, dan menghasilkan prediksi harga rumah menggunakan model terbaik. Hasil prediksi disertai dengan informasi model yang digunakan, seperti metrik kinerja dan ID *run* di MLflow. Selain itu, tersedia endpoint kesehatan (/health) yang memberikan laporan status API, termasuk informasi model yang sedang aktif, memastikan API siap melayani permintaan.

Implementasi Streamlit

Setelah model machine learning sudah dibuat, kemudian model tersebut akan digunakan untuk melakukan prediksi dengan menggunakan UI streamlit. Aplikasi streamlit yang digunakan terbagi menjadi 5 halaman yaitu Home, Overview, Profile, Analytics, dan Predictions.

1. Home.py

Aplikasi *House Price Prediction* adalah platform berbasis *Streamlit* yang dirancang untuk menganalisis dan memprediksi harga rumah menggunakan dataset *House Prices*

dari Kaggle. Aplikasi ini menawarkan berbagai fitur, seperti eksplorasi data, analisis hubungan fitur terhadap harga, prediksi harga menggunakan model pembelajaran mesin, serta evaluasi performa model dengan metrik seperti MAE dan R^2 Score. Dengan antarmuka yang ramah pengguna, aplikasi ini membantu pengguna memahami pola data, memprediksi harga properti, dan mendukung pengambilan keputusan berbasis data dalam pasar perumahan.

2. Overview.py

Bagian ini bertujuan untuk menampilkan gambaran awal mengenai dataset yang digunakan, yaitu dataset *House Prices* dari Kaggle. Berikut adalah penjelasan detail setiap bagian dari kode:

1. **Import Library dan Modul**. Script ini memanfaatkan berbagai modul dan pustaka untuk mendukung pengembangan aplikasi:
   - streamlit: Digunakan untuk membuat antarmuka aplikasi berbasis web.
   - config, styling, dan logger: Modul tambahan dari folder src.utils untuk konfigurasi aplikasi, pengaturan gaya antarmuka, dan pencatatan log aktivitas aplikasi.
   - DataLoader: Modul untuk memuat dataset dan memproses data.
2. **Konfigurasi Halaman Aplikasi**. Fungsi st.set_page_config() digunakan untuk mengatur tampilan halaman aplikasi, seperti judul halaman (PAGE_TITLE), ikon (PAGE_ICON), dan tata letak (LAYOUT), yang diambil dari file konfigurasi config.
3. **Fungsi read_dataset()**. Fungsi ini bertujuan untuk memuat dataset pelatihan (df_train) dan dataset pengujian (df_test). Dataset ini akan digunakan untuk proses analisis dan prediksi.
   - **Inisialisasi Data**: Fungsi DataLoader() diinisialisasi untuk menyiapkan proses pemuatan data.
   - **Memuat Data**: Dataset pelatihan (df_train) dan dataset pengujian (df_test) dimuat menggunakan fungsi load_data() dari modul DataLoader.
   - **Log Aktivitas**: Proses pemuatan data dicatat menggunakan modul logger untuk memantau keberhasilan atau kendala yang terjadi.

4. **Pengaturan Style dengan load_css()**. Fungsi load_css() dipanggil untuk mengatur gaya antarmuka aplikasi agar lebih menarik.
5. **Visualisasi Overview Dataset**
    - **Kolom Statistik:** Dua kolom dibuat menggunakan st.columns() untuk menampilkan jumlah baris (jumlah_row) dan jumlah kolom (jumlah_columns) dari dataset pelatihan. Hasilnya ditampilkan dengan elemen st.metric().
    - **Preview Dataset**: Dataset pelatihan (df_train) ditampilkan dalam bentuk tabel menggunakan st.dataframe(), dengan menampilkan 20 baris pertama.

3. Profile.py

Bagian ini bertujuan untuk membuatprofil menggunakan *Streamlit*. Bagian ini dirancang untuk menampilkan informasi profil lengkap, termasuk foto, kontak, pengalaman kerja, keahlian, pendidikan, dan sertifikasi. Fungsi get_pdf_download_link menghasilkan tautan untuk mengunduh file PDF, seperti CV, dalam format yang dapat diakses melalui browser. File PDF dikonversi menjadi string *base64* untuk mendukung pengunduhan langsung.

4. Analytics

Bagian ini dirancang dengan tiga tab utama yaitu *Data Analysis* untuk menampilkan distribusi data dari fitur yang dipilih pengguna menggunakan histogram, box plot, dan statistik deskriptif. *Feature Relationships* digunakan untuk menganalisis hubungan antar fitur melalui matriks korelasi dan scatter plot, serta *Model Performance* untuk menampilkan metrik evaluasi model seperti R^2, RMSE, dan MAE, disertai analisis *feature importance* melalui diagram batang dan lingkaran. Dataset dimuat menggunakan modul DataLoader, sementara metrik dan *feature importance* diambil dari file JSON.