House Price Prediction Model - Report & Documentation

1. Objective

The goal of this project was to develop and deploy a machine learning model capable of predicting house prices based on various attributes. The process involved data preprocessing, model training and evaluation, inference testing, and deployment using FastAPI.

2. Dataset

- **Source**: Kaggle House Data

3. Data Preprocessing

Exploratory Data Analysis (EDA)

- **Size**: 4600 rows, 18 columns (13 numerical, 5 categorical)
- Examined correlation between features and price
- Visualized the correlation and outliers using Boxplot and Heatmaps
- **Observations**:
    - No missing values or duplicates
    - Presence of significant outliers
    - Some features moderately correlated with price

Feature Engineering

- **Location Encoding**: Categorical columns (`street`, `city`, `statezip`, `country`) were converted into geographic coordinates using the **Azure Maps API**.
- **Clustering**:
    - Optimal number of clusters determined (k=10) using silhouette score.
    - Created a new categorical feature `area_category` from cluster labels.
    - One-hot encoded `area_category` and merged into the main dataset.
- **Outlier Removal**:
    - IQR method was used to detect and remove extreme outliers.
    - Final dataset size reduced to **3727 rows**.
- **New Features**:
    - `yr_updated`: Maximum of `yr_built` and `yr_renovated`.
    - `is_renovated`: Binary indicator if house was renovated.
- **Feature Selection & Scaling**:
    - Relevant features selected for training.
    - MinMaxScaler applied for normalization.

4. Model Training & Evaluation

- **Algorithm Selection**:
    - Tried multiple regression models.
    - **GradientBoostingRegressor** provided the best results.
- **Hyperparameter Tuning**:
    - GridSearchCV with 5-fold cross-validation.
    - Found optimal parameters for best performance.
- **Train/Test Split**:
    - Data split into **85% training / 15% testing**.
    - Model trained using optimal parameters.
- **Evaluation Metrics**:
    - R^2 score - **75.88%**
    - Mean Absolute Error (MAE) - **72648**, which is **35.08%** of standard deviation
    - Mean Squared Error (MSE) - **10919533397**
- **Visualization**:
    - Scatterplots and error analysis plots used to visualize model performance.
- **Final Model Training**:
    - Model retrained on the full dataset.
    - Model, scalers, and coordinate mapping exported as `.pkl` files.

5. Inference Testing

- Model and pre-processing components loaded independently.
- Custom input values tested to ensure inference correctness.
- Functions modularized for clean and efficient inference.

6. Model Deployment

Framework & API

- **FastAPI** used for building the REST API.
- **Pydantic** for request validation.
- **Uvicorn** for running the application.

API & Web Interface

- **Schema**:
  - Defined `Pydantic` BaseModel schemas for input validation.
- **Prediction Flow**:
  - User inputs values via a web form.
  - JavaScript sends data as JSON to `/predict` endpoint.
  - API processes input, applies transformations, and returns a price prediction.
  - JavaScript updates the webpage with the predicted price.
- **Frontend**:
  - Clean HTML, CSS, and JavaScript template.
  - Input validation in the frontend to restrict invalid values.

Deployment

- **Local Testing**:
  - API successfully tested on localhost.
- **Hosting**:
  - Application deployed on **Render**.
  - Public URL: House Price Prediction App

Additional Notes

- Running the project **locally** requires setting up the `AZURE_MAPS_API_KEY` environment variable.
- Hosted version on **Render** can be accessed without API key requirements.
- **Future Improvements**:
  - Implement logging and improve error handling.
  - Integrate **DVC or MLflow** for model versioning.

Conclusion

This project successfully demonstrates the complete lifecycle of an ML model—from data preprocessing and training to deployment. The **GradientBoostingRegressor** model performed best, with an R2 score upto **75.88%** and was deployed as a web-based FastAPI application, providing users with a seamless way to predict house prices based on input attributes.