House Price Prediction - End-to-End Report

1. Introduction

This report provides a structured overview of the **House Price Prediction** project, detailing the approaches used for **data preprocessing, feature engineering, model training**, and **evaluation** using Machine Learning (ML) and Artificial Neural Networks (ANN).

---

2. Data Preprocessing

2.1 Data Loading & Initial Exploration

- The dataset was loaded using **pandas**.
- Displayed the first few rows (`df.head()`) to understand the structure.
- Checked for missing values using `missingno` and `df.isnull().sum()`.

2.2 Handling Missing Values

- Columns with high missing values were dropped.
- Numerical columns with moderate missing values were **filled with median/mean**.
- Categorical missing values were **filled with mode or 'None'**.

2.3 Data Visualization

- **Heatmap** (`seaborn.heatmap`) was used to check feature correlations.
- **Distribution plots** (`sns.distplot`) analyzed numerical variables.
- **Box plots** (`sns.boxplot`) helped in detecting outliers.

2.4 Feature Engineering

- **Skewed features** were transformed using `np.log1p()`.
- **Encoding categorical variables**:
  - One-hot encoding for nominal variables.
  - Label encoding for ordinal variables.
- **Feature scaling** applied using `StandardScaler`.

---

3. Model Training & Evaluation

3.1 Machine Learning Models

Several regression models were trained, including:

1. **Linear Regression**
2. **Random Forest Regressor**
3. **Gradient Boosting (XGBoost)**

Performance Evaluation Metrics:

- **RMSE (Root Mean Squared Error)**
- **R^2 Score**

Model	RMSE	R^2 Score
Linear Regression	9.650158357116333e-07	0.9999999999946964
Gradient Boosting	0.006164869968149626	0.9997835515493321
Random Forest	0.009346862976938224	0.999502448096361
XGBoost	0.026308936805809095	0.9960580307188373

3.2 Artificial Neural Network (ANN) Model

- Built an ANN model using **TensorFlow/Keras**.
- **Network Architecture:**
  - **Input layer**: 40 features
  - **Hidden layers**: Dense layers with ReLU activation
  - **Dropout layers**: Added after dense layers to prevent overfitting.
  - **Output layer**: Single neuron (for regression)
  - **Loss function**: Mean Squared Error (MSE)
  - **Optimizer**: Adam
- **Regularization Techniques Used:**
  - **Dropout Layers**: Applied to randomly deactivate neurons and reduce overfitting.
  - **Early Stopping**: Monitored validation loss to stop training when performance stopped improving.

ANN Training Setup:

- Model trained for **500 epochs** with **batch size 32 (default)**.
- Used **EarlyStopping** to stop training if validation loss did not improve for **330 epochs**.

ANN Performance:

- **MSE**: 0.0084
- **R^2 Score**: 0.9416