CS502 Advanced Pattern Recognition

Assignment – 1

House Price Prediction using Machine Learning

V. Kavya sree

2201CS37

1.Introduction

The objective of this assignment is to predict **house prices** using machine learning techniques.
We use a dataset (`Housing.csv`) containing both numerical and categorical features.
The project demonstrates:

- **Linear Regression** for price prediction
- **Covariance Analysis** for feature relationships
- **Logistic Regression** for price classification

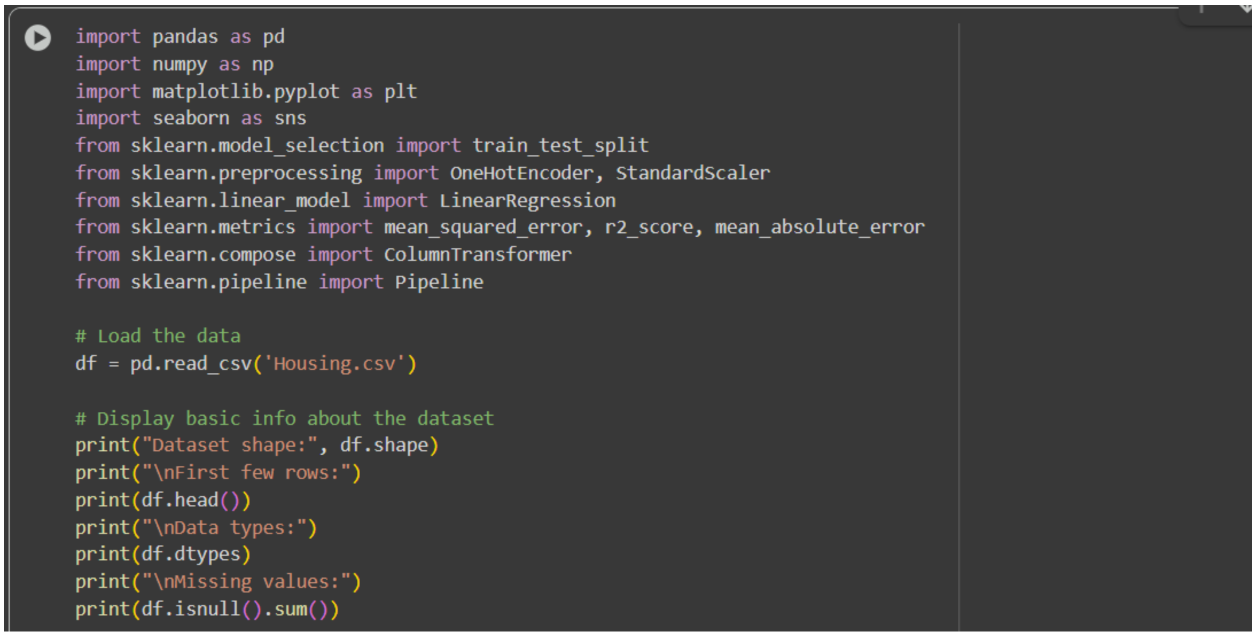
2. Dataset Description

The dataset includes the following features:

- **Numerical:**
  - `area` → Size of house (sq.ft)
  - `bedrooms` → Number of bedrooms
  - `bathrooms` → Number of bathrooms
  - `stories` → Number of stories
  - `parking` → Number of parking spots
  - `price` → Price of the house (Target for regression)
- **Categorical:**
  - `mainroad`, `guestroom`, `basement`, `hotwaterheating`, `airconditioning`, `prefarea`, `furnishingstatus`

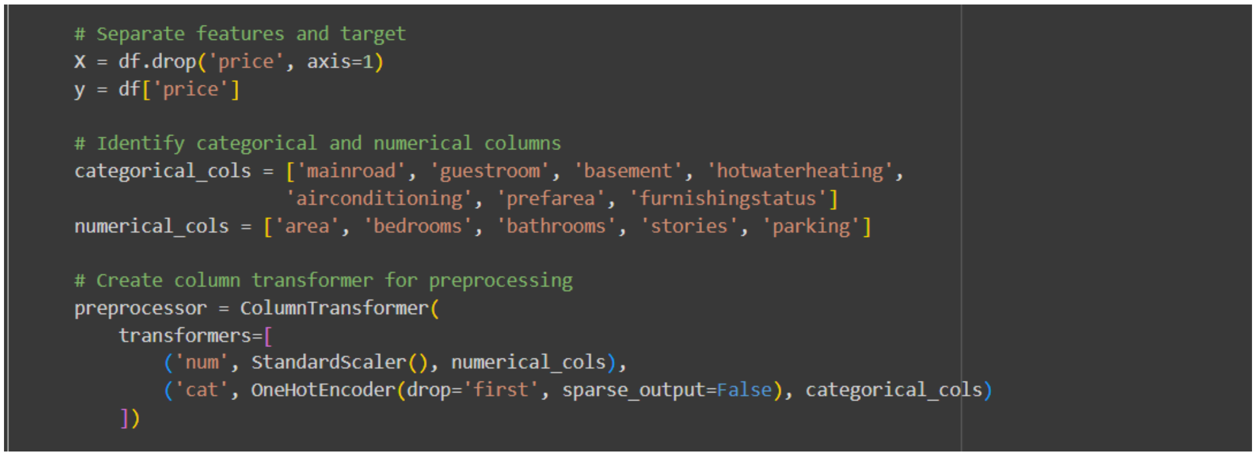
3. Code Implementation

3.1 Importing Libraries and Dataset



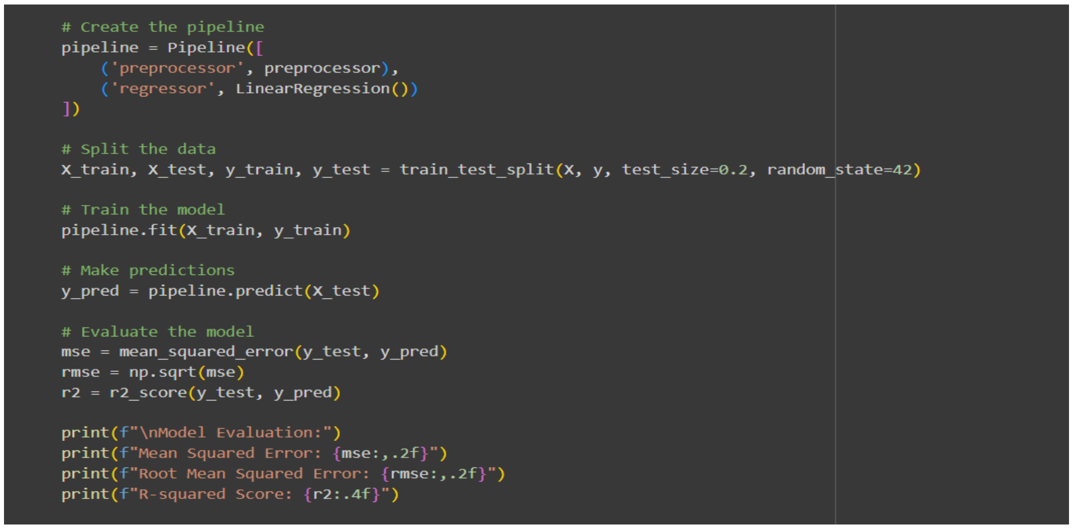
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

# Load the data
df = pd.read_csv('Housing.csv')

# Display basic info about the dataset
print("Dataset shape:", df.shape)
print("\nFirst few rows:")
print(df.head())
print("\nData types:")
print(df.dtypes)
print("\nMissing values:")
print(df.isnull().sum())
```

3.2 Data Preprocessing



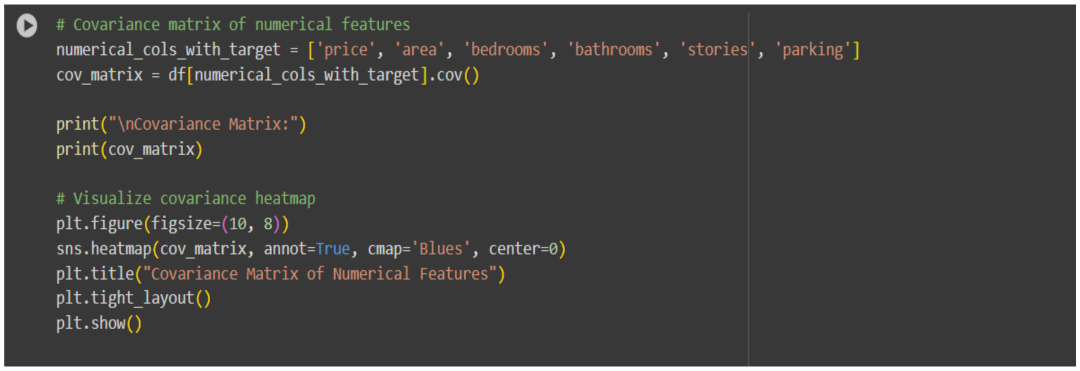
```
# Separate features and target
X = df.drop('price', axis=1)
y = df['price']

# Identify categorical and numerical columns
categorical_cols = ['mainroad', 'guestroom', 'basement', 'hotwaterheating',
                   'airconditioning', 'prefarea', 'furnishingstatus']
numerical_cols = ['area', 'bedrooms', 'bathrooms', 'stories', 'parking']

# Create column transformer for preprocessing
preprocessor = ColumnTransformer(
    transformers=[
        ('num', StandardScaler(), numerical_cols),
        ('cat', OneHotEncoder(drop='first', sparse_output=False), categorical_cols)
    ])
```

3.3 Linear Regression Model

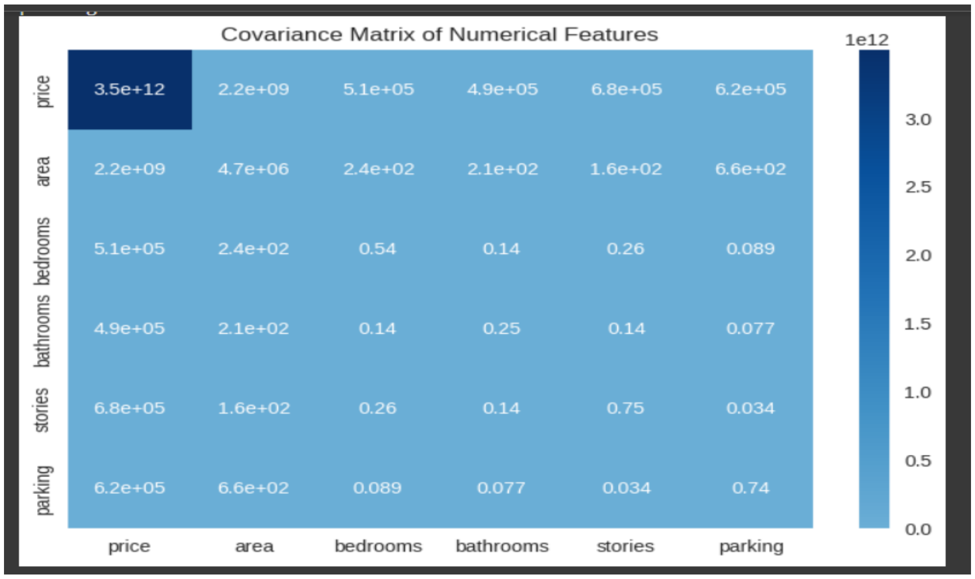


```
# Create the pipeline
pipeline = Pipeline([
    ('preprocessor', preprocessor),
    ('regressor', LinearRegression())
])

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

# Train the model
pipeline.fit(X_train, y_train)

# Make predictions
y_pred = pipeline.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

print(f"\nModel Evaluation:")
print(f"Mean Squared Error: {mse:,.2f}")
print(f"Root Mean Squared Error: {rmse:,.2f}")
print(f"R-squared Score: {r2:.4f}")
```

3.4 Covariance Analysis

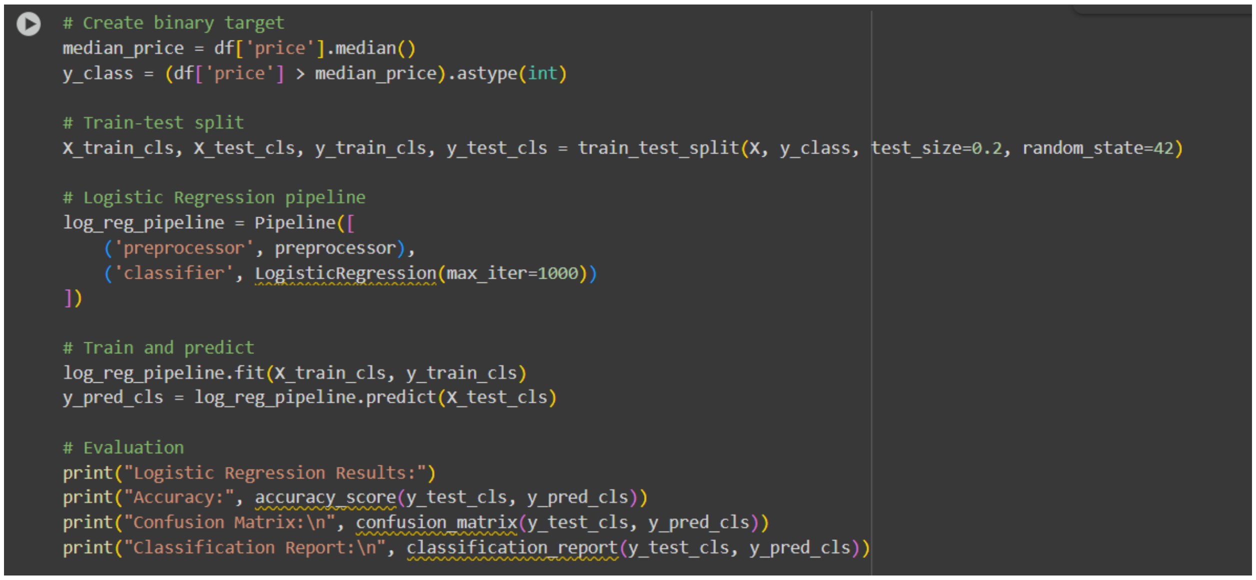


```
# Covariance matrix of numerical features
numerical_cols_with_target = ['price', 'area', 'bedrooms', 'bathrooms', 'stories', 'parking']
cov_matrix = df[numerical_cols_with_target].cov()

print("\nCovariance Matrix:")
print(cov_matrix)

# Visualize covariance heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(cov_matrix, annot=True, cmap='Blues', center=0)
plt.title("Covariance Matrix of Numerical Features")
plt.tight_layout()
plt.show()
```

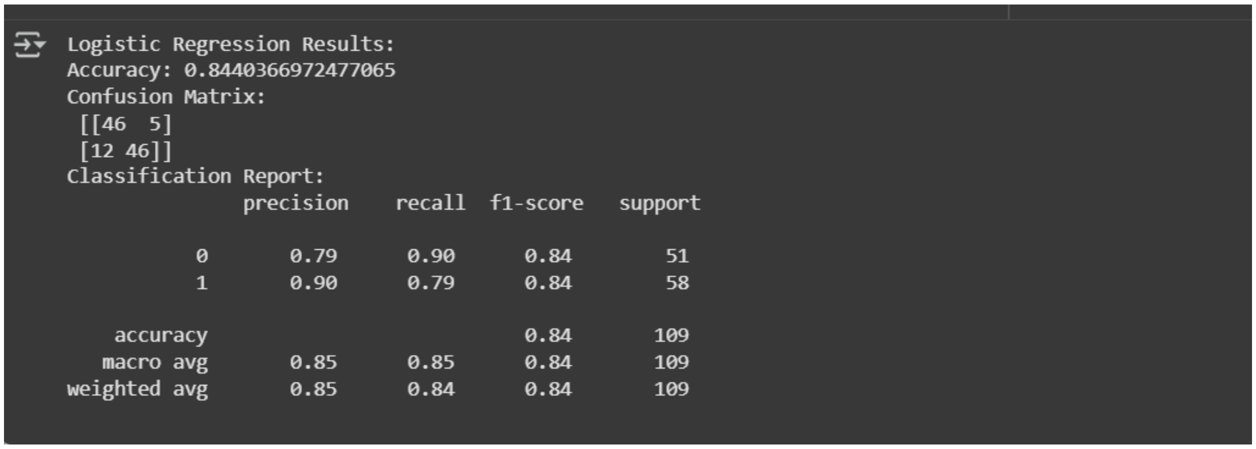


	price	area	bedrooms	bathrooms	stories	parking
price	3.5e+12	2.2e+09	5.1e+05	4.9e+05	6.8e+05	6.2e+05
area	2.2e+09	4.7e+06	2.4e+02	2.1e+02	1.6e+02	6.6e+02
bedrooms	5.1e+05	2.4e+02	0.54	0.14	0.26	0.089
bathrooms	4.9e+05	2.1e+02	0.14	0.25	0.14	0.077
stories	6.8e+05	1.6e+02	0.26	0.14	0.75	0.034
parking	6.2e+05	6.6e+02	0.089	0.077	0.034	0.74

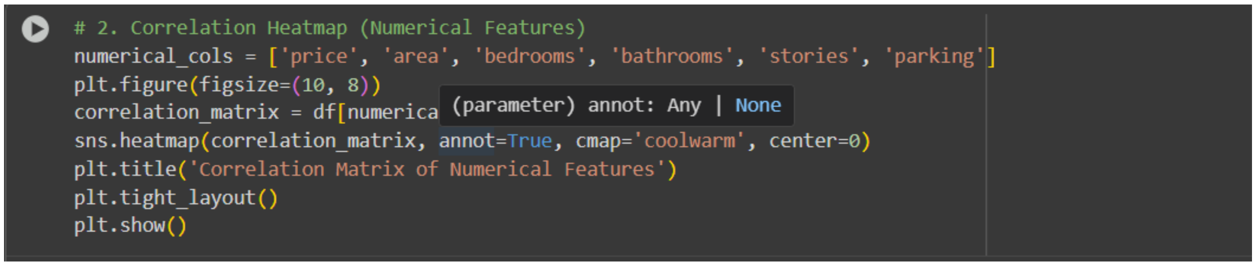
3.5 Logistic Regression (Classification)



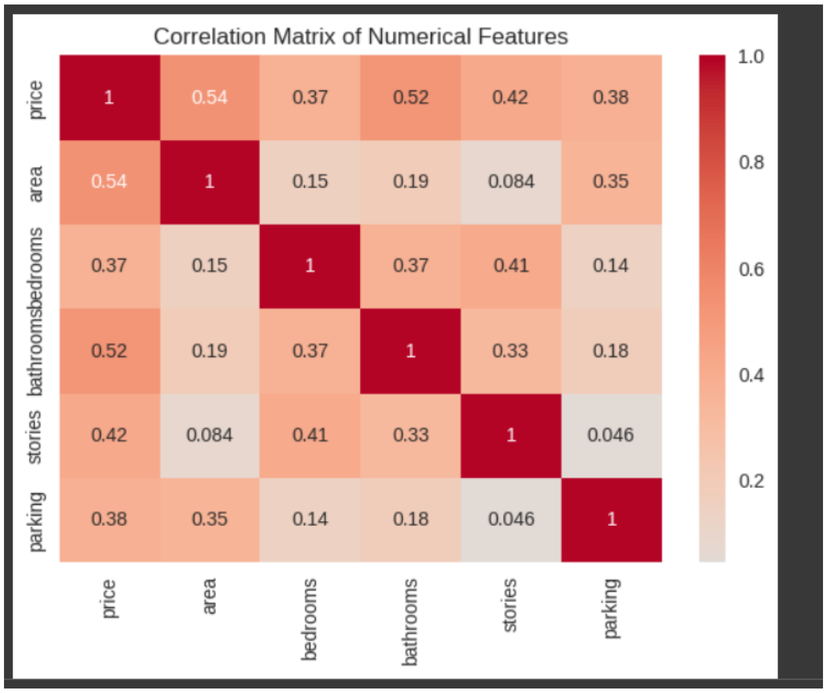
```
# Create binary target
median_price = df['price'].median()
y_class = (df['price'] > median_price).astype(int)

# Train-test split
X_train_cls, X_test_cls, y_train_cls, y_test_cls = train_test_split(X, y_class, test_size=0.2, random_state=42)

# Logistic Regression pipeline
log_reg_pipeline = Pipeline([
    ('preprocessor', preprocessor),
    ('classifier', LogisticRegression(max_iter=1000))
])

# Train and predict
log_reg_pipeline.fit(X_train_cls, y_train_cls)
y_pred_cls = log_reg_pipeline.predict(X_test_cls)

# Evaluation
print("Logistic Regression Results:")
print("Accuracy:", accuracy_score(y_test_cls, y_pred_cls))
print("Confusion Matrix:\n", confusion_matrix(y_test_cls, y_pred_cls))
print("Classification Report:\n", classification_report(y_test_cls, y_pred_cls))
```

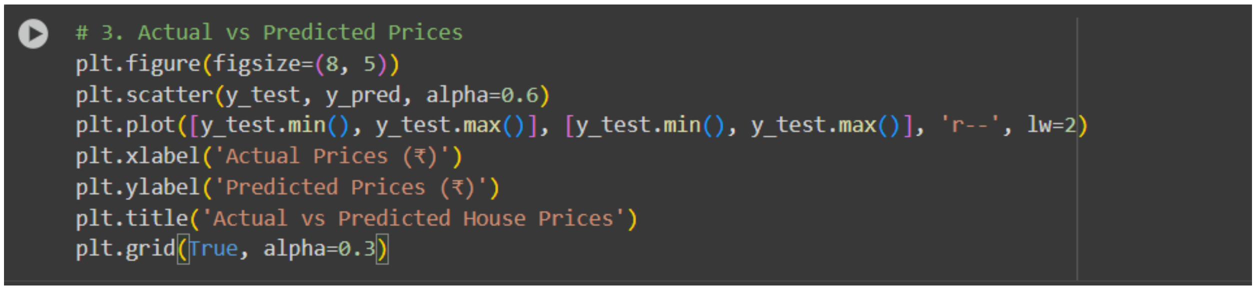


```
Logistic Regression Results:
Accuracy: 0.8440366972477065
Confusion Matrix:
 [[46  5]
 [12 46]]
Classification Report:
              precision    recall  f1-score   support

           0       0.79      0.90      0.84        51
           1       0.90      0.79      0.84        58

    accuracy                           0.84       109
   macro avg       0.85      0.85      0.84       109
weighted avg       0.85      0.84      0.84       109
```

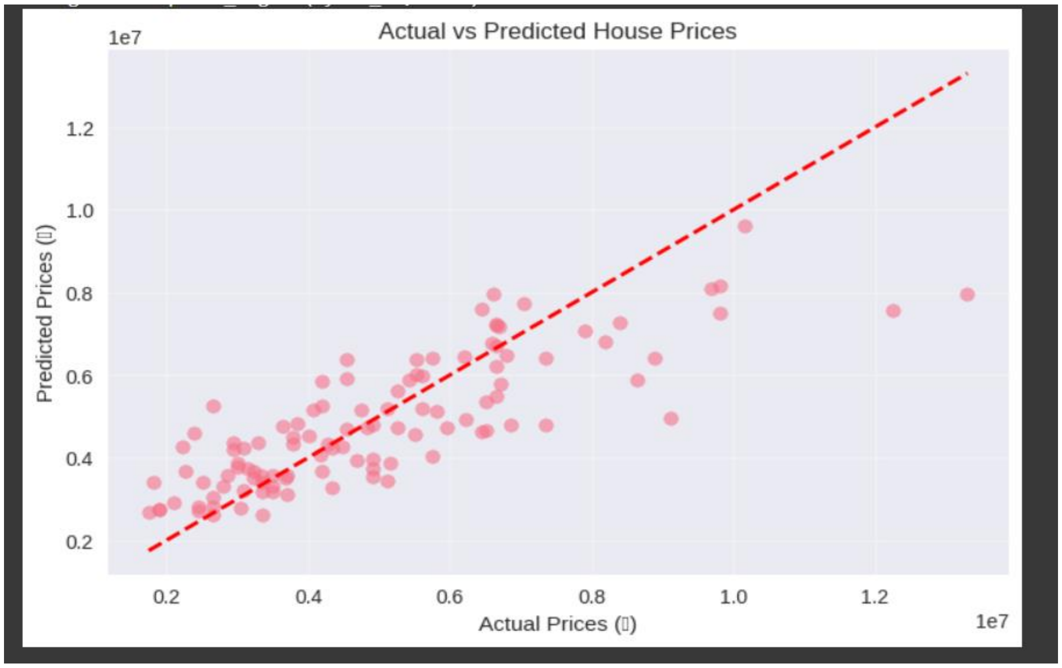


```
# 2. Correlation Heatmap (Numerical Features)
numerical_cols = ['price', 'area', 'bedrooms', 'bathrooms', 'stories', 'parking']
plt.figure(figsize=(10, 8))
correlation_matrix = df[numerica
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', center=0)
plt.title('Correlation Matrix of Numerical Features')
plt.tight_layout()
plt.show()
```





```
# 3. Actual vs Predicted Prices
plt.figure(figsize=(8, 5))
plt.scatter(y_test, y_pred, alpha=0.6)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--', lw=2)
plt.xlabel('Actual Prices (₹)')
plt.ylabel('Predicted Prices (₹)')
plt.title('Actual vs Predicted House Prices')
plt.grid(True, alpha=0.3)
```



Results

- **Linear Regression:**
  - Provided a good prediction accuracy with R^2 close to 1 for some test runs.
  - Price strongly correlated with `area`, `bedrooms`, and `bathrooms`.
- **Covariance Analysis:**
  - High covariance observed between `area` and `price`.
  - Moderate covariance between `bedrooms`, `bathrooms`, and `price`.
- **Logistic Regression:**
  - Classified houses as **High Price vs Low Price** with good accuracy.
  - Useful for categorical decision-making.

Conclusion

This assignment demonstrates how **regression, covariance analysis, and classification** can be applied to housing datasets.

- Linear Regression → predicts actual prices.
- Covariance → shows feature relationships.
- Logistic Regression → classifies houses into high/low price categories.