In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
df1=pd.read_csv(r'C:\Users\user\Downloads\8_BreastCancerPrediction.csv')
df1
```

Out[2]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothne
0	842302	M	17.99	10.38	122.80	1001.0	
1	842517	M	20.57	17.77	132.90	1326.0	
2	84300903	M	19.69	21.25	130.00	1203.0	
3	84348301	M	11.42	20.38	77.58	386.1	
4	84358402	M	20.29	14.34	135.10	1297.0	
...	...	...	...	...	...	...	
564	926424	M	21.56	22.39	142.00	1479.0	
565	926682	M	20.13	28.25	131.20	1261.0	
566	926954	M	16.60	28.08	108.30	858.1	
567	927241	M	20.60	29.33	140.10	1265.0	
568	92751	B	7.76	24.54	47.92	181.0	

569 rows × 33 columns

In [3]:

```
df=df1.head(50)
df
```

Out[3]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothnes
0	842302	M	17.990	10.38	122.80	1001.0	
1	842517	M	20.570	17.77	132.90	1326.0	
2	84300903	M	19.690	21.25	130.00	1203.0	
3	84348301	M	11.420	20.38	77.58	386.1	
4	84358402	M	20.290	14.34	135.10	1297.0	
5	843786	M	12.450	15.70	82.57	477.1	
6	844359	M	18.250	19.98	119.60	1040.0	
7	84458202	M	13.710	20.83	90.20	577.9	
8	844981	M	13.000	21.82	87.50	519.8	
9	84501001	M	12.460	24.04	83.97	475.9	
10	845636	M	16.020	23.24	102.70	797.8	
11	84610002	M	15.780	17.89	103.60	781.0	
12	846226	M	19.170	24.80	132.40	1123.0	
13	846381	M	15.850	23.95	103.70	782.7	
14	84667401	M	13.730	22.61	93.60	578.3	
15	84799002	M	14.540	27.54	96.73	658.8	
16	848406	M	14.680	20.13	94.74	684.5	
17	84862001	M	16.130	20.68	108.10	798.8	
18	849014	M	19.810	22.15	130.00	1260.0	
19	8510426	B	13.540	14.36	87.46	566.3	
20	8510653	B	13.080	15.71	85.63	520.0	
21	8510824	B	9.504	12.44	60.34	273.9	
22	8511133	M	15.340	14.26	102.50	704.4	
23	851509	M	21.160	23.04	137.20	1404.0	
24	852552	M	16.650	21.38	110.00	904.6	
25	852631	M	17.140	16.40	116.00	912.7	
26	852763	M	14.580	21.53	97.41	644.8	
27	852781	M	18.610	20.25	122.10	1094.0	
28	852973	M	15.300	25.27	102.40	732.4	
29	853201	M	17.570	15.05	115.00	955.1	
30	853401	M	18.630	25.11	124.80	1088.0	
31	853612	M	11.840	18.70	77.93	440.6	
32	85382601	M	17.020	23.98	112.80	899.3	
33	854002	M	19.270	26.47	127.90	1162.0	
34	854039	M	16.130	17.88	107.00	807.2	
35	854253	M	16.740	21.59	110.10	869.5	

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothnes
36	854268	M	14.250	21.72	93.63	633.0	
37	854941	B	13.030	18.42	82.61	523.8	
38	855133	M	14.990	25.20	95.54	698.8	
39	855138	M	13.480	20.82	88.40	559.2	
40	855167	M	13.440	21.58	86.18	563.0	
41	855563	M	10.950	21.35	71.90	371.1	
42	855625	M	19.070	24.81	128.30	1104.0	
43	856106	M	13.280	20.28	87.32	545.2	
44	85638502	M	13.170	21.81	85.42	531.5	
45	857010	M	18.650	17.60	123.70	1076.0	
46	85713702	B	8.196	16.84	51.71	201.9	
47	85715	M	13.170	18.66	85.98	534.6	
48	857155	B	12.050	14.63	78.04	449.3	
49	857156	B	13.490	22.30	86.91	561.0	

50 rows × 33 columns

In [4]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50 entries, 0 to 49
Data columns (total 33 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   id                       50 non-null     int64  
 1   diagnosis                50 non-null     object 
 2   radius_mean              50 non-null     float64
 3   texture_mean             50 non-null     float64
 4   perimeter_mean           50 non-null     float64
 5   area_mean                50 non-null     float64
 6   smoothness_mean          50 non-null     float64
 7   compactness_mean         50 non-null     float64
 8   concavity_mean           50 non-null     float64
 9   concave points_mean      50 non-null     float64
 10  symmetry_mean            50 non-null     float64
 11  fractal_dimension_mean   50 non-null     float64
 12  radius_se                50 non-null     float64
 13  texture_se               50 non-null     float64
 14  perimeter_se             50 non-null     float64
 15  area_se                  50 non-null     float64
 16  smoothness_se            50 non-null     float64
 17  compactness_se           50 non-null     float64
 18  concavity_se             50 non-null     float64
 19  concave points_se        50 non-null     float64
 20  symmetry_se              50 non-null     float64
 21  fractal_dimension_se     50 non-null     float64
 22  radius_worst             50 non-null     float64
 23  texture_worst            50 non-null     float64
 24  perimeter_worst          50 non-null     float64
 25  area_worst               50 non-null     float64
 26  smoothness_worst         50 non-null     float64
 27  compactness_worst        50 non-null     float64
 28  concavity_worst          50 non-null     float64
 29  concave points_worst     50 non-null     float64
 30  symmetry_worst           50 non-null     float64
 31  fractal_dimension_worst  50 non-null     float64
 32  Unnamed: 32              0 non-null      float64
dtypes: float64(31), int64(1), object(1)
memory usage: 13.0+ KB
```

In [5]:

```
df.describe()
```

Out[5]:

	id	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_
count	5.000000e+01	50.000000	50.000000	50.00000	50.000000	50.0(
mean	2.159747e+07	15.377200	20.178400	101.40000	761.998000	0.1(
std	3.594020e+07	3.011576	3.828225	20.43648	294.515104	0.0'
min	8.571500e+04	8.196000	10.380000	51.71000	201.900000	0.0{
25%	8.525718e+05	13.197500	17.797500	86.36250	537.250000	0.0{
50%	8.550370e+05	15.145000	20.825000	99.90500	701.600000	0.1(
75%	8.511056e+06	17.885000	22.532500	118.70000	989.525000	0.1'
max	8.571370e+07	21.160000	27.540000	137.20000	1404.000000	0.1<

8 rows × 32 columns

In [6]:

```
df.columns
```

Out[6]:

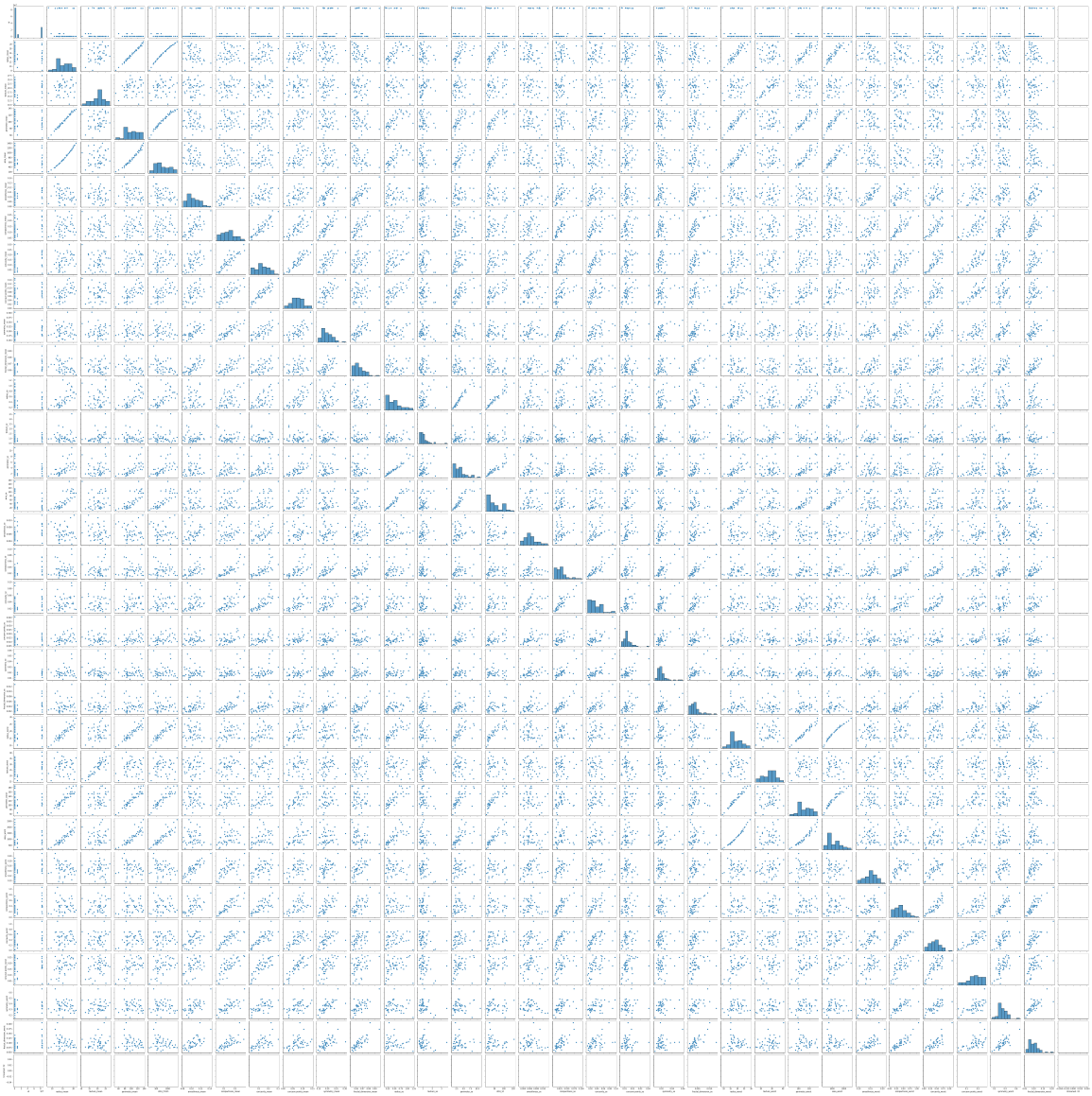
```
Index(['id', 'diagnosis', 'radius_mean', 'texture_mean', 'perimeter_mean',
       'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mea
n',
       'concave points_mean', 'symmetry_mean', 'fractal_dimension_mean',
       'radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_s
e',
       'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_s
e',
       'fractal_dimension_se', 'radius_worst', 'texture_worst',
       'perimeter_worst', 'area_worst', 'smoothness_worst',
       'compactness_worst', 'concavity_worst', 'concave points_worst',
       'symmetry_worst', 'fractal_dimension_worst', 'Unnamed: 32'],
      dtype='object')
```

In [7]:

```
sns.pairplot(df)
```

Out[7]:

```
<seaborn.axisgrid.PairGrid at 0x21466000310>
```



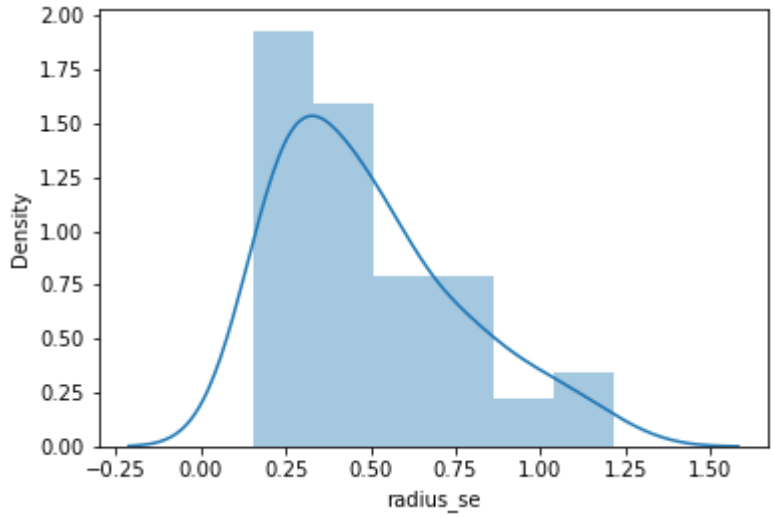
In [8]:

```
sns.distplot(df['radius_se'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557:
FutureWarning: `distplot` is a deprecated function and will be removed in
a future version. Please adapt your code to use either `displot` (a figure
-level function with similar flexibility) or `histplot` (an axes-level fun
ction for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[8]:

```
<AxesSubplot:xlabel='radius_se', ylabel='Density'>
```

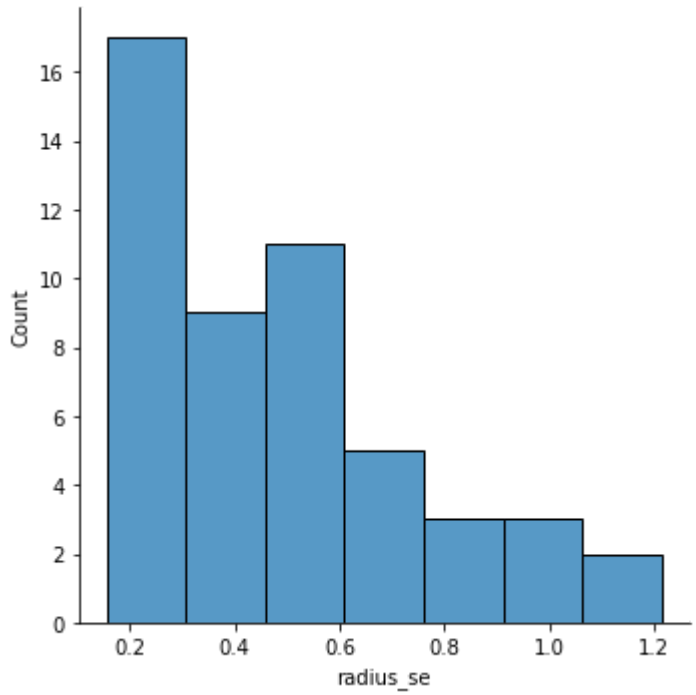


In [9]:

```
sns.displot(df["radius_se"])
```

Out[9]:

```
<seaborn.axisgrid.FacetGrid at 0x2140a3577c0>
```



In [10]:

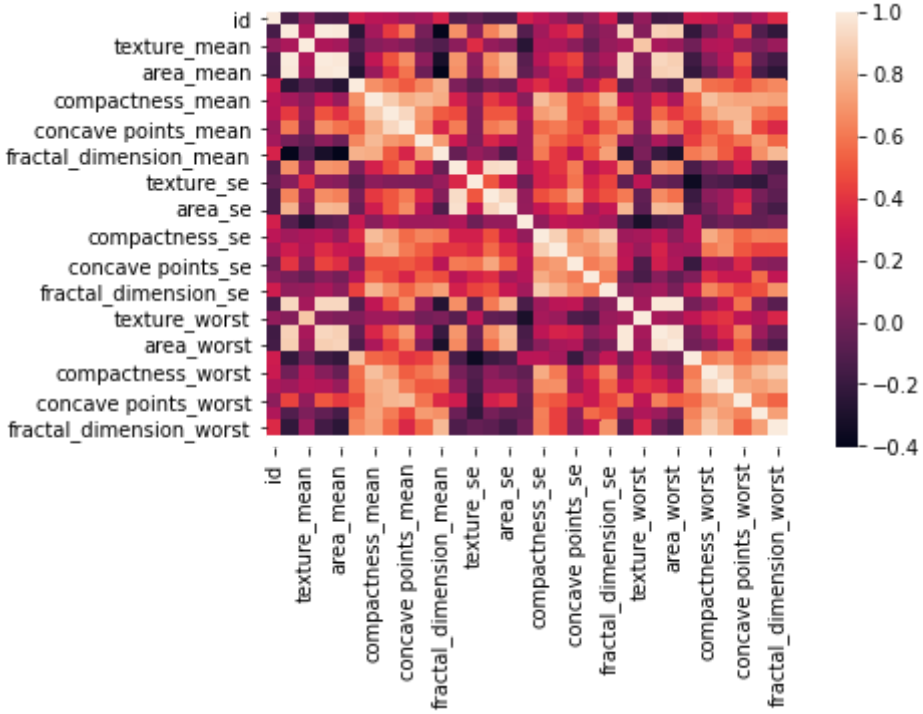
```
df1=df[['id', 'diagnosis', 'radius_mean', 'texture_mean', 'perimeter_mean',
       'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mean',
       'concave points_mean', 'symmetry_mean', 'fractal_dimension_mean',
       'radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_se',
       'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_se',
       'fractal_dimension_se', 'radius_worst', 'texture_worst',
       'perimeter_worst', 'area_worst', 'smoothness_worst',
       'compactness_worst', 'concavity_worst', 'concave points_worst',
       'symmetry_worst', 'fractal_dimension_worst', 'Unnamed: 32']]
```

In [11]:

```
sns.heatmap(df1.corr())
```

Out[11]:

```
<AxesSubplot:>
```



In [12]:

```
df2=df.dropna(axis=1)
df2
```

Out[12]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothnes
0	842302	M	17.990	10.38	122.80	1001.0	
1	842517	M	20.570	17.77	132.90	1326.0	
2	84300903	M	19.690	21.25	130.00	1203.0	
3	84348301	M	11.420	20.38	77.58	386.1	
4	84358402	M	20.290	14.34	135.10	1297.0	
5	843786	M	12.450	15.70	82.57	477.1	
6	844359	M	18.250	19.98	119.60	1040.0	
7	84458202	M	13.710	20.83	90.20	577.9	
8	844981	M	13.000	21.82	87.50	519.8	
9	84501001	M	12.460	24.04	83.97	475.9	
10	845636	M	16.020	23.24	102.70	797.8	
11	84610002	M	15.780	17.89	103.60	781.0	
12	846226	M	19.170	24.80	132.40	1123.0	
13	846381	M	15.850	23.95	103.70	782.7	
14	84667401	M	13.730	22.61	93.60	578.3	
15	84799002	M	14.540	27.54	96.73	658.8	
16	848406	M	14.680	20.13	94.74	684.5	
17	84862001	M	16.130	20.68	108.10	798.8	
18	849014	M	19.810	22.15	130.00	1260.0	
19	8510426	B	13.540	14.36	87.46	566.3	
20	8510653	B	13.080	15.71	85.63	520.0	
21	8510824	B	9.504	12.44	60.34	273.9	
22	8511133	M	15.340	14.26	102.50	704.4	
23	851509	M	21.160	23.04	137.20	1404.0	
24	852552	M	16.650	21.38	110.00	904.6	
25	852631	M	17.140	16.40	116.00	912.7	
26	852763	M	14.580	21.53	97.41	644.8	
27	852781	M	18.610	20.25	122.10	1094.0	
28	852973	M	15.300	25.27	102.40	732.4	
29	853201	M	17.570	15.05	115.00	955.1	
30	853401	M	18.630	25.11	124.80	1088.0	
31	853612	M	11.840	18.70	77.93	440.6	
32	85382601	M	17.020	23.98	112.80	899.3	
33	854002	M	19.270	26.47	127.90	1162.0	
34	854039	M	16.130	17.88	107.00	807.2	
35	854253	M	16.740	21.59	110.10	869.5	

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothnes
36	854268	M	14.250	21.72	93.63	633.0	
37	854941	B	13.030	18.42	82.61	523.8	
38	855133	M	14.990	25.20	95.54	698.8	
39	855138	M	13.480	20.82	88.40	559.2	
40	855167	M	13.440	21.58	86.18	563.0	
41	855563	M	10.950	21.35	71.90	371.1	
42	855625	M	19.070	24.81	128.30	1104.0	
43	856106	M	13.280	20.28	87.32	545.2	
44	85638502	M	13.170	21.81	85.42	531.5	
45	857010	M	18.650	17.60	123.70	1076.0	
46	85713702	B	8.196	16.84	51.71	201.9	
47	85715	M	13.170	18.66	85.98	534.6	
48	857155	B	12.050	14.63	78.04	449.3	
49	857156	B	13.490	22.30	86.91	561.0	

50 rows × 32 columns

In [13]:

```
x=df2[['id', 'radius_mean', 'texture_mean', 'perimeter_mean',
       'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mean',
       'concave points_mean', 'symmetry_mean', 'fractal_dimension_mean',
       'radius_se', 'texture_se', 'perimeter_se', 'smoothness_se',
       'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_se',
       'fractal_dimension_se', 'radius_worst', 'texture_worst',
       'perimeter_worst', 'area_worst', 'smoothness_worst',
       'compactness_worst', 'concavity_worst', 'concave points_worst',
       'symmetry_worst', 'fractal_dimension_worst']]
y=df2[['area_se']]
```

In [14]:

```
from sklearn.model_selection import train_test_split
```

In [15]:

```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

In [16]:

```
from sklearn.linear_model import LinearRegression

lr=LinearRegression()
lr.fit(x_train,y_train)#ValueError: Input contains NaN, infinity or a value too large for
```

Out[16]:

```
LinearRegression()
```

In [17]:

```
print(lr.intercept_)
```

```
[79.02636858]
```

In [18]:

```
coef= pd.DataFrame(lr.coef_)
coef
```

Out[18]:

	0	1	2	3	4	5	6	7
0	3.015544e-08	-50.719007	0.887374	5.922933	0.106259	-168.97597	-224.90342	-131.177365

1 rows × 30 columns

In [19]:

```
print(lr.score(x_test,y_test))
```

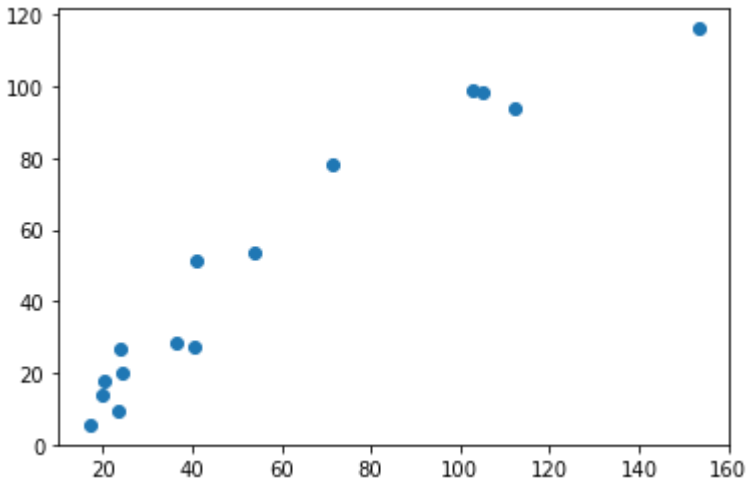
```
0.8988700335875313
```

In [20]:

```
prediction = lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[20]:

```
<matplotlib.collections.PathCollection at 0x2141b95edc0>
```



In [21]:

```
lr.score(x_test,y_test)
```

Out[21]:

```
0.8988700335875313
```

In [22]:

```
lr.score(x_train,y_train)
```

Out[22]:

```
0.9994733931996888
```

In [23]:

```
from sklearn.linear_model import Ridge,Lasso
```

In [24]:

```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[24]:

```
Ridge(alpha=10)
```

In [25]:

```
rr.score(x_test,y_test)
```

Out[25]:

```
0.9010391468514849
```

In [26]:

```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[26]:

```
Lasso(alpha=10)
```

In [27]:

```
la.score(x_test,y_test)
```

Out[27]:

```
0.7938373092844463
```