In [7]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [19]:
```
df = pd.read_csv('HousingData.csv')
```

In [20]:
```
df.head()
```

Out[20]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTA
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.9
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.1
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.0
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.9
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	Na

In [21]:
```
df
```

Out[21]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LS
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	N
...	...	...	...	...	...	...	...	...	...	...	...	...	
501	0.06263	0.0	11.93	0.0	0.573	6.593	69.1	2.4786	1	273	21.0	391.99	N
502	0.04527	0.0	11.93	0.0	0.573	6.120	76.7	2.2875	1	273	21.0	396.90	9
503	0.06076	0.0	11.93	0.0	0.573	6.976	91.0	2.1675	1	273	21.0	396.90	5
504	0.10959	0.0	11.93	0.0	0.573	6.794	89.3	2.3889	1	273	21.0	393.45	6
505	0.04741	0.0	11.93	0.0	0.573	6.030	NaN	2.5050	1	273	21.0	396.90	7

506 rows × 14 columns

In [22]:
```
df.shape
```

Out[22]:
```
(506, 14)
```

In [23]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 14 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   CRIM     486 non-null    float64
 1   ZN       486 non-null    float64
 2   INDUS    486 non-null    float64
 3   CHAS     486 non-null    float64
 4   NOX      506 non-null    float64
 5   RM       506 non-null    float64
 6   AGE      486 non-null    float64
 7   DIS      506 non-null    float64
 8   RAD      506 non-null    int64  
 9   TAX      506 non-null    int64  
 10  PTRATIO  506 non-null    float64
 11  B        506 non-null    float64
 12  LSTAT    486 non-null    float64
 13  MEDV     506 non-null    float64
dtypes: float64(12), int64(2)
memory usage: 55.5 KB
```

In [46]:

```
df.isna().sum()

# filling null walues with mean
crim_mean = df['CRIM'].mean()
df['CRIM'].fillna(value=crim_mean, inplace=True)

zn_mean = df['ZN'].mean()
df['ZN'].fillna(value=zn_mean, inplace=True)

INDUS_mean = df['INDUS'].mean()
df['INDUS'].fillna(value=INDUS_mean, inplace=True)

CHAS_mean = df['CHAS'].mean()
df['CHAS'].fillna(value=CHAS_mean, inplace=True)

AGE_mean = df['AGE'].mean()
df['AGE'].fillna(value=AGE_mean, inplace=True)

LSTAT_mean = df['LSTAT'].mean()
df['LSTAT'].fillna(value=LSTAT_mean, inplace=True)
```

In [47]:

```
target = 'MEDV'

y = df[target]
x = df.drop(target, axis=1)
```

In [48]:

```
x.head()
```

Out[48]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LS
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	12.71

In [49]:

```
y.head()
```

Out[49]:

```
0    24.0
1    21.6
2    34.7
3    33.4
4    36.2
Name: MEDV, dtype: float64
```

In [50]:

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size = 0.2, rando
```

In [51]:

```
from sklearn.linear_model import LinearRegression
regression = LinearRegression()

# x should not contain any nan values
regression.fit(x_train, y_train)
```

Out[51]:

```
▸ LinearRegression
```

In [54]:

```
# train score
train_score = round(regression.score(x_train, y_train)*100, 2)
train_score
```

Out[54]:

```
76.62
```

In [55]:

```
# checking the model with the test data
y_predict = regression.predict(x_test)
```

In [59]:

```
from sklearn.metrics import r2_score
score = round(r2_score(y_test, y_predict)*100, 2)
score
```

Out[59]:

```
55.29
```

In [61]:
```
round(regression.score(x_test, y_test)*100, 2)
```

Out[61]:
```
55.29
```

In [63]:
```
y_predict
```

Out[63]:
```
array([23.11597021, 19.37743801, 20.16436764, 19.37004239,  4.8607811 ,
       11.58990176, 21.04280622, 28.72326213, 29.08248133, 13.9256125 ,
        6.27033707, 32.7102621 , 18.87768655, 20.27514856, 37.1838814 ,
       22.36400344, 28.42558146, 32.47408175, 11.07643106, 24.38232183,
       20.91779414, 27.78724167, 37.75899506, 14.01207366,  9.43910388,
       15.08963152, 35.75052876, 26.20011258, 25.59461576, 27.19087921,
       18.81106287, 30.69809414, 31.64105636, 16.36474586, 39.72759778,
       20.59506095, 19.07693199, 17.35393174, 21.75620256, 28.54306016,
       27.23664354, 23.01532111, 14.67235626, 26.08898883, 18.18665615,
       14.21627193, 24.97565816, 19.00608819, 20.99025076,  5.85262294,
       27.48964564, 24.62939499, 11.84130594, 40.20093182, 14.71760042,
       22.03128072, 19.93587786, 20.05053166, 23.5471769 , 22.02882179,
       20.95387649, 35.4139075 , 17.58713046, 21.26236695, 23.5151731 ,
       43.33528881, 19.33849847, 19.93631795, 22.58128466, 28.15473685,
       25.80196033, 16.17976307, 13.92439355, 33.44638968, 25.56710091,
       21.98619096, 12.56019147, 17.01818314, 28.72067153, 18.16611114,
       24.4556493 , 27.89738075, 22.56329715, 24.10551232, 27.42792176,
       30.32425741, 24.50960781, 19.7404164 , 31.38592746, 21.7740549 ,
       19.47073823, 33.84481499, 37.89952554, 24.28237526, 24.87310103,
       12.21321178, 28.53705285,  9.51602417, 13.48524373, 28.61072682,
       20.55623896, 15.69060344])
```

In [64]:
```
y_predict = pd.DataFrame(y_predict,columns=['target'])
```

In [65]:
```
y_predict
```

Out[65]:

	target
0	23.115970
1	19.377438
2	20.164368
3	19.370042
4	4.860781
...	...
97	9.516024
98	13.485244
99	28.610727
100	20.556239
101	15.690603

102 rows × 1 columns

```
In [78]:
```

```
In [ ]:
```

```
In [ ]:
```