This Python code implements a **linear regression model** to predict house prices based on three key features: **square footage (GrLivArea)**, **number of bedrooms (BedroomAbvGr)**, and **number of full bathrooms (FullBath)**. Below is a step-by-step explanation:

**1. Import Necessary Libraries**

import pandas as pd

import numpy as np

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, r2\_score

* pandas: Handles data manipulation.
* numpy: Provides numerical computing capabilities.
* matplotlib.pyplot: Used for plotting graphs.
* seaborn: Advanced visualization library (though not explicitly used here).
* sklearn.model\_selection.train\_test\_split: Splits the dataset into training and testing sets.
* sklearn.linear\_model.LinearRegression: Implements linear regression.
* sklearn.metrics.mean\_squared\_error, r2\_score: Evaluate model performance.

**2. Load the Dataset**

df = pd.read\_csv(r'C:\Users\sathi\OneDrive\Desktop\WORKSPACE\Prodigy\TASK-1\house-prices-advanced-regression-techniques\train.csv')

* Loads the **House Prices** dataset from a CSV file into a **pandas DataFrame**.

**3. Select Relevant Features**

df = df[['GrLivArea', 'BedroomAbvGr', 'FullBath', 'SalePrice']]

* Extracts only the necessary columns:
  + **GrLivArea** → Total square footage of the house.
  + **BedroomAbvGr** → Number of bedrooms.
  + **FullBath** → Number of full bathrooms.
  + **SalePrice** → Target variable (house price).

**4. Handle Missing Values**

df.dropna(inplace=True)

* **Removes** any rows that contain missing values to avoid errors during model training.

**5. Define Features (X) and Target (y)**

X = df[['GrLivArea', 'BedroomAbvGr', 'FullBath']]

y = df['SalePrice']

* X contains the **independent variables** (features used for prediction).
* y contains the **dependent variable** (house price).

**6. Split Data into Training and Testing Sets**

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

* Splits the dataset into:
  + **80% training data** (X\_train, y\_train).
  + **20% testing data** (X\_test, y\_test).
* random\_state=42 ensures reproducibility.

**7. Initialize the Linear Regression Model**

model = LinearRegression()

* Creates an instance of the LinearRegression model.

**8. Train the Model**

model.fit(X\_train, y\_train)

* Fits (trains) the model using the **training data**.

**9. Make Predictions on the Test Set**

y\_pred = model.predict(X\_test)

* Uses the trained model to predict house prices for the test dataset.

**10. Evaluate Model Performance**

**Calculate Mean Squared Error (MSE)**

mse = mean\_squared\_error(y\_test, y\_pred)

* MSE measures the average squared difference between **actual** and **predicted** prices.
  + Lower MSE → Better accuracy.

**Calculate R-squared (R²) Score**

r2 = r2\_score(y\_test, y\_pred)

* R² score represents how well the model explains the variance in house prices.
  + Closer to **1** → Better fit.
  + Closer to **0** → Poor model.

**11. Display Results**

print(f"Mean Squared Error (MSE): {mse}")

print(f"R-squared (R²): {r2}")

* Prints the **MSE** and **R² score** to assess model performance.

**12. Visualize Actual vs Predicted Prices**

plt.scatter(y\_test, y\_pred)

plt.xlabel('Actual SalePrice')

plt.ylabel('Predicted SalePrice')

plt.title('Actual vs Predicted SalePrice')

plt.show()

* Creates a **scatter plot** to compare actual vs. predicted prices.
  + **Perfect predictions** would lie along a straight diagonal line.

**13. Display Model Coefficients and Intercept**

print(f"Coefficients: {model.coef\_}")

print(f"Intercept: {model.intercept\_}")

* model.coef\_ → The impact of each feature on house price.
* model.intercept\_ → The base price when all features are **zero**.

**Final Summary**

1. **Loads & preprocesses data** (removes missing values).
2. **Selects key features** for predicting house prices.
3. **Splits data** into training & testing sets.
4. **Trains a linear regression model** on the training data.
5. **Predicts house prices** on the test set.
6. **Evaluates model performance** using MSE & R².
7. **Visualizes predictions** using a scatter plot.
8. **Displays model coefficients** to understand feature importance.

This is a **basic house price prediction model** using **linear regression**. 🚀