Name of the Students:

Sujal Gupta **(202410116100214)**

Shivam Chaudhary **(202410116100199)**

Shubhranshu **(202410116100205)**

Branch: MCA

Section: D

Session: 2024-2025

Submitted to: Mrs Komal Salgotra

Predicting House Prices with supervised learning

Introduction:

Predicting house prices is a fundamental problem in real estate analytics and a classic example of regression in machine learning. The goal of this project is to develop a predictive model that estimates house prices based on various property features using a linear regression approach.

The dataset used contains information about houses, including features like the number of bedrooms, bathrooms, living space area (sqft_living), number of floors, the condition of the house, square footage above ground and in the basement, and the year the house was built. These features are commonly correlated with the price of a property and form the basis for training the regression model.

To begin with, the dataset is loaded and cleaned. Missing values in numerical columns are handled by replacing them with the respective column means to avoid bias or skewed results. A feature selection step is applied to choose the most relevant columns that are likely to influence the price significantly.

The dataset is then split into training and testing sets using an 80/20 ratio to ensure the model can be trained and evaluated on separate data. A Linear Regression model from scikit-learn is trained on the training set. The model attempts to learn the relationship between the input features and the target variable (price), minimizing the difference between actual and predicted values.

After training, predictions are made on the test set, and the performance is evaluated using the Root Mean Squared Error (RMSE), a standard metric for regression tasks that penalizes large errors more heavily.

To demonstrate the practical use of the model, a sample prediction is performed for a hypothetical new house with predefined features. This showcases the model's real-world application potential for price estimation.

The project also includes an in-depth visualization section:

- **Scatter plots** are generated to visually explore the relationship between individual features and house price.
- **A comparison plot** of actual vs. predicted prices helps evaluate how well the model is performing.
- **A residuals distribution plot** helps assess the prediction errors, ensuring they are randomly distributed and that no patterns are left unexplained.

Overall, this project highlights the application of machine learning in the real estate domain, focusing on data preprocessing, model training, evaluation, and visualization to deliver a robust price prediction tool. It serves as a foundational template for building more advanced models by incorporating techniques like feature engineering, regularization, or even deep learning in future iterations.

Methodology:

The methodology for this house price prediction project involves several sequential steps that follow a standard machine learning pipeline. Each step is designed to ensure data quality, model reliability, and interpretability of results.

1. Data Acquisition

The dataset is loaded from a CSV file (`/content/data.csv`). A try-except block is used to handle potential file errors gracefully. The dataset is assumed to include various features about residential properties, with `price` as the target variable.

2. Data Cleaning and Preprocessing

Handling missing data is a critical step. All missing values in numerical columns are filled using the mean of each respective column. This simple imputation ensures that no data points are lost due to missing entries and avoids bias in the regression model.

3. Feature Selection

From the complete dataset, a subset of features is selected that are expected to significantly influence the house price. These include:

- **bedrooms, bathrooms**: Basic descriptors of the home.
- **sqft_living, floors, condition**: Reflects usable space and property quality.
- **sqft_above, sqft_basement**: More detailed space breakdown.
- **yr_built**: Indicates age of the property.

These features are used to construct the input matrix `X`, while the target variable `y` is the price.

4. Data Splitting

To evaluate model performance, the dataset is split into training and testing sets using an 80/20 split. The training set is used to fit the model, while the test set is held out to assess how well the model generalizes to unseen data.

5. Model Training

A **Linear Regression** model from the `scikit-learn` library is used. This model fits a linear equation to the relationship between the features and the target variable. The goal is to minimize the sum of squared residuals between actual and predicted prices.

6. Model Evaluation

Once predictions are made on the test set, performance is measured using the **Root Mean Squared Error (RMSE)**, which quantifies the average magnitude of the prediction error. RMSE is a preferred metric for regression tasks because it penalizes larger errors more severely.

7. Price Prediction for New Data

To demonstrate practical usage, the model is used to predict the price of a hypothetical new house using custom input values for all selected features.

8. Visualization and Insight Extraction

Three types of visualizations are included:

- **Scatter plots** showing the relationship between individual features and house prices.
- **Actual vs. Predicted price plot**, which visually evaluates the model's performance.
- **Residuals histogram**, which helps detect bias or patterns in the model's errors.

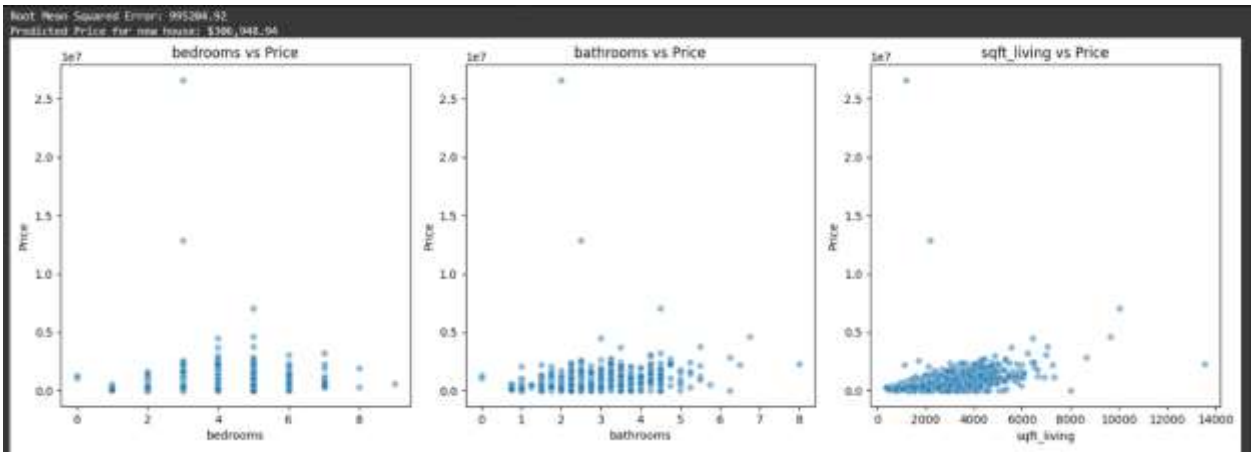
Code:

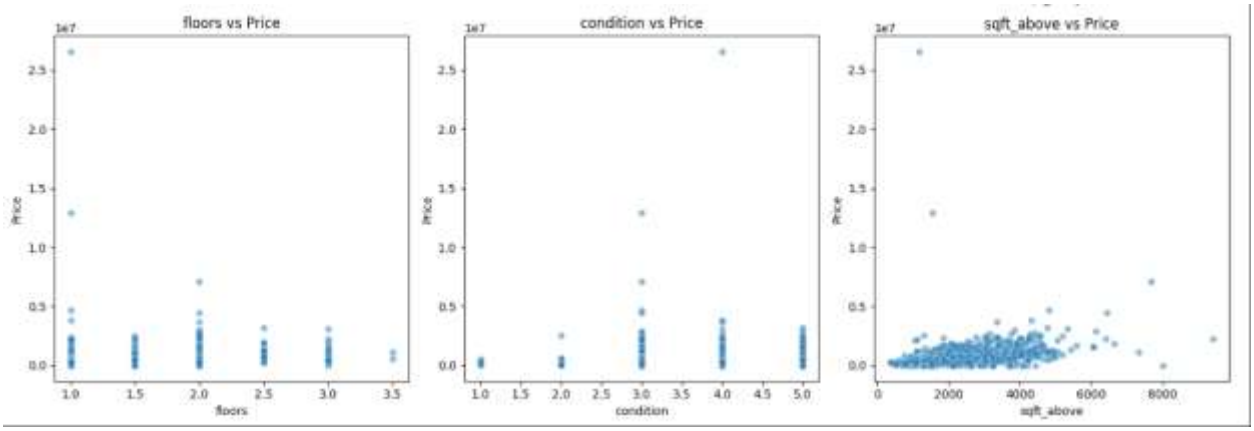
```
import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression

from sklearn.metrics import mean_squared_error


# Load the dataset

try:

    data = pd.read_csv('/content/data.csv')

except FileNotFoundError:

    print("Error: '/content/data.csv' not found. Please upload the file to your Colab environment.")

    exit()


# Handle missing values (replace with mean for numerical features)

numerical_cols = data.select_dtypes(include=['number']).columns

for col in numerical_cols:

    data[col].fillna(data[col].mean(), inplace=True)


# Feature Selection

features = ['bedrooms', 'bathrooms', 'sqft_living', 'floors', 'condition', 'sqft_above', 'sqft_basement', 'yr_built']

target = 'price'

X = data[features]

y = data[target]
```

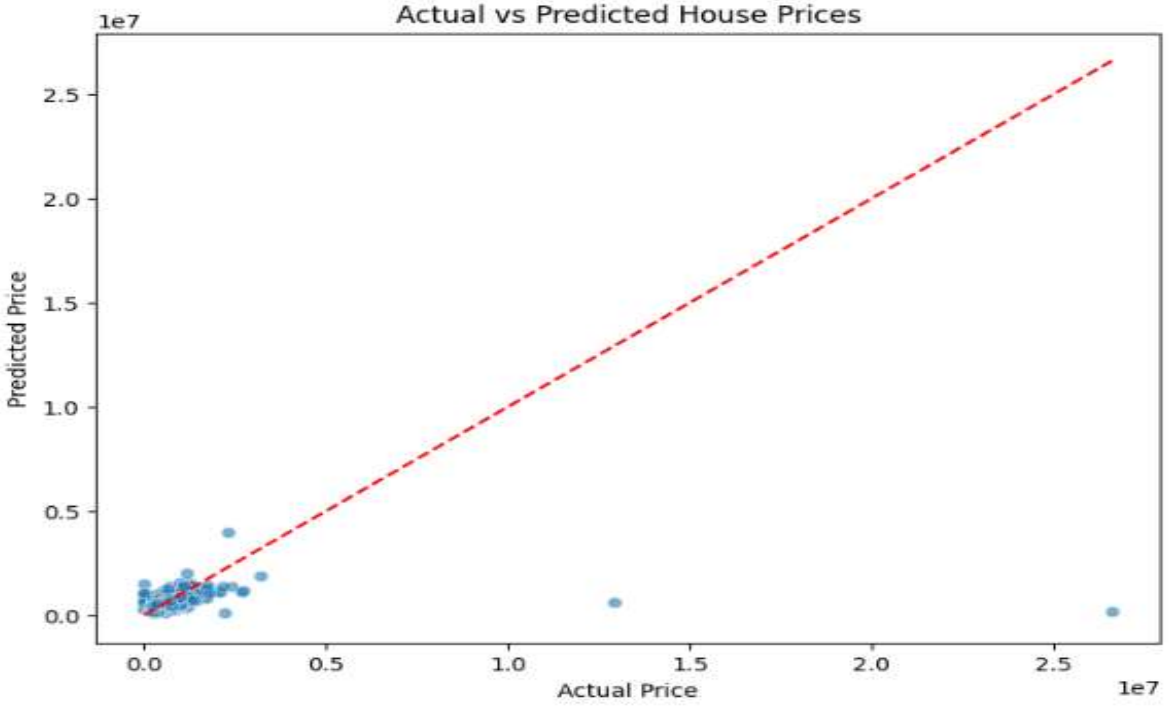
```
# Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train model
model = LinearRegression()
model.fit(X_train, y_train)

# Predict
y_pred = model.predict(X_test)

# Evaluate
mse = mean_squared_error(y_test, y_pred)
rmse = mse**0.5
print(f"Root Mean Squared Error: {rmse:.2f}")

# Predict new house
new_house_features = pd.DataFrame([[3, 2.0, 1600, 1.0, 4, 1500, 100, 1995]],
columns=features)
predicted_price = model.predict(new_house_features)
print(f"Predicted Price for new house: ${predicted_price[0]:,.2f}")

# 📈 Visualization Section
# 1. Feature vs Price scatter plots
plt.figure(figsize=(15, 10))
for i, feature in enumerate(features[:6]):
    plt.subplot(2, 3, i + 1)
    sns.scatterplot(x=data[feature], y=data['price'], alpha=0.5)
    plt.title(f'{feature} vs Price')
```

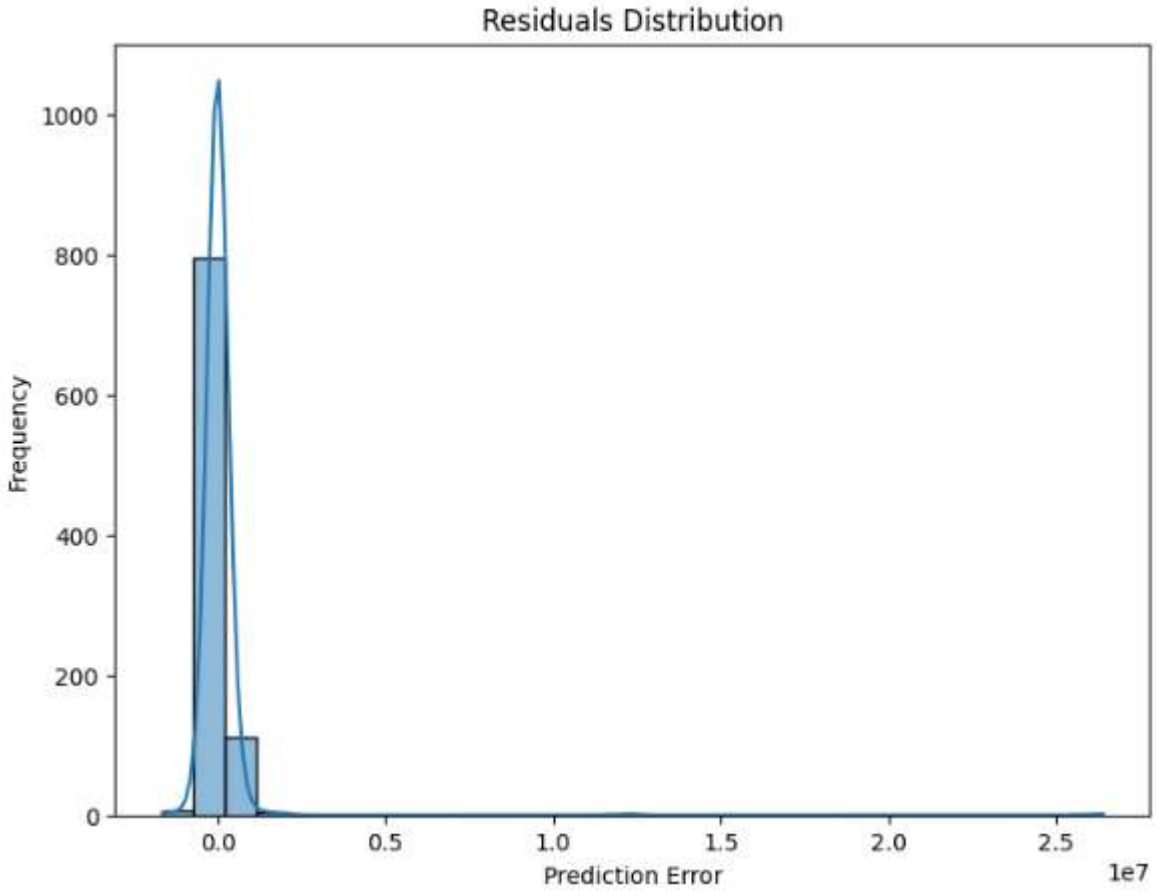
```
    plt.xlabel(feature)
    plt.ylabel('Price')
plt.tight_layout()
plt.show()

# 2. Actual vs Predicted Prices
plt.figure(figsize=(8, 6))
sns.scatterplot(x=y_test, y=y_pred, alpha=0.6)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], '--r')
plt.xlabel('Actual Price')
plt.ylabel('Predicted Price')
plt.title('Actual vs Predicted House Prices')
plt.show()


# 3. Residuals plot
residuals = y_test - y_pred
plt.figure(figsize=(8, 6))
sns.histplot(residuals, kde=True, bins=30)
plt.title('Residuals Distribution')
plt.xlabel('Prediction Error')
plt.ylabel('Frequency')
plt.show()
```

Output:


Root Mean Squared Error: 995204.92
Predicted Price for new house: $300,948.94
bedrooms vs Price
bathrooms vs Price
sqft_living vs Price
Price
bedrooms
bathrooms
sqft_living


floors vs Price
condition vs Price
sqft_above vs Price
Price
floors
condition
sqft_above


Actual vs Predicted House Prices
Predicted Price
Actual Price
1e7


Residuals Distribution
Frequency
1000
800
600
400
200
0
0.0
0.5
1.0
1.5
2.0
2.5
Prediction Error
1e7