

```
In [1]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

In [2]: df=pd.read_csv("Boston.csv")
df.head()

Out[2]:
```

	Unnamed: 0	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
0	1	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0
1	2	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6
2	3	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
3	4	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
4	5	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.2

```


In [3]: df.describe()

Out[3]:
```

	Unnamed: 0	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
count	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000
mean	253.500000	3.613524	11.363636	11.136779	0.069170	0.554695	6.284634	68.574901	3.795043	9.549407	408.237154	18.455534	356.674032	12.653063	22.532806
std	146.213884	8.601545	23.322453	6.860353	0.253994	0.115878	0.702617	28.148861	2.105710	8.707259	168.537116	2.164946	91.294864	7.141062	9.197104
min	1.000000	0.006320	0.000000	0.460000	0.000000	0.385000	3.561000	2.900000	1.129600	1.000000	187.000000	12.600000	0.320000	1.730000	5.000000
25%	127.250000	0.082045	0.000000	5.190000	0.000000	0.449000	5.885500	45.025000	2.100175	4.000000	279.000000	17.400000	375.377500	6.950000	17.025000
50%	253.500000	0.256510	0.000000	9.690000	0.000000	0.538000	6.208500	77.500000	3.207450	5.000000	330.000000	19.050000	391.440000	11.360000	21.200000
75%	379.750000	3.677083	12.500000	18.100000	0.000000	0.624000	6.623500	94.075000	5.188425	24.000000	666.000000	20.200000	396.225000	16.955000	25.000000
max	506.000000	88.976200	100.000000	27.740000	1.000000	0.871000	8.780000	100.000000	12.126500	24.000000	711.000000	22.000000	396.900000	37.970000	50.000000

```


In [4]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 15 columns):
#   Column      Non-Null Count  Dtype
---  -
0   Unnamed: 0   506 non-null    int64
1   crim        506 non-null    float64
2   zn          506 non-null    float64
3   indus       506 non-null    float64
4   chas        506 non-null    int64
5   nox         506 non-null    float64
6   rm          506 non-null    float64
7   age         506 non-null    float64
8   dis         506 non-null    float64
9   rad         506 non-null    int64
10  tax         506 non-null    int64
11  ptratio     506 non-null    float64
12  black       506 non-null    float64
13  lstat       506 non-null    float64
14  medv        506 non-null    float64
dtypes: float64(11), int64(4)
memory usage: 59.4 KB

In [5]: df.isna().sum()

Out[5]:
```

Unnamed: 0	0
crim	0
zn	0
indus	0
chas	0
nox	0
rm	0
age	0
dis	0
rad	0
tax	0
ptratio	0
black	0
lstat	0
medv	0

```
dtype: int64

In [6]: df.isnull().sum()

Out[6]:
```

Unnamed: 0	0
crim	0
zn	0
indus	0
chas	0
nox	0
rm	0
age	0
dis	0
rad	0
tax	0
ptratio	0
black	0
lstat	0
medv	0

```
dtype: int64

In [7]: df.shape

Out[7]: (506, 15)

In [11]: df1=df.drop(columns=["Unnamed: 0"])
df1

Out[11]:
```

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
501	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	273	21.0	391.99	9.67	22.4
502	0.04527	0.0	11.93	0	0.573	6.120	76.7	2.2875	1	273	21.0	396.90	9.08	20.6
503	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273	21.0	396.90	5.64	23.9
504	0.10959	0.0	11.93	0	0.573	6.794	89.3	2.3889	1	273	21.0	393.45	6.48	22.0
505	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273	21.0	396.90	7.88	11.9

506 rows x 14 columns

```


In [13]: x=df.drop ("medv",axis=True)
y=df["medv"]

In [14]: x,y

Out[14]:
```

	Unnamed: 0	crim	zn	indus	chas	nox	rm	age	dis	rad	\
0	1	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	
1	2	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	
2	3	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	
3	4	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	
4	5	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	
...	...	...	...	...	...	...	...	...	...	...	
501	502	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	
502	503	0.04527	0.0	11.93	0	0.573	6.120	76.7	2.2875	1	
503	504	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	
504	505	0.10959	0.0	11.93	0	0.573	6.794	89.3	2.3889	1	
505	506	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	

  

	tax	ptratio	black	lstat
0	296	15.3	396.90	4.98
1	242	17.8	396.90	9.14
2	242	17.8	392.83	4.03
3	222	18.7	394.63	2.94
4	222	18.7	396.90	5.33
...	...	...	...	...
501	273	21.0	391.99	9.67
502	273	21.0	396.90	9.08
503	273	21.0	396.90	5.64
504	273	21.0	393.45	6.48
505	273	21.0	396.90	7.88

```


[506 rows x 14 columns],
0      24.0
1      21.6
2      34.7
3      33.4
4      36.2
...
501     22.4
502     20.6
503     23.9
504     22.0
505     11.9
Name: medv, Length: 506, dtype: float64)

In [21]: from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split

In [22]: x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)

In [23]: x_train.shape

Out[23]: (404, 14)

In [24]: model=LinearRegression()
model.fit(x_train,y_train)

Out[24]:
```

LinearRegression

Parameters

```


In [27]: model.intercept_

Out[27]: np.float64(30.412686521180227)

In [28]: y_pred=model.predict(x_test)

In [31]: print(y_pred[:50])

[28.97335364 35.87795794 13.97366183 25.61583978 18.66608153 23.64466022
17.05971239 14.5264485 22.79402759 19.60618783 24.62043501 19.06428675
-6.27197735 22.20788474 18.64849181 26.15022718 20.62835191 5.77374284
40.72117979 17.48530737 27.08436097 29.90818637 11.65330429 22.99879937
17.85048447 15.5556251 23.31650588 14.44013189 22.11106256 18.20109763
22.33391113 24.79774986 26.15304018 17.89425941 16.52927207 17.10592481
31.77372627 19.58243073 23.603617 25.2202457 12.89702838 31.91216634
```

