```
# importing the pandas library
import pandas as pd
```

```
# loading the dataset
df = pd.read_csv("/content/housing (2).csv")
```

```
# printing the top 5 rows
df.head()
```

| | RM | LSTAT | PTRATIO | MEDV |
|---|---|---|---|---|
| **0** | 6.575 | 4.98 | 15.3 | 504000.0 |
| **1** | 6.421 | 9.14 | 17.8 | 453600.0 |
| **2** | 7.185 | 4.03 | 17.8 | 728700.0 |
| **3** | 6.998 | 2.94 | 18.7 | 701400.0 |
| **4** | 7.147 | 5.33 | 18.7 | 760200.0 |

Next steps: View recommended plots New interactive sheet

```
# printing the last 5 rows
df.tail()
```

| | RM | LSTAT | PTRATIO | MEDV |
|---|---|---|---|---|
| **484** | 6.593 | 9.67 | 21.0 | 470400.0 |
| **485** | 6.120 | 9.08 | 21.0 | 432600.0 |
| **486** | 6.976 | 5.64 | 21.0 | 501900.0 |
| **487** | 6.794 | 6.48 | 21.0 | 462000.0 |
| **488** | 6.030 | 7.88 | 21.0 | 249900.0 |

```
# determining the column names and it's Dtype
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 489 entries, 0 to 488
Data columns (total 4 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   RM       489 non-null    float64
 1   LSTAT    489 non-null    float64
 2   PTRATIO  489 non-null    float64
 3   MEDV     489 non-null    float64
dtypes: float64(4)
memory usage: 15.4 KB
```

```
# describing the dataset
df.describe()
```

| | RM | LSTAT | PTRATIO | MEDV |
|---|---|---|---|---|
| **count** | 489.000000 | 489.000000 | 489.000000 | 4.890000e+02 |
| **mean** | 6.240288 | 12.939632 | 18.516564 | 4.543429e+05 |
| **std** | 0.643650 | 7.081990 | 2.111268 | 1.653403e+05 |
| **min** | 3.561000 | 1.980000 | 12.600000 | 1.050000e+05 |
| **25%** | 5.880000 | 7.370000 | 17.400000 | 3.507000e+05 |
| **50%** | 6.185000 | 11.690000 | 19.100000 | 4.389000e+05 |
| **75%** | 6.575000 | 17.120000 | 20.200000 | 5.187000e+05 |

```
# finding the null values
df.isna().sum() # there are no null values in the dataset
```

| | 0 |
|---|---|
| RM | 0 |
| LSTAT | 0 |
| PTRATIO | 0 |
| MEDV | 0 |

```
# To find the No.of rows & columns
df.shape
```

```
(489, 4)
```

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

```
X = df.drop('MEDV', axis=1) # droping the MEDV column
y = df['MEDV'] # here we treated it as target column
```

```
# splitting the data into training dataset and testing dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
Model = LinearRegression() # Building a model
Model.fit(X_train, y_train) # fitting data into it
```

```
LinearRegression
```

```
y_pred = Model.predict(X_test) # make prediction
```

```
# calculating the mean squared error
MSE = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error is: {MSE}")
```

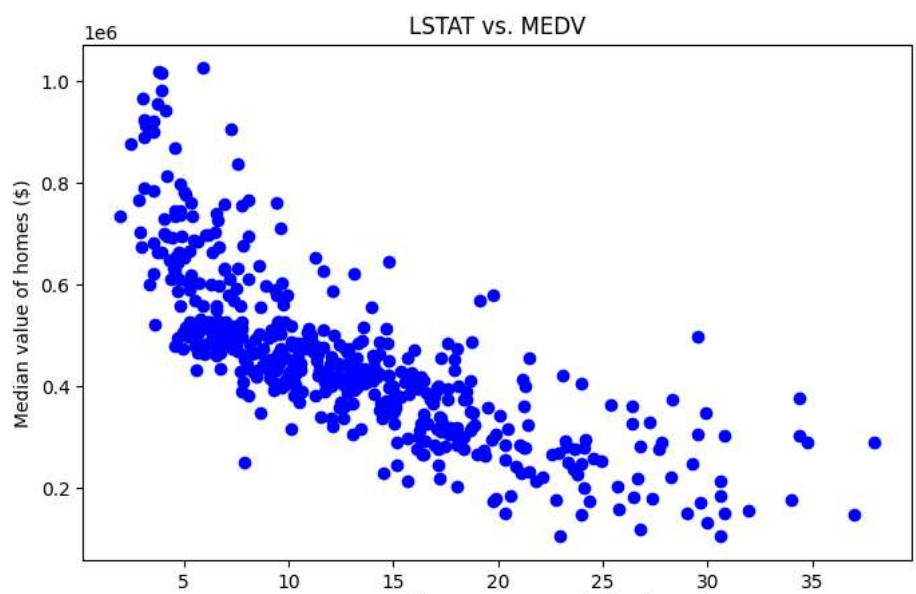
```
Mean Squared Error is: 6789025559.265892
```

```
# calculating the r2_score
r2 = r2_score(y_test, y_pred)
print(f"r2 Score is: {r2}")
```

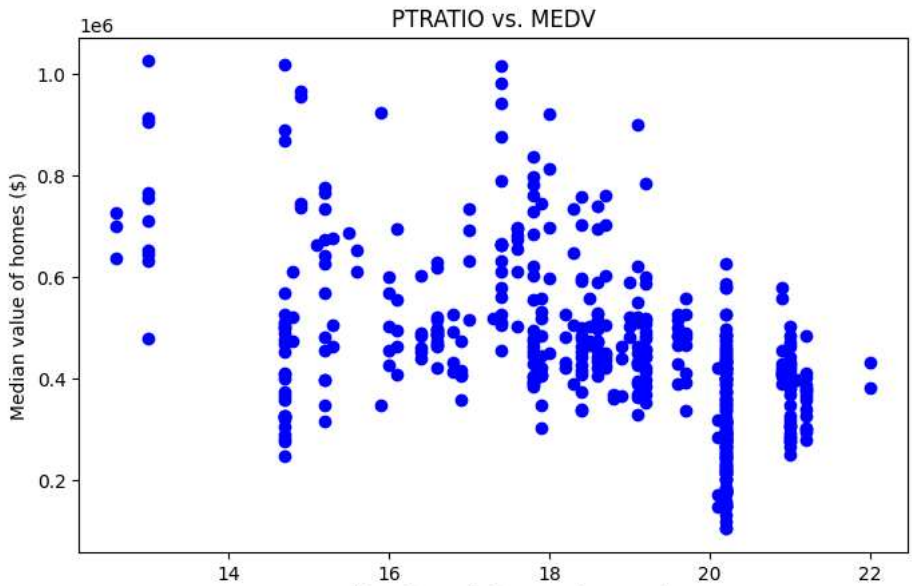
```
r2 Score is: 0.691093400309851
```

```
# Visualization
import matplotlib.pyplot as plt
plt.figure(figsize=(8,5))
plt.scatter(y_test, y_pred, color='red')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], '--',lw=2)
plt.xlabel("Actual Prices")
plt.ylabel("Predicted Prices")
plt.title("Actual vs Predicted Housing Price")
plt.show()
```

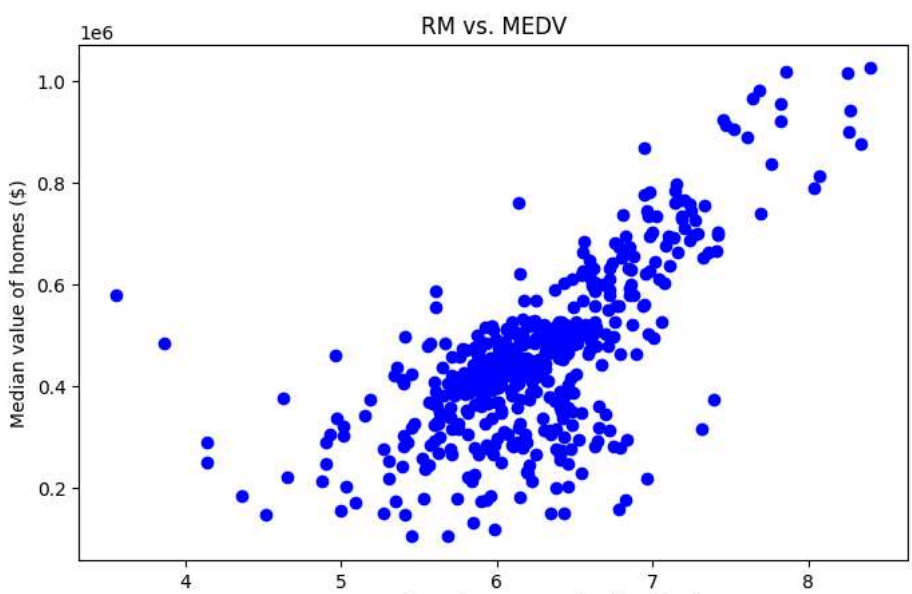

```
plt.figure(figsize=(8, 5))
plt.scatter(df['LSTAT'], df['MEDV'], color='blue')
plt.xlabel('Percentage of lower status population (LSTAT)')
plt.ylabel('Median value of homes ($)')
plt.title('LSTAT vs. MEDV')
plt.show()
```




```
plt.figure(figsize=(8,5))
plt.scatter(df['PTRATIO'], df['MEDV'], color='blue')
plt.xlabel('Pupil-teacher ratio by town (PTRATIO)')
plt.ylabel('Median value of homes ($)')
plt.title('PTRATIO vs. MEDV')
plt.show()
```




```
plt.figure(figsize=(8, 5))
plt.scatter(df['RM'], df['MEDV'], color='blue')
plt.xlabel('Average number of rooms per dwelling (RM)')
plt.ylabel('Median value of homes ($)')
plt.title('RM vs. MEDV')
plt.show()
```




```
import seaborn as sns
# Compute the correlation matrix
correlation_matrix = df.corr().round(2)

# Display the correlation matrix
print(correlation_matrix)

# Visualize the correlation matrix
plt.figure(figsize=(6, 4))
sns.heatmap(data=correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```

```
           RM  LSTAT  PTRATIO  MEDV
RM       1.00  -0.61    -0.30  0.70
LSTAT   -0.61   1.00     0.36 -0.76
PTRATIO -0.30   0.36     1.00 -0.52
MEDV     0.70  -0.76    -0.52  1.00
```