A Linear Regression to Predict House Prices by Matindi Steve - Perfomance of The Model is Analyzed Using MSE, MAE, & RMSE

March 30, 2024

[2]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_error
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

[3]:
```
# Loading our data
data = pd.read_csv("./dataset/Real estate.csv")
print(data.columns)
```

```
Index(['No', 'X1 transaction date', 'X2 house age',
       'X3 distance to the nearest MRT station',
       'X4 number of convenience stores', 'X5 latitude', 'X6 longitude',
       'Y house price of unit area'],
      dtype='object')
```

[4]:
```
# Step 3: Data preprocessing
# Check for missing values
missing_values = data.isnull().sum()
print("Missing values:")
print(missing_values)
```

```
Missing values:
No                                        0
X1 transaction date                       0
X2 house age                              0
X3 distance to the nearest MRT station    0
X4 number of convenience stores           0
X5 latitude                               0
X6 longitude                              0
Y house price of unit area                0
dtype: int64
```

[5]:
```
# Check data types of columns
data_types = data.dtypes
```

```python
print("\nData types:")
print(data_types)
```

```
Data types:
No                                           int64
X1 transaction date                        float64
X2 house age                               float64
X3 distance to the nearest MRT station    float64
X4 number of convenience stores             int64
X5 latitude                                float64
X6 longitude                               float64
Y house price of unit area                 float64
dtype: object
```

[6]:
```python
# Splitting features & target variable
X = data.drop(columns=['No', 'Y house price of unit area'])
Y = data['Y house price of unit area']
```

[7]:
```python
# Step 4: Split the data
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2,␣
↪random_state=42)
```

[8]:
```python
# Step 5: Model training
model = LinearRegression()
model.fit(X_train, Y_train)
```

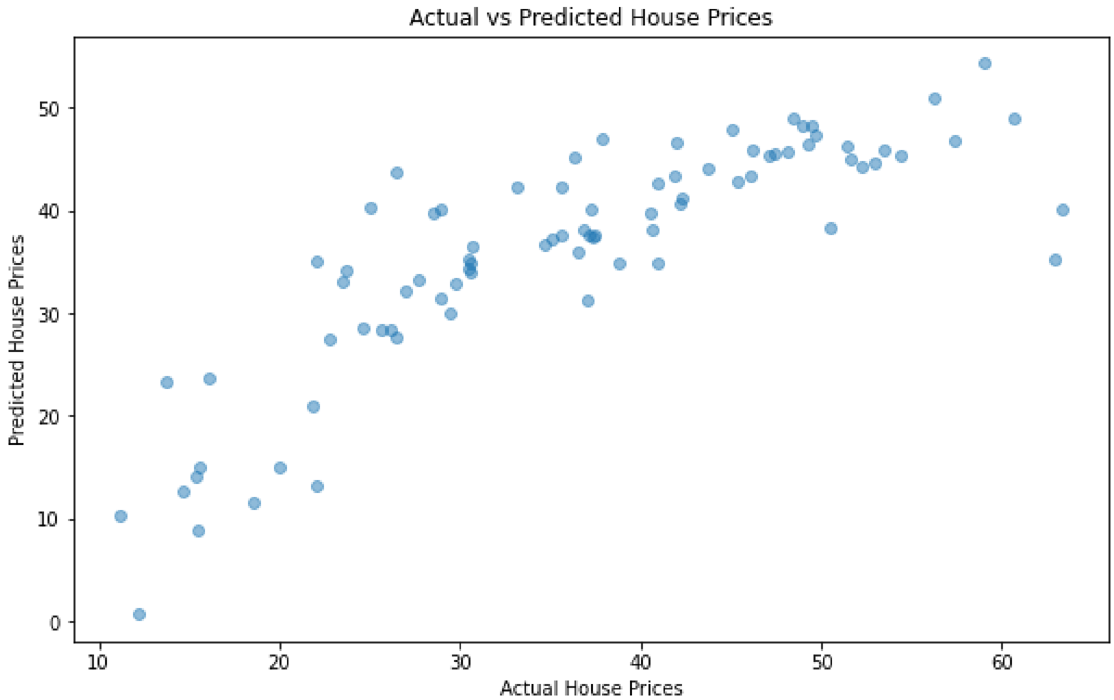
[8]:
```
LinearRegression()
```

[9]:
```python
# Step 6: Model evaluation
Y_pred = model.predict(X_test)

mse = mean_squared_error(Y_test, Y_pred)
mae = mean_absolute_error(Y_test, Y_pred)
rmse = np.sqrt(mse)

print("Mean Squared Error:", mse)
print("Mean Absolute Error:", mae)
print("Root Mean Squared Error:", rmse)
```

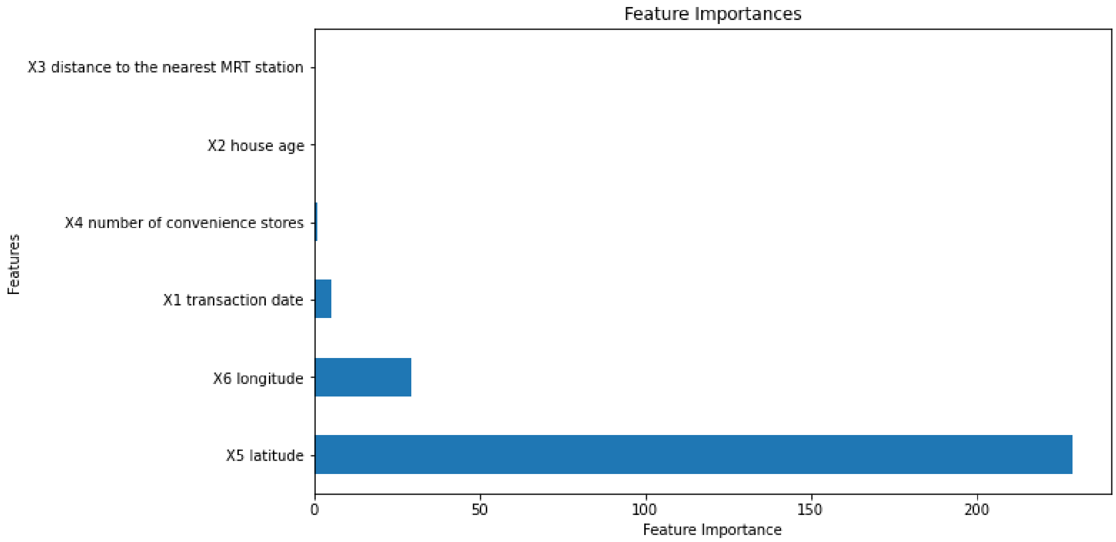
```
Mean Squared Error: 53.50561912450212
Mean Absolute Error: 5.3053556900741405
Root Mean Squared Error: 7.314753524521665
```

[10]:
```python
# Visualize predicted vs actual house prices
plt.figure(figsize=(10, 6))
plt.scatter(Y_test, Y_pred, alpha=0.5)
plt.xlabel("Actual House Prices")
```

```python
plt.ylabel("Predicted House Prices")
plt.title("Actual vs Predicted House Prices")
plt.show()
```



[11]:
```python
# Feature importances
importances = pd.Series(model.coef_, index=X.columns)
importances_sorted = importances.abs().sort_values(ascending=False)
plt.figure(figsize=(10, 6))
importances_sorted.plot(kind='barh')
plt.xlabel("Feature Importance")
plt.ylabel("Features")
plt.title("Feature Importances")
plt.show()
```



[12]:
```python
# Step 7: Interpretation
coefficients = pd.DataFrame({'feature': X.columns, 'coefficient': model.coef_})
print("\nCoefficients:")
print(coefficients)
```

```
Coefficients:
                                  feature  coefficient
0                     X1 transaction date     5.440742
1                            X2 house age    -0.270791
2  X3 distance to the nearest MRT station    -0.004759
3         X4 number of convenience stores     1.091425
4                             X5 latitude   229.043054
5                            X6 longitude   -29.492591
```

[ ]: