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

In [32]:
```
x=pd.read_csv(r"C:\Users\user\Downloads\8_BreastCancerPrediction.csv")
```

Out[32]:

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

569 rows × 33 columns

In [33]:
```
x=x.head(100)
```

Out[33]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_r
0	842302	M	17.990	10.38	122.80	1001.0	0.1
1	842517	M	20.570	17.77	132.90	1326.0	0.0
2	84300903	M	19.690	21.25	130.00	1203.0	0.1
3	84348301	M	11.420	20.38	77.58	386.1	0.1
4	84358402	M	20.290	14.34	135.10	1297.0	0.1
...	...	...	...	...	...	...	
95	86208	M	20.260	23.03	132.40	1264.0	0.0
96	86211	B	12.180	17.84	77.79	451.1	0.1
97	862261	B	9.787	19.94	62.11	294.5	0.1
98	862485	B	11.600	12.84	74.34	412.6	0.0
99	862548	M	14.420	19.77	94.48	642.5	0.0

100 rows × 33 columns

In [34]:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 33 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   id                       100 non-null    int64  
 1   diagnosis                100 non-null    object 
 2   radius_mean              100 non-null    float64
 3   texture_mean             100 non-null    float64
 4   perimeter_mean           100 non-null    float64
 5   area_mean                100 non-null    float64
 6   smoothness_mean          100 non-null    float64
 7   compactness_mean         100 non-null    float64
 8   concavity_mean           100 non-null    float64
 9   concave points_mean      100 non-null    float64
 10  symmetry_mean            100 non-null    float64
 11  fractal_dimension_mean   100 non-null    float64
 12  radius_se                100 non-null    float64
 13  texture_se               100 non-null    float64
 14  perimeter_se             100 non-null    float64
 15  area_se                  100 non-null    float64
 16  smoothness_se            100 non-null    float64
 17  compactness_se           100 non-null    float64
 18  concavity_se             100 non-null    float64
 19  concave points_se        100 non-null    float64
 20  symmetry_se              100 non-null    float64
 21  fractal_dimension_se     100 non-null    float64
 22  radius_worst             100 non-null    float64
 23  texture_worst            100 non-null    float64
 24  perimeter_worst          100 non-null    float64
 25  area_worst               100 non-null    float64
 26  smoothness_worst         100 non-null    float64
 27  compactness_worst        100 non-null    float64
 28  concavity_worst          100 non-null    float64
 29  concave points_worst     100 non-null    float64
 30  symmetry_worst           100 non-null    float64
 31  fractal_dimension_worst  100 non-null    float64
 32  Unnamed: 32              0 non-null      float64
dtypes: float64(31), int64(1), object(1)
memory usage: 25.9+ KB
```

In [35]:

Out[35]:

```
Index(['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'],
      dtype='object')
```

In [52]:
```
d=x[['id','radius_mean', 'diagnosis',  'texture_mean', 'perimeter_mean',
'area_mean']]
```

Out[52]:

	id	radius_mean	diagnosis	texture_mean	perimeter_mean	area_mean
0	842302	17.990	M	10.38	122.80	1001.0
1	842517	20.570	M	17.77	132.90	1326.0
2	84300903	19.690	M	21.25	130.00	1203.0
3	84348301	11.420	M	20.38	77.58	386.1
4	84358402	20.290	M	14.34	135.10	1297.0
...	...	...	...	...	...	...
95	86208	20.260	M	23.03	132.40	1264.0
96	86211	12.180	B	17.84	77.79	451.1
97	862261	9.787	B	19.94	62.11	294.5
98	862485	11.600	B	12.84	74.34	412.6
99	862548	14.420	M	19.77	94.48	642.5

100 rows × 6 columns

In [37]:

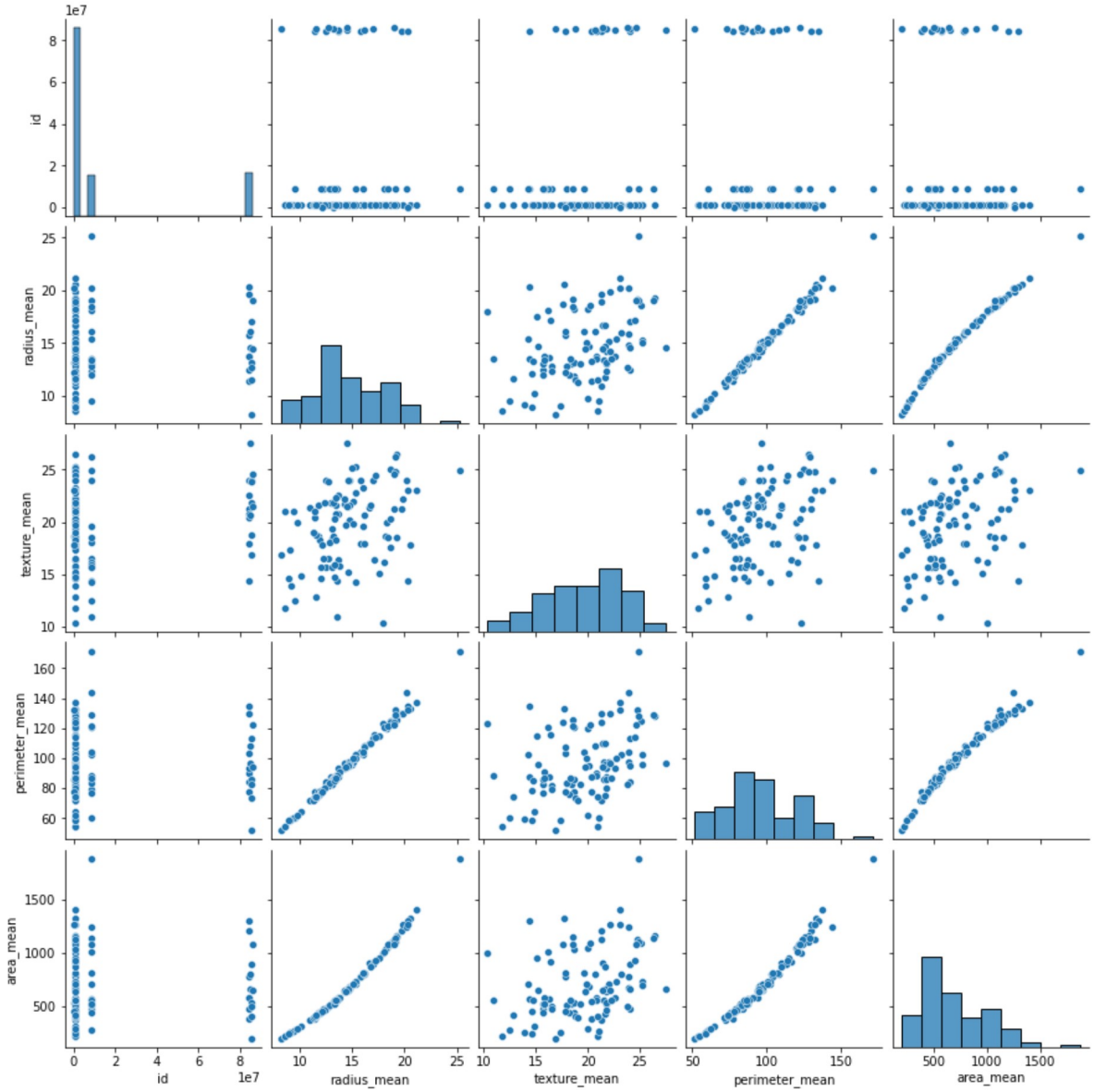
Out[37]:

	id	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mea
count	1.000000e+02	100.000000	100.000000	100.000000	100.000000	100.0000(
mean	1.547093e+07	14.707780	19.692200	96.471200	703.293000	0.1020-
std	3.066549e+07	3.349245	3.759176	23.187471	320.152301	0.0131!
min	8.571500e+04	8.196000	10.380000	51.710000	201.900000	0.0735!
25%	8.542642e+05	12.457500	16.760000	82.270000	476.800000	0.0934(
50%	8.593735e+05	14.335000	20.190000	94.365000	643.650000	0.1011!
75%	8.610460e+06	17.155000	22.150000	114.400000	916.875000	0.1103]
max	8.613550e+07	25.220000	27.540000	171.500000	1878.000000	0.1425(

8 rows × 32 columns

In [38]:

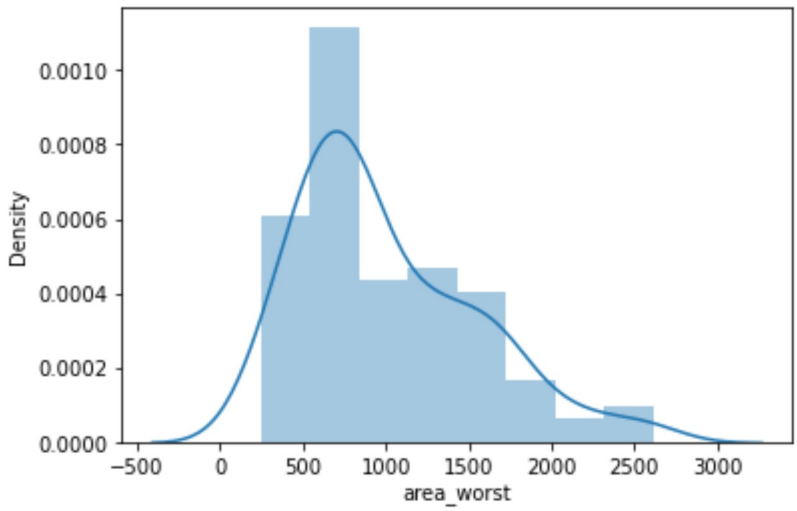
Out[38]: <seaborn.axisgrid.PairGrid at 0x22d08fd0e50>



In [40]:

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

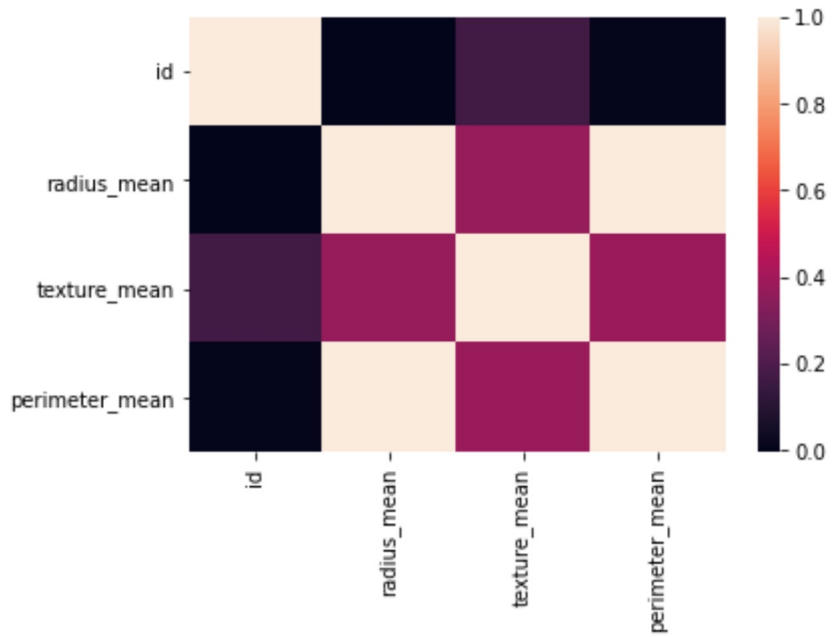
Out[40]: <AxesSubplot:xlabel='area_worst', ylabel='Density'>



In [55]:

In [56]:

Out[56]: <AxesSubplot:>



In [59]:

```
x=x1[['id',  'texture_mean', 'perimeter_mean']]
y=x1['radius_mean']
```

In [60]:
```
# to split my dataset into traning and test date

from sklearn.model_selection import train_test_split
```

In [61]:
```
from sklearn.linear_model import LinearRegression

lr=LinearRegression()
```

Out[61]:
```
LinearRegression()
```

In [62]:

```
0.8453417766051636
```

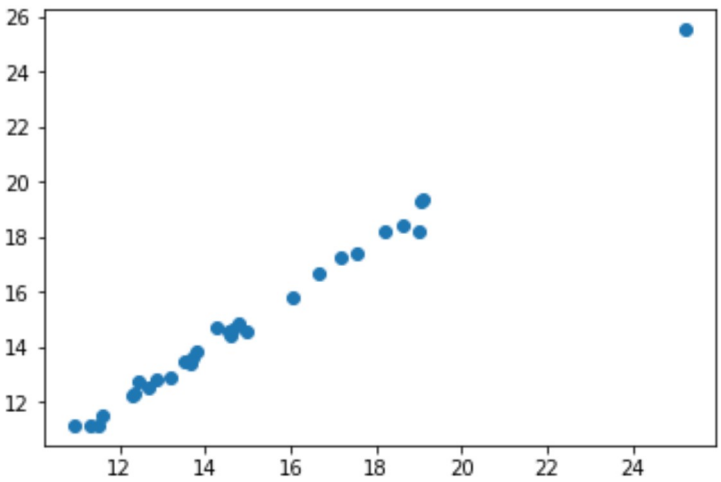
In [63]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
```

Out[63]:

	Co-efficient
id	-2.291171e-09
texture_mean	-3.837036e-03
perimeter_mean	1.446925e-01

In [64]:
```
prediction=lr.predict(x_test)
```

Out[64]:
```
<matplotlib.collections.PathCollection at 0x22d2ca70d30>
```



In [65]:

Out[65]:
```
0.9934076618045018
```

In [66]:

Out[66]:
```
0.9923979622027217
```

In [67]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [68]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[68]: 0.9934169553429775

In [69]:
```
la=Lasso(alpha=10)
```

Out[69]: Lasso(alpha=10)

In [70]:

Out[70]: 0.9811229954740027

In [ ]: