Pipeline Prediksi Harga Rumah

Pipeline ini dirancang untuk membantu memprediksi harga rumah dengan menggunakan data yang ada secara terstruktur dan sistematis. Seluruh proses mulai dari memuat data, membersihkan, melatih model, hingga memilih model terbaik telah disusun dengan baik. Selain itu, pipeline ini juga memanfaatkan **MLflow** untuk melacak eksperimen sehingga semua langkah mudah dipantau dan diulang.

Langkah-Langkah Pipeline

1. Persiapan Lingkungan

Langkah pertama adalah memastikan lingkungan sudah siap. Ini termasuk:

- Membuat folder yang diperlukan, seperti logs, models, dan data.
- Menghapus database lama **MLflow** jika ada, untuk memulai eksperimen dari awal.
- Mengatur **MLflow** berdasarkan konfigurasi di file config.yaml.
    - URI untuk melacak eksperimen.
    - Nama eksperimen, yaitu houseprice_prediction.
    - Tempat penyimpanan model dan hasil lainnya.

Langkah ini memastikan semua kebutuhan teknis sudah siap sebelum pipeline dijalankan.

---

2. Memuat Data

Data dimuat dari file CSV yang sudah ditentukan di config.yaml (default: artifacts/train.csv). Pada tahap ini, pipeline menggunakan DataLoader untuk:

- Memastikan data berhasil dimuat tanpa kesalahan.
- Mengecek apakah kolom yang dibutuhkan sudah tersedia, seperti kolom target SalePrice.
- Melaporkan jika ada data yang hilang atau nilai kosong.

Selain itu, jumlah baris, kolom, dan nama kolom akan dicatat ke **MLflow** untuk memastikan transparansi.

---

3. Preprocessing Data

Setelah data berhasil dimuat, proses berikutnya adalah preprocessing menggunakan DataProcessor. Di sini, pipeline memanfaatkan daftar fitur numerik dan kategorikal yang sudah ditentukan di config.yaml untuk membersihkan dan memproses data.

- **Fitur Numerikal**:
    - Nilai kosong diisi dengan 0 agar tidak mengganggu analisis.
    - Data distandarkan ke dalam skala 0-1 menggunakan **Min-Max Scaler**.
- **Fitur Kategorikal**:
    - Nilai kategori (misalnya MSZoning, Neighborhood, dll.) diubah menjadi angka menggunakan **LabelEncoder**.
- **Target (SalePrice)**:
    - Kolom target diubah ke skala logaritma dengan fungsi np.log10 untuk mengatasi outlier dan distribusi yang tidak normal.

Setelah preprocessing selesai, data dibagi menjadi dua bagian:

- **Training set (80%)** untuk melatih model.
- **Test set (20%)** untuk mengevaluasi model.

Preprocessing ini memastikan data siap digunakan untuk tahap berikutnya.

---

4. Melatih dan Mengevaluasi Model

Pada tahap ini, pipeline melatih beberapa model yang telah dikonfigurasi di config.yaml, yaitu:

1. **Decision Tree**
2. **Random Forest**
3. **XGBoost**

Untuk setiap model, prosesnya meliputi:

- Membuat model dengan parameter yang sudah ditentukan (misalnya max_depth, learning_rate, dll.).
- Melatih model pada data training.
- Membuat prediksi pada data test.
- Menghitung metrik evaluasi, seperti:
    - **RMSE** (Root Mean Squared Error): Seberapa besar kesalahan prediksi.
    - **MAE** (Mean Absolute Error): Rata-rata kesalahan prediksi.
    - **R^2 Score**: Seberapa baik model menjelaskan variabilitas data.

Hasil evaluasi ini dicatat ke **MLflow** untuk setiap model. Jika model mendukung *feature importance* (misalnya Random Forest dan XGBoost), pipeline juga akan menyimpan informasi ini.

---

5. Memilih Model Terbaik

Setelah semua model selesai dilatih, pipeline akan memilih model dengan nilai **R^2 Score** tertinggi. Model terbaik ini:

- Dicatat sebagai model terbaik di **MLflow**.
- Dipromosikan ke tahap produksi (*Production*) sehingga siap digunakan untuk prediksi di dunia nyata.
- Parameter model dan hasil evaluasinya juga disimpan.

---

6. Logging dan Pelacakan

Seluruh langkah pipeline dicatat menggunakan **MLflow**. Ini termasuk:

- Parameter model (seperti max_depth, n_estimators).
- Hasil preprocessing (jumlah data latih dan uji).
- Metrik evaluasi model.
- Model terbaik yang dipilih.

Dengan pelacakan ini, kita bisa dengan mudah membandingkan eksperimen, melihat model mana yang berkinerja lebih baik, atau mengulang proses jika diperlukan.

---

7. Integrasi API

Pipeline ini juga mendukung pembuatan API berdasarkan konfigurasi di schemas.py, main.py, dan config.yaml. API memungkinkan kita menggunakan model terbaik untuk melakukan prediksi secara real-time. Parameter API seperti host, port, dan versi API sudah diatur sehingga siap digunakan.

Alur Pipeline Secara Visual

1. Setup Lingkungan (MLflow, direktori)

    ↓

2. Load Data (DataLoader)

    ↓

3. Validasi dan Preprocessing (DataProcessor)

    ↓

4. Split Data (Train/Test)

    ↓

5. Train Models (Decision Tree, Random Forest, XGBoost)

↓

6. Evaluasi Model

↓

7. Pilih Model Terbaik

↓

8. Log dan Simpan Hasil (MLflow)

↓

9. Integrasi API untuk Prediksi

Keunggulan Pipeline

1. **Fleksibel**:
    - Seluruh parameter pipeline disimpan di config.yaml, sehingga kita bisa mengubahnya tanpa menyentuh kode utama.
2. **Mudah Dilacak**:
    - Semua langkah dicatat di **MLflow**, membuatnya mudah untuk dilacak, diulang, atau dibandingkan.
3. **Ekstensibel**:
    - Kita bisa menambahkan model baru atau memperluas pipeline dengan fitur lain hanya dengan menyesuaikan konfigurasi.
4. **Siap Produksi**:
    - Model terbaik siap dipromosikan untuk digunakan di aplikasi dunia nyata, bahkan mendukung REST API untuk prediksi real-time.

Pipeline ini memberikan solusi yang sistematis dan terstruktur untuk memprediksi harga rumah.