Lab 1

Task: Kaggle Competition: House Price Prediction

Here's a step-by-step explanation of how the code works:

- **Import Libraries:**
  - `pandas`, `numpy`, `seaborn`, `matplotlib`: For data manipulation and visualization.
  - `train_test_split`: To split the data into training and testing sets.
  - `LinearRegression`: To apply linear regression modeling.
  - `mean_squared_error`, `r2_score`: To evaluate the model's performance.
  - `StandardScaler`: To scale the features for better model performance.
  - `LabelEncoder`: To convert categorical variables into numerical values.
- **Load the Dataset:**
  - `df = pd.read_csv('Housing.csv')`: Loads the housing dataset from a CSV file into a pandas DataFrame.
- **Data Preprocessing:**
  - **Label Encoding**:
    - The categorical columns like `mainroad`, `guestroom`, etc., which have "yes/no" values are converted to binary values (`1` for "yes" and `0` for "no") using `LabelEncoder()`.
    - This is necessary as machine learning algorithms generally require numerical input.
- **Prepare Features and Target:**
  - `X = df.drop('price', axis=1)`: All columns except `price` are used as features.
  - `y = df['price']`: The `price` column is extracted as the target variable (what we want to predict).
- **Splitting the Data:**
  - `train_test_split(X, y, test_size=0.2, random_state=42)` splits the dataset into:
    - 80% for training (`X_train`, `y_train`).
    - 20% for testing (`X_test`, `y_test`).
  - `random_state=42` ensures that the split is reproducible.
- **Feature Scaling:**
  - `StandardScaler()` standardizes the features by removing the mean and scaling them to unit variance.
  - `X_train_scaled = scaler.fit_transform(X_train)` scales the training data.
  - `X_test_scaled = scaler.transform(X_test)` scales the testing data using the same scaling parameters as the training data.
- **Training the Model:**
  - A **Linear Regression model** is created and trained using the scaled training data: `model.fit(X_train_scaled, y_train)`.
- **Making Predictions:**

- The trained model is used to make predictions on the test data: `y_pred = model.predict(X_test_scaled)`.

- **Model Evaluation:**
  - **Mean Squared Error (MSE)** is computed to measure the average squared difference between the actual and predicted values: `mse = mean_squared_error(y_test, y_pred)`.
  - **R-squared (R^2)** is computed to measure the proportion of the variance in the target variable that is explained by the model: `r2 = r2_score(y_test, y_pred)`.
  - These values are printed to evaluate model performance.
- **Visualization:**
  - A scatter plot is created to visualize the relationship between the actual prices (`y_test`) and predicted prices (`y_pred`).
  - A **red line** representing perfect predictions is plotted where `actual = predicted`. Points close to this line indicate better predictions.

OUTPUT OF CODE: