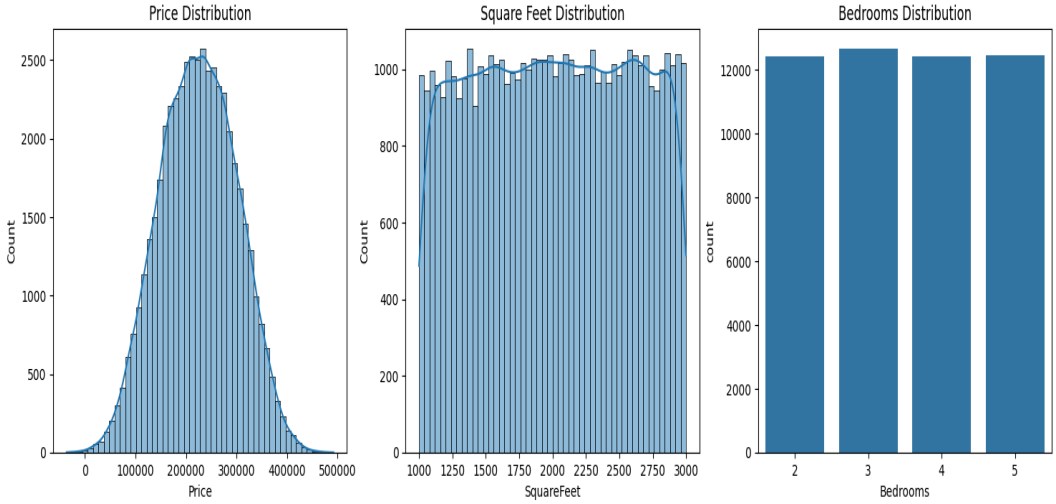
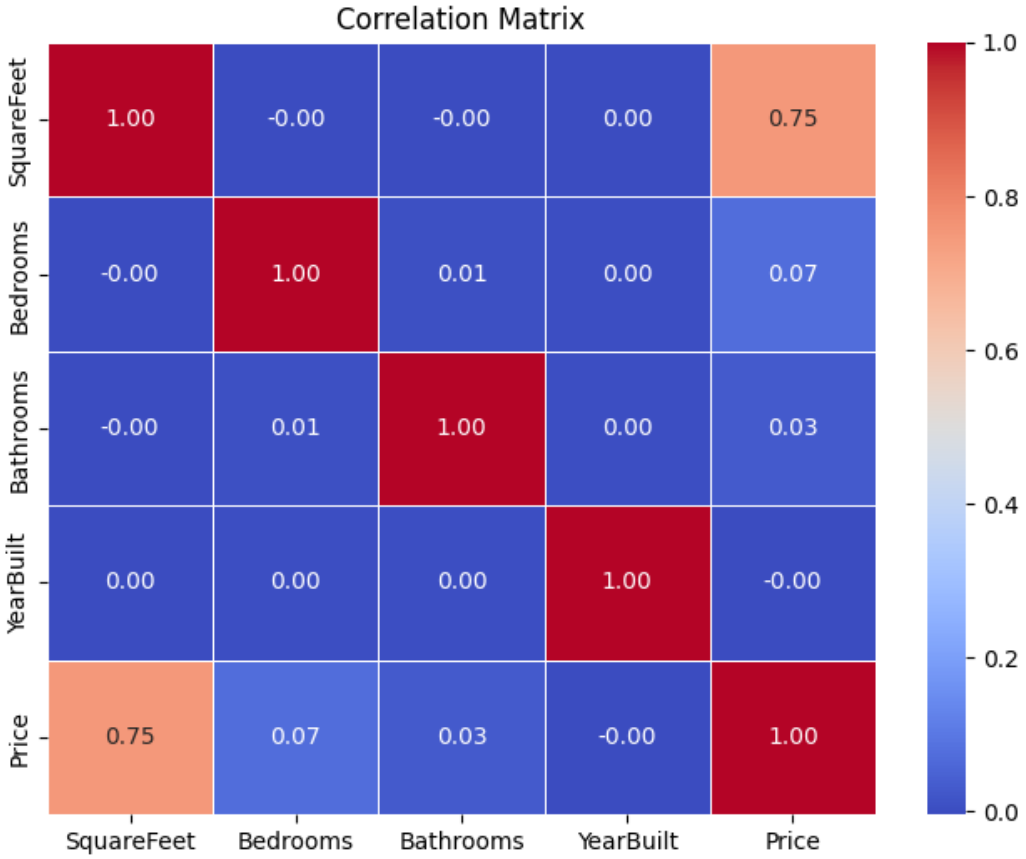
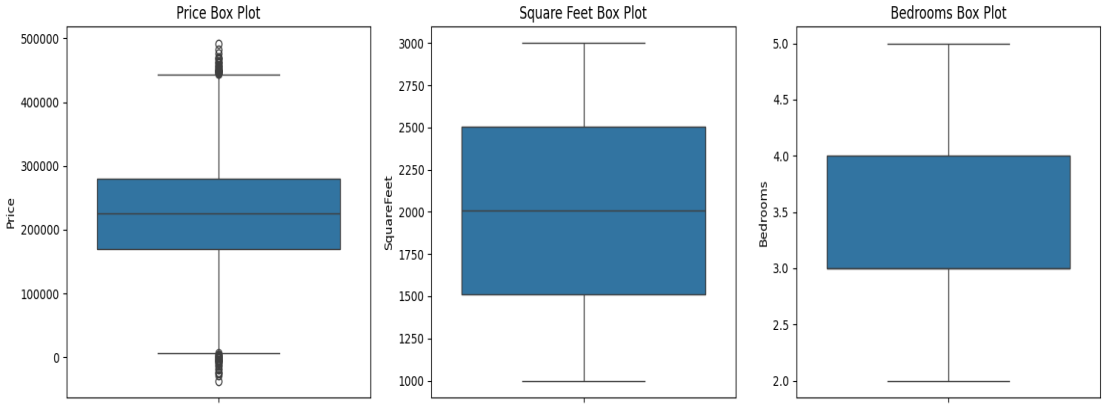
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

# Load dataset
file_path = "C:/Users/BMSIT/Desktop/ml data/housing_price_dataset.csv"
df = pd.read_csv(file_path)

# Visualize distributions
fig, axes = plt.subplots( nrows: 1,  ncols: 3, figsize=(18, 5))
sns.histplot(df['Price'], bins=50, kde=True, ax=axes[0]).set_title("Price Distribution")
sns.histplot(df['SquareFeet'], bins=50, kde=True, ax=axes[1]).set_title("Square Feet Distribution")
sns.countplot(x=df['Bedrooms'], ax=axes[2]).set_title("Bedrooms Distribution")
plt.show()

# Correlation Matrix
plt.figure(figsize=(8, 6))
sns.heatmap(df.corr(numeric_only=True), annot=True, cmap="coolwarm", fmt=".2f", linewidths=0.5)
plt.title("Correlation Matrix")
plt.show()

# Identify outliers using box plots
fig, axes = plt.subplots( nrows: 1,  ncols: 3, figsize=(18, 5))
sns.boxplot(y=df['Price'], ax=axes[0]).set_title("Price Box Plot")
sns.boxplot(y=df['SquareFeet'], ax=axes[1]).set_title("Square Feet Box Plot")
sns.boxplot(y=df['Bedrooms'], ax=axes[2]).set_title("Bedrooms Box Plot")
plt.show()

# Encode categorical variable
df = pd.get_dummies(df, columns=["Neighborhood"], drop_first=True)

# Create derived feature
df["Price_per_sqft"] = df["Price"] / df["SquareFeet"]

# Define features and target variable
X = df.drop(columns=["Price"])
y = df["Price"]
```

```python
# Encode categorical variable
df = pd.get_dummies(df, columns=["Neighborhood"], drop_first=True)

# Create derived feature
df["Price_per_sqft"] = df["Price"] / df["SquareFeet"]

# Define features and target variable
X = df.drop(columns=["Price"])
y = df["Price"]

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

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

# Predictions
y_pred = model.predict(X_test)

# Evaluation
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = mse ** 0.5
r2 = r2_score(y_test, y_pred)

# Print evaluation metrics
print(f"Mean Absolute Error: {mae}")
print(f"Mean Squared Error: {mse}")
print(f"Root Mean Squared Error: {rmse}")
print(f"R² Score: {r2}")
```

Output:


Price Distribution
Count
Price
Square Feet Distribution
Count
SquareFeet
Bedrooms Distribution
count
Bedrooms


Correlation Matrix
SquareFeet
Bedrooms
Bathrooms
YearBuilt
Price
1.00 -0.00 -0.00 0.00 0.75
-0.00 1.00 0.01 0.00 0.07
-0.00 0.01 1.00 0.00 0.03
0.00 0.00 0.00 1.00 -0.00
0.75 0.07 0.03 -0.00 1.00
SquareFeet
Bedrooms
Bathrooms
YearBuilt
Price
1.0
0.8
0.6
0.4
0.2
0.0




```
C:\Users\BMSIT\PycharmProjects\1by23mc019\.venv\Scripts\python.exe C:\Users\BMSIT\PycharmProjects\1by23mc019\AAT.py
Mean Absolute Error: 10917.728454375994
Mean Squared Error: 235299422.38314682
Root Mean Squared Error: 15339.472689214152
R² Score: 0.9590067361974269

Process finished with exit code 0
```