```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.datasets import fetch_california_housing
housing = fetch_california_housing()

data = pd.DataFrame(housing.data, columns=housing.feature_names)

data['PRICE'] = housing.target

print(data.isnull().sum())
```

```
MedInc        0
HouseAge      0
AveRooms      0
AveBedrms     0
Population    0
AveOccup      0
Latitude      0
Longitude     0
PRICE         0
dtype: int64
```

```python
x = data.drop(['PRICE'], axis=1)
y = data['PRICE']

from sklearn.model_selection import train_test_split
xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size=0.2,
random_state=0)

print(f"Training data shape (X): {xtrain.shape}")
print(f"Testing data shape (X): {xtest.shape}")
print(f"Training data shape (y): {ytrain.shape}")
print(f"Testing data shape (y): {ytest.shape}")
```

```
Training data shape (X): (16512, 8)
Testing data shape (X): (4128, 8)
Training data shape (y): (16512,)
Testing data shape (y): (4128,)
```

```python
from sklearn.linear_model import LinearRegression
lm = LinearRegression()

model = lm.fit(xtrain, ytrain)

ytrain_pred = lm.predict(xtrain)
ytest_pred = lm.predict(xtest)

train_evaluation = pd.DataFrame({'True Values': ytrain, 'Predicted Values':
ytrain_pred})
```

```
test_evaluation = pd.DataFrame({'True Values': ytest, 'Predicted Values':
ytest_pred})
```

```
from sklearn.metrics import mean_squared_error, r2_score
```

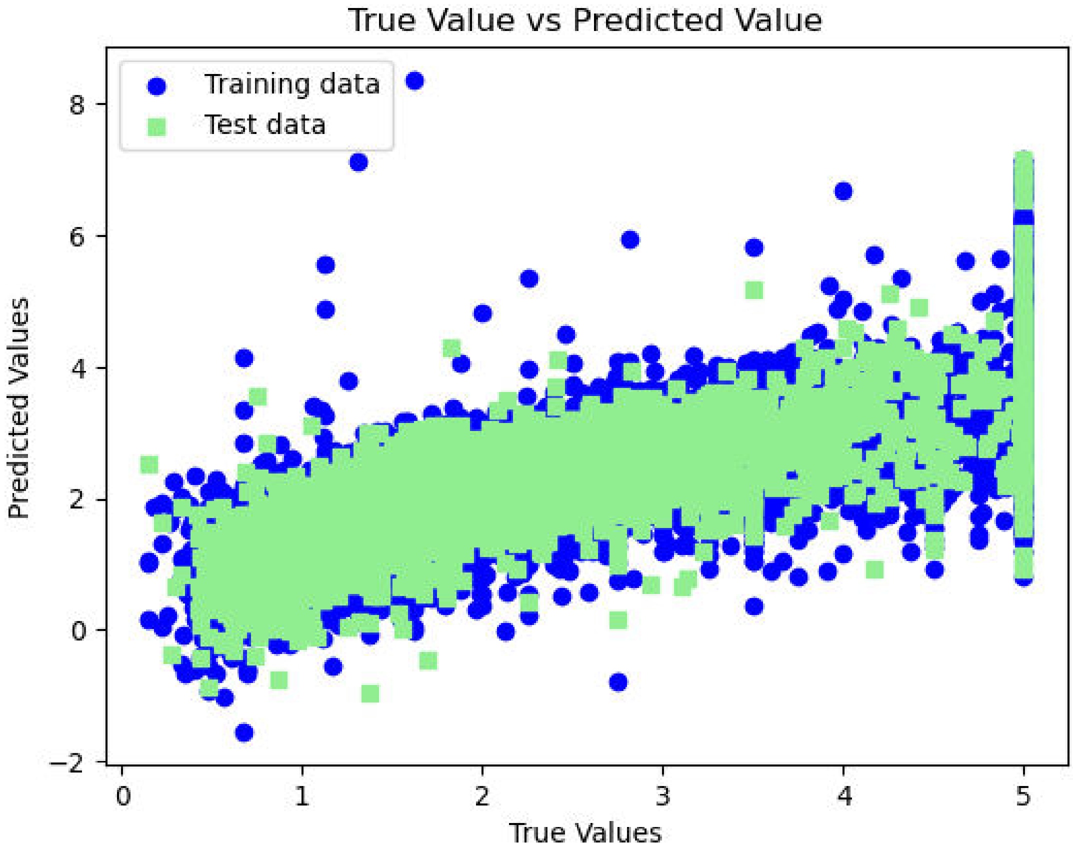
```
mse_train = mean_squared_error(ytrain, ytrain_pred)
print(f"MSE for Training Data: {mse_train}")
```

```
MSE for Training Data: 0.5234413607125449
```

```
mse_test = mean_squared_error(ytest, ytest_pred)
print(f"MSE for Testing Data: {mse_test}")
```

```
MSE for Testing Data: 0.5289841670367224
```

```
plt.scatter(ytrain, ytrain_pred, c='blue', marker='o', label='Training data')
plt.scatter(ytest, ytest_pred, c='lightgreen', marker='s', label='Test data')
plt.xlabel('True Values')
plt.ylabel('Predicted Values')
plt.title("True Value vs Predicted Value")
plt.legend(loc='upper left')
plt.show()
```


True Value vs Predicted Value
● Training data
🟩 Test data
Predicted Values
True Values