In [1]:
```
import pandas as pd
import numpy as np
import sklearn as sk
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
```

In [2]:
```
df=pd.read_csv("C:\\Users\\asus\\Desktop\\Boston.csv")
df
```

Out[2]:

	Unnamed: 0	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	bl
0	1	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396
1	2	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396
2	3	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392
3	4	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394
4	5	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396
...	...	...	...	...	...	...	...	...	...	...	...	...	
501	502	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	273	21.0	391
502	503	0.04527	0.0	11.93	0	0.573	6.120	76.7	2.2875	1	273	21.0	396
503	504	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273	21.0	396
504	505	0.10959	0.0	11.93	0	0.573	6.794	89.3	2.3889	1	273	21.0	393
505	506	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273	21.0	396

506 rows × 15 columns

Data Exploration

In [3]:
```
df.shape
```

Out[3]: (506, 15)

In [4]:
```
df.describe()
```

Out[4]:

	Unnamed: 0	crim	zn	indus	chas	nox	rm	
count	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	5
mean	253.500000	3.613524	11.363636	11.136779	0.069170	0.554695	6.284634	
std	146.213884	8.601545	23.322453	6.860353	0.253994	0.115878	0.702617	
min	1.000000	0.006320	0.000000	0.460000	0.000000	0.385000	3.561000	
25%	127.250000	0.082045	0.000000	5.190000	0.000000	0.449000	5.885500	
50%	253.500000	0.256510	0.000000	9.690000	0.000000	0.538000	6.208500	
75%	379.750000	3.677083	12.500000	18.100000	0.000000	0.624000	6.623500	
max	506.000000	88.976200	100.000000	27.740000	1.000000	0.871000	8.780000	1

In [5]:
```
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 [6]:
```
df.isnull().sum().sum()
```

Out[6]: 0

In [7]:
```
df['chas'].value_counts()
```

Out[7]:
```
0    471
1     35
Name: chas, dtype: int64
```

In [8]:
```
df['zn'].value_counts()
```

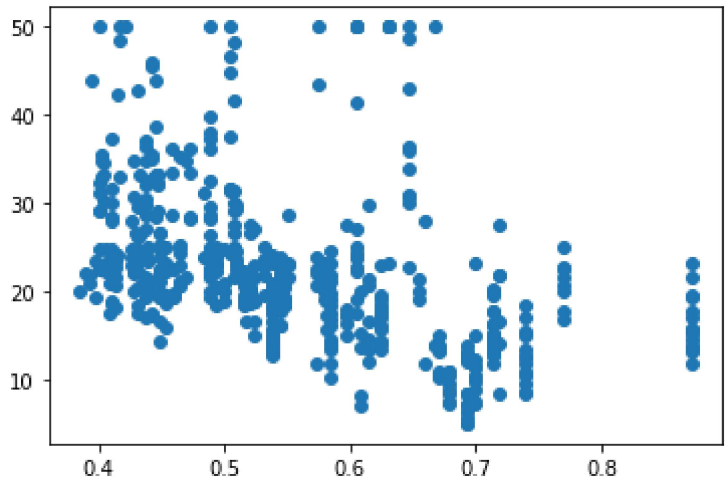
Out[8]:
```
0.0      372
20.0      21
80.0      15
22.0      10
12.5      10
25.0      10
40.0       7
45.0       6
30.0       6
90.0       5
95.0       4
60.0       4
21.0       4
33.0       4
55.0       3
70.0       3
34.0       3
52.5       3
35.0       3
28.0       3
75.0       3
82.5       2
85.0       2
17.5       1
100.0      1
18.0       1
Name: zn, dtype: int64
```

In [9]:
```
df.dtypes
```

Out[9]:
```
Unnamed: 0      int64
crim          float64
zn            float64
indus         float64
chas            int64
nox           float64
rm            float64
age           float64
dis           float64
rad             int64
tax             int64
ptratio       float64
black         float64
lstat         float64
medv          float64
dtype: object
```

In [10]:
```
df.corr()
plt.figure(figsize=(15,8))
sns.heatmap(df.corr(),annot=True)
plt.show()
```

In [11]:
```
y3=df['medv']
for i in df.columns:
 x3=df[i]
 plt.scatter(x3,y3)
 plt.show()
```



Data Preprocessing

In [12]:
```
x_train=df.drop(['medv'],axis=1)
y_train=df['medv']
x_train
```

Out[12]:

	Unnamed: 0	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	bla
0	1	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396
1	2	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396
2	3	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392
3	4	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394
4	5	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396
...	...	...	...	...	...	...	...	...	...	...	...	...	
501	502	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	273	21.0	391
502	503	0.04527	0.0	11.93	0	0.573	6.120	76.7	2.2875	1	273	21.0	396
503	504	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273	21.0	396
504	505	0.10959	0.0	11.93	0	0.573	6.794	89.3	2.3889	1	273	21.0	393
505	506	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273	21.0	396

506 rows × 14 columns

Data Modeling

In [13]:
```
from sklearn.linear_model import LinearRegression
```

In [14]:
```
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[14]:
```
LinearRegression()
```

In [15]:
```
df2=pd.read_csv("C:\\Users\\asus\\Desktop\\Boston_Test (1).csv")
df2.head()
```

Out[15]:

	Unnamed: 0	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	blac
0	351	0.07950	60.0	1.69	0	0.411	6.579	35.9	10.7103	4	411	18.3	370.7
1	352	0.07244	60.0	1.69	0	0.411	5.884	18.5	10.7103	4	411	18.3	392.3
2	353	0.01709	90.0	2.02	0	0.410	6.728	36.1	12.1265	5	187	17.0	384.4
3	354	0.04301	80.0	1.91	0	0.413	5.663	21.9	10.5857	4	334	22.0	382.8
4	355	0.10659	80.0	1.91	0	0.413	5.936	19.5	10.5857	4	334	22.0	376.0

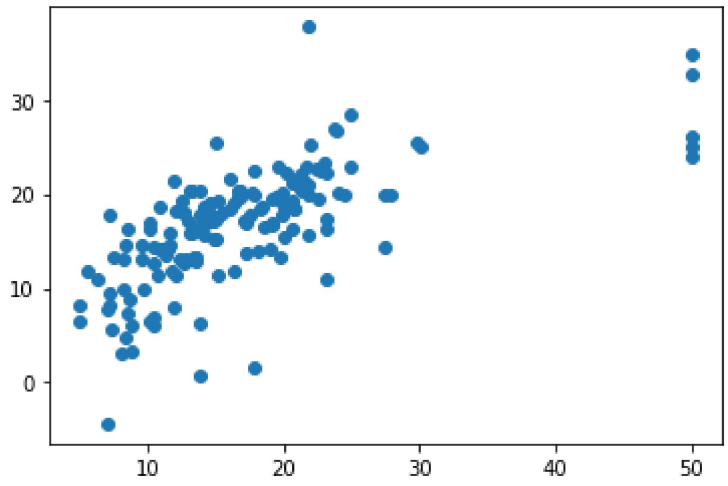
In [16]:
```
x_test=df2.drop(['medv'],axis=1)
y_test=df2['medv']
x_test
```

Out[16]:

	Unnamed: 0	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	b
0	351	0.07950	60.0	1.69	0	0.411	6.579	35.9	10.7103	4	411	18.3	37
1	352	0.07244	60.0	1.69	0	0.411	5.884	18.5	10.7103	4	411	18.3	39
2	353	0.01709	90.0	2.02	0	0.410	6.728	36.1	12.1265	5	187	17.0	38
3	354	0.04301	80.0	1.91	0	0.413	5.663	21.9	10.5857	4	334	22.0	38
4	355	0.10659	80.0	1.91	0	0.413	5.936	19.5	10.5857	4	334	22.0	37
...	...	...	...	...	...	...	...	...	...	...	...	...	
150	501	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	273	21.0	39
151	502	0.04527	0.0	11.93	0	0.573	6.120	76.7	2.2875	1	273	21.0	39
152	503	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273	21.0	39
153	504	0.10959	0.0	11.93	0	0.573	6.794	89.3	2.3889	1	273	21.0	39
154	505	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273	21.0	39

155 rows × 14 columns

In [17]:
```
y_pred=lr.predict(x_test)
plt.scatter(y_test,y_pred)
plt.show()
```



In [18]:
```
from sklearn.metrics import mean_squared_error,mean_absolute_error,r2_score
mse=mean_squared_error(y_test,y_pred)
ae=mean_absolute_error(y_test,y_pred)
re=r2_score(y_test,y_pred)
print(mse)
print(ae)
print(re)
```

```
37.13832013971392
4.1529745209205755
0.44226910905239236
```

Score By Using Linear Regression

In [19]:
```
lr.score(x_test,y_test)
```

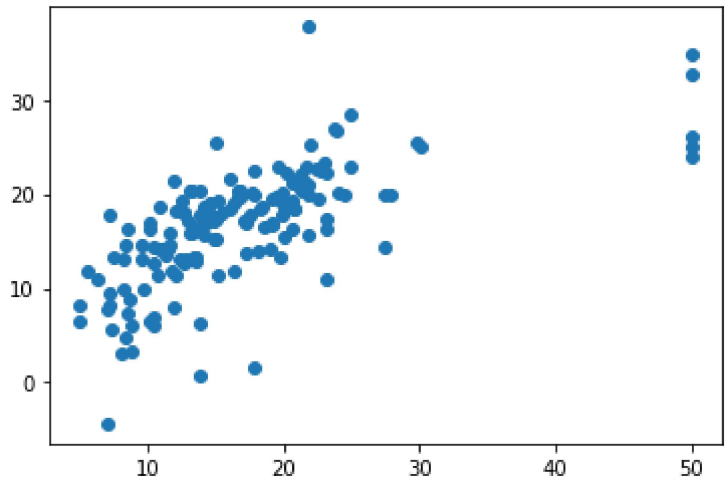
Out[19]: 0.44226910905239236

Random Forest

In [24]:
```
from sklearn.ensemble import RandomForestRegressor
regressor = RandomForestRegressor(n_estimators = 100, random_state = 0)
regressor.fit(x_train, y_train)
```

Out[24]:
```
RandomForestRegressor(random_state=0)
```

In [21]:
```
y_pred=lr.predict(x_test)
plt.scatter(y_test,y_pred)
plt.show()
```



In [22]:
```
regressor.score(x_test,y_test)
```

Out[22]:
```
0.9574974486719926
```

In [ ]:
```

```