In [13]:

```python
# importing required lib

import pandas as pd                  # to work on data sets
import numpy as np                   # for scientific and math calculation
import seaborn as sns                # to visualize data using matplot
import matplotlib.pyplot as plt      # make visuals of data such as graph, bar graph, scatter plot etc...

%matplotlib inline
```

In [14]:

```python
# reading the data for training the model
HouseDF = pd.read_csv("boston-housing/train.csv")
```

In [15]:

```python
# top 5 element/row of the training data
HouseDF.head()
```

Out[15]:

	ID	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	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
3	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
4	7	0.08829	12.5	7.87	0	0.524	6.012	66.6	5.5605	5	311	15.2	395.60	12.43	22.9

In [16]:

```python
# information about the colums of the data set
HouseDF.info()
# we did not found any null values so we can proceed withoud removing null values
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 333 entries, 0 to 332
Data columns (total 15 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   ID       333 non-null    int64  
 1   crim     333 non-null    float64
 2   zn       333 non-null    float64
 3   indus    333 non-null    float64
 4   chas     333 non-null    int64  
 5   nox      333 non-null    float64
 6   rm       333 non-null    float64
 7   age      333 non-null    float64
 8   dis      333 non-null    float64
 9   rad      333 non-null    int64  
 10  tax      333 non-null    int64  
 11  ptratio  333 non-null    float64
 12  black    333 non-null    float64
 13  lstat    333 non-null    float64
 14  medv     333 non-null    float64
dtypes: float64(11), int64(4)
memory usage: 39.2 KB
```

In [7]:

```python
# describe data set and values like mean, deviation, min, max etc...
HouseDF.describe()
```

Out[7]:

	ID	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
count	333.000000	333.000000	333.000000	333.000000	333.000000	333.000000	333.000000	333.000000	333.000000	333.000000	333.000000	333.000000	333.000000	333.000000	333.000000
mean	250.951952	3.360341	10.689189	11.293483	0.060060	0.557144	6.265619	68.226426	3.709934	9.633634	409.279279	18.448048	359.466096	12.515435	22.768769
std	147.859438	7.352272	22.674762	6.998123	0.237956	0.114955	0.703952	28.133344	1.981123	8.742174	170.841988	2.151821	86.584567	7.067781	9.173468
min	1.000000	0.006320	0.000000	0.740000	0.000000	0.385000	3.561000	6.000000	1.129600	1.000000	188.000000	12.600000	3.500000	1.730000	5.000000
25%	123.000000	0.078960	0.000000	5.130000	0.000000	0.453000	5.884000	45.400000	2.122400	4.000000	279.000000	17.400000	376.730000	7.180000	17.400000
50%	244.000000	0.261690	0.000000	9.900000	0.000000	0.538000	6.202000	76.700000	3.092300	5.000000	330.000000	19.000000	392.050000	10.970000	21.600000
75%	377.000000	3.678220	12.500000	18.100000	0.000000	0.631000	6.595000	93.800000	5.116700	24.000000	666.000000	20.200000	396.240000	16.420000	25.000000
max	506.000000	73.534100	100.000000	27.740000	1.000000	0.871000	8.725000	100.000000	10.710300	24.000000	711.000000	21.200000	396.900000	37.970000	50.000000

In [19]:

```python
HouseDF.columns
```

Out[19]:

```
Index(['ID', 'crim', 'zn', 'indus', 'chas', 'nox', 'rm', 'age', 'dis', 'rad',
       'tax', 'ptratio', 'black', 'lstat', 'medv'],
      dtype='object')
```

In [21]:

```python
sns.pairplot(HouseDF)
```

Out[21]:

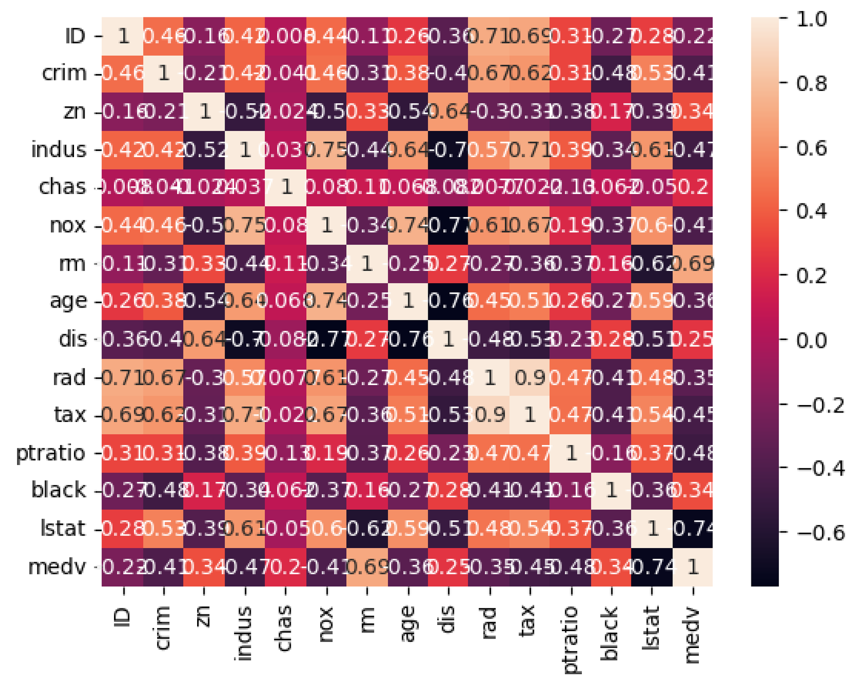
```
<seaborn.axisgrid.PairGrid at 0x228fe421a00>
```

ID
crim
zn
indus
chas
nox


rm
age
dis
rad
tax
ptratio

In [22]:
```python
sns.heatmap(HouseDF.corr(), annot=True)
```

Out[22]: <Axes: >



In [23]:
```python
X = HouseDF[['crim', 'zn', 'indus', 'chas', 'nox', 'rm', 'age', 'dis', 'rad',
       'tax', 'ptratio', 'black', 'lstat']]

y = HouseDF['medv']
```

In [24]:
```python
# importing moduls to split the data to train and test
from sklearn.model_selection import train_test_split
```

In [25]:
```python
# splitting data into training and testing data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.40, random_state=101)
```

In [27]:
```python
#importing linear regression model to train our model
from sklearn.linear_model import LinearRegression
```

In [28]:
```python
lm = LinearRegression()
```

In [30]:
```python
# fitting the model
lm.fit(X_train, y_train)
```

Out[30]:
```
▼ LinearRegression
LinearRegression()
```

In [31]:
```python
coeff_DF = pd.DataFrame(lm.coef_, X.columns, columns=['Coefficient'])
```

In [32]:
```python
coeff_DF
```

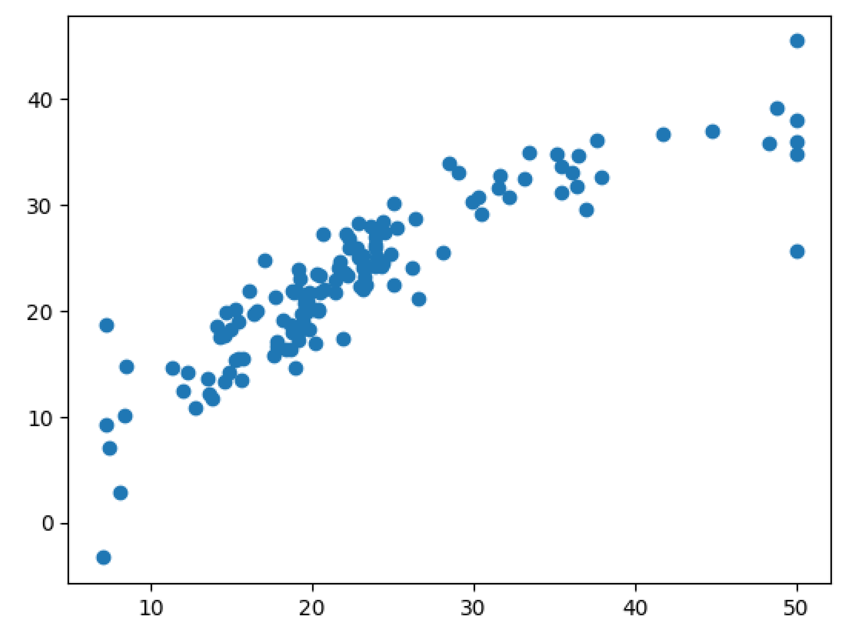
Out[32]:

	Coefficient
crim	-0.097415
zn	0.056597
indus	0.094682
chas	4.634273
nox	-13.323376
rm	3.230758
age	-0.013663
dis	-1.509314
rad	0.397022
tax	-0.017004
ptratio	-0.804268
black	0.011120
lstat	-0.566284

In [36]:
```python
# predicting the data
predictions =  lm.predict(X_test)
```

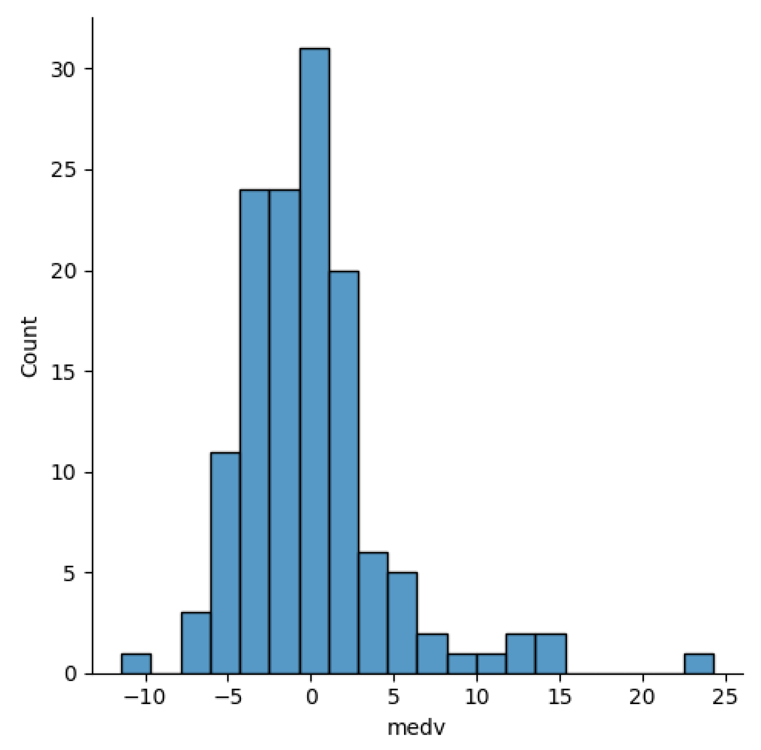
In [37]:
```python
plt.scatter(y_test, predictions)
```

Out[37]: <matplotlib.collections.PathCollection at 0x22893891ee0>



In [46]:
```python
sns.displot((y_test-predictions), bins=20)
```

Out[46]: <seaborn.axisgrid.FacetGrid at 0x228970d4dd0>



In [ ]: