

In [1]:

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

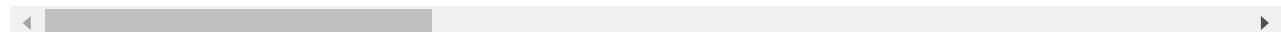
In [2]:

```
df=pd.read_csv("8_BreastCancerPrediction.csv")
df
```

Out[2]:

	<b>id</b>	<b>diagnosis</b>	<b>radius_mean</b>	<b>texture_mean</b>	<b>perimeter_mean</b>	<b>area_mean</b>	<b>smoothness_mean</b>	<b>compactness_mean</b>	<b>concavity_mean</b>	<b>concave points_mean</b>	<b>symmetry_mean</b>	<b>fractal_dimension_mean</b>	<b>radius_se</b>	<b>texture_se</b>	<b>perimeter_se</b>	<b>area_se</b>	<b>smoothness_se</b>	<b>compactness_se</b>	<b>concavity_se</b>	<b>concave points_se</b>	<b>symmetry_se</b>	<b>fractal_dimension_se</b>	<b>radius_worst</b>	<b>texture_worst</b>	<b>perimeter_worst</b>	<b>area_worst</b>	<b>smoothness_worst</b>	<b>compactness_worst</b>	<b>concavity_worst</b>	<b>concave points_worst</b>	<b>symmetry_worst</b>	<b>fractal_dimension_worst</b>	
<b>0</b>	842302	M	17.99	10.38	122.80	1001.0	0.11840																										
<b>1</b>	842517	M	20.57	17.77	132.90	1326.0	0.08474																										
<b>2</b>	84300903	M	19.69	21.25	130.00	1203.0	0.10960																										
<b>3</b>	84348301	M	11.42	20.38	77.58	386.1	0.14250																										
<b>4</b>	84358402	M	20.29	14.34	135.10	1297.0	0.10030																										
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...			
<b>564</b>	926424	M	21.56	22.39	142.00	1479.0	0.11100																										
<b>565</b>	926682	M	20.13	28.25	131.20	1261.0	0.09780																										
<b>566</b>	926954	M	16.60	28.08	108.30	858.1	0.08455																										
<b>567</b>	927241	M	20.60	29.33	140.10	1265.0	0.11780																										
<b>568</b>	92751	B	7.76	24.54	47.92	181.0	0.05263																										

569 rows × 33 columns



In [3]:

```
df.head()
```

Out[3]:

	<b>id</b>	<b>diagnosis</b>	<b>radius_mean</b>	<b>texture_mean</b>	<b>perimeter_mean</b>	<b>area_mean</b>	<b>smoothness_mean</b>	<b>compactness_mean</b>	<b>concavity_mean</b>	<b>concave points_mean</b>	<b>symmetry_mean</b>	<b>fractal_dimension_mean</b>	<b>radius_se</b>	<b>texture_se</b>	<b>perimeter_se</b>	<b>area_se</b>	<b>smoothness_se</b>	<b>compactness_se</b>	<b>concavity_se</b>	<b>concave points_se</b>	<b>symmetry_se</b>	<b>fractal_dimension_se</b>	<b>radius_worst</b>	<b>texture_worst</b>	<b>perimeter_worst</b>	<b>area_worst</b>	<b>smoothness_worst</b>	<b>compactness_worst</b>	<b>concavity_worst</b>	<b>concave points_worst</b>	<b>symmetry_worst</b>	<b>fractal_dimension_worst</b>
<b>0</b>	842302	M	17.99	10.38	122.80	1001.0	0.11840																									
<b>1</b>	842517	M	20.57	17.77	132.90	1326.0	0.08474																									
<b>2</b>	84300903	M	19.69	21.25	130.00	1203.0	0.10960																									
<b>3</b>	84348301	M	11.42	20.38	77.58	386.1	0.14250																									
<b>4</b>	84358402	M	20.29	14.34	135.10	1297.0	0.10030																									

5 rows × 33 columns



## DATA CLEANING AND DATA PREPROCESSING

In [4]:

`df.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 33 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   id               569 non-null    int64  
 1   diagnosis        569 non-null    object  
 2   radius_mean      569 non-null    float64 
 3   texture_mean     569 non-null    float64 
 4   perimeter_mean   569 non-null    float64 
 5   area_mean        569 non-null    float64 
 6   smoothness_mean  569 non-null    float64 
 7   compactness_mean 569 non-null    float64 
 8   concavity_mean   569 non-null    float64 
 9   concave_points_mean 569 non-null    float64 
 10  symmetry_mean   569 non-null    float64 
 11  fractal_dimension_mean 569 non-null    float64 
 12  radius_se        569 non-null    float64 
 13  texture_se       569 non-null    float64 
 14  perimeter_se    569 non-null    float64 
 15  area_se          569 non-null    float64 
 16  smoothness_se   569 non-null    float64 
 17  compactness_se  569 non-null    float64 
 18  concavity_se    569 non-null    float64 
 19  concave_points_se 569 non-null    float64 
 20  symmetry_se     569 non-null    float64 
 21  fractal_dimension_se 569 non-null    float64 
 22  radius_worst    569 non-null    float64 
 23  texture_worst   569 non-null    float64 
 24  perimeter_worst 569 non-null    float64 
 25  area_worst      569 non-null    float64 
 26  smoothness_worst 569 non-null    float64 
 27  compactness_worst 569 non-null    float64 
 28  concavity_worst 569 non-null    float64 
 29  concave_points_worst 569 non-null    float64 
 30  symmetry_worst  569 non-null    float64 
 31  fractal_dimension_worst 569 non-null    float64 
 32  Unnamed: 32      0 non-null    float64 
dtypes: float64(31), int64(1), object(1)
memory usage: 146.8+ KB
```

In [5]:

`df.describe()`

Out[5]:

	<b>id</b>	<b>radius_mean</b>	<b>texture_mean</b>	<b>perimeter_mean</b>	<b>area_mean</b>	<b>smoothness_mean</b>	<b>compactness_mean</b>	<b>concavity_mean</b>	<b>concave_points_mean</b>	<b>symmetry_mean</b>	<b>fractal_dimension_mean</b>	<b>radius_se</b>	<b>texture_se</b>	<b>perimeter_se</b>	<b>area_se</b>	<b>smoothness_se</b>	<b>compactness_se</b>	<b>concavity_se</b>	<b>concave_points_se</b>	<b>symmetry_se</b>	<b>fractal_dimension_se</b>	<b>radius_worst</b>	<b>texture_worst</b>	<b>perimeter_worst</b>	<b>area_worst</b>	<b>smoothness_worst</b>	<b>compactness_worst</b>	<b>concavity_worst</b>	<b>concave_points_worst</b>	<b>symmetry_worst</b>	<b>fractal_dimension_worst</b>	<b>Unnamed: 32</b>
<b>count</b>	5.690000e+02	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000																					
<b>mean</b>	3.037183e+07	14.127292	19.289649	91.969033	654.889104	0.096360																										
<b>std</b>	1.250206e+08	3.524049	4.301036	24.298981	351.914129	0.014064																										
<b>min</b>	8.670000e+03	6.981000	9.710000	43.790000	143.500000	0.052630																										
<b>25%</b>	8.692180e+05	11.700000	16.170000	75.170000	420.300000	0.086370																										
<b>50%</b>	9.060240e+05	13.370000	18.840000	86.240000	551.100000	0.095870																										
<b>75%</b>	8.813129e+06	15.780000	21.800000	104.100000	782.700000	0.105300																										

```
   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  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
max  9.113205e+08      28.110000      39.280000     188.500000    2501.000000        0.163400
```

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\_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 [7]: df1=df.dropna(axis=1)  
df1

Out[7]:

	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		
0	842302	M	17.99	10.38	122.80	1001.0	0.11840																												
1	842517	M	20.57	17.77	132.90	1326.0	0.08474																												
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960																												
3	84348301	M	11.42	20.38	77.58	386.1	0.14250																												
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030																												
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...		
564	926424	M	21.56	22.39	142.00	1479.0	0.11100																												
565	926682	M	20.13	28.25	131.20	1261.0	0.09780																												
566	926954	M	16.60	28.08	108.30	858.1	0.08455																												
567	927241	M	20.60	29.33	140.10	1265.0	0.11780																												
568	92751	B	7.76	24.54	47.92	181.0	0.05263																												

569 rows × 32 columns

In [8]: df1.columns

Out[8]: 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')

```
'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'],
dtype='object')
```

In [9]:

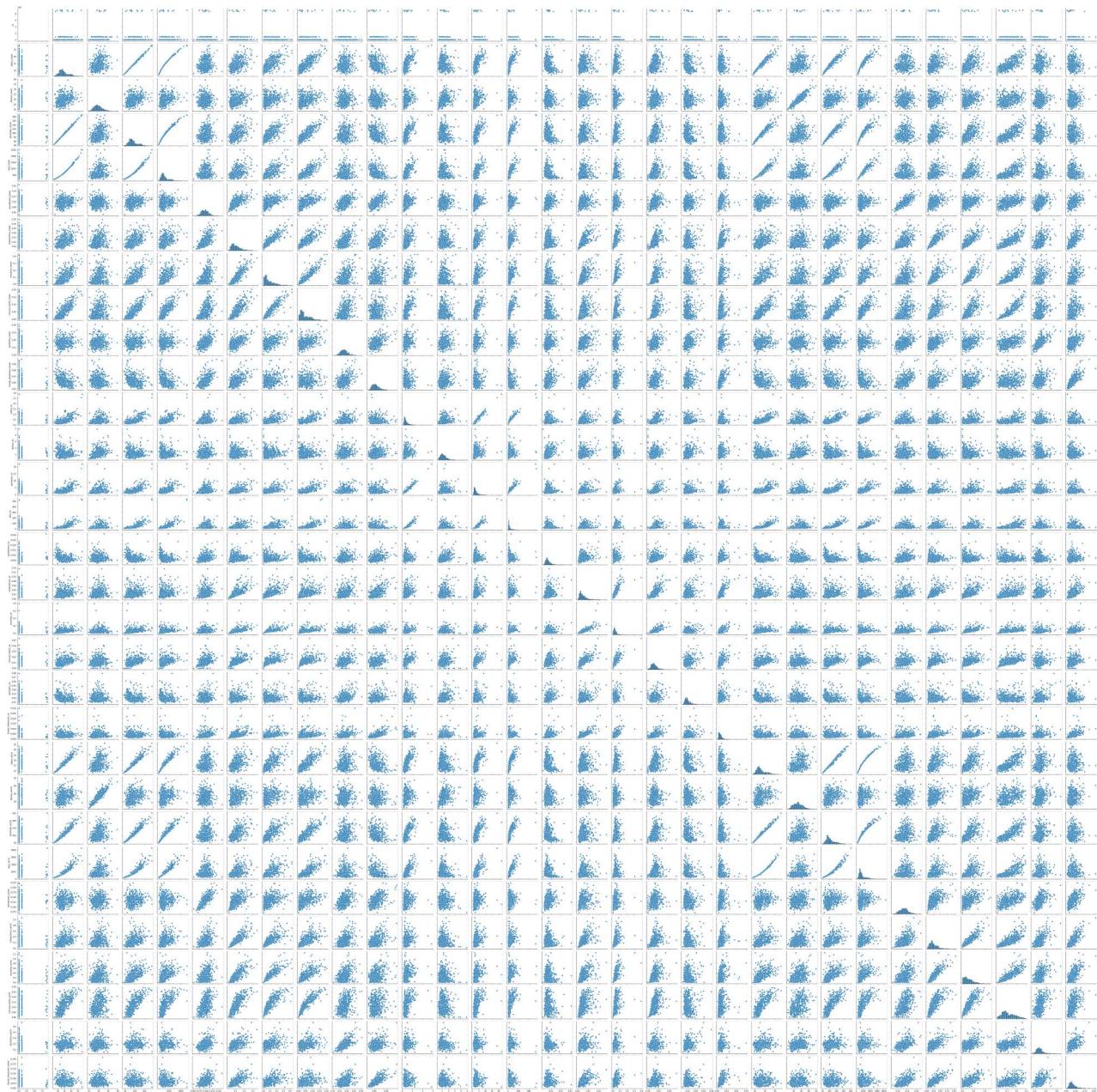
```
df1=df1[['id', 'radius_mean', 'texture_mean', 'perimeter_mean',
'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mean',
'concave points_mean', 'symmetry_mean', 'fractal_dimension_mean',
'radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_se',
'compactness_se', 'concavity_se', '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']]
```

## EDA AND VISUALIZATION

In [10]:

```
sns.pairplot(df1)
```

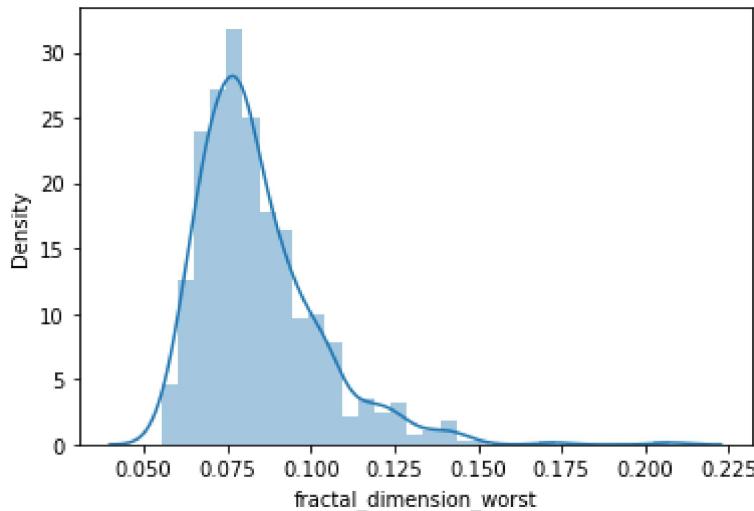
Out[10]: &lt;seaborn.axisgrid.PairGrid at 0x2d2cb11f520&gt;



```
In [11]: sns.distplot(df1['fractal_dimension_worst'])
```

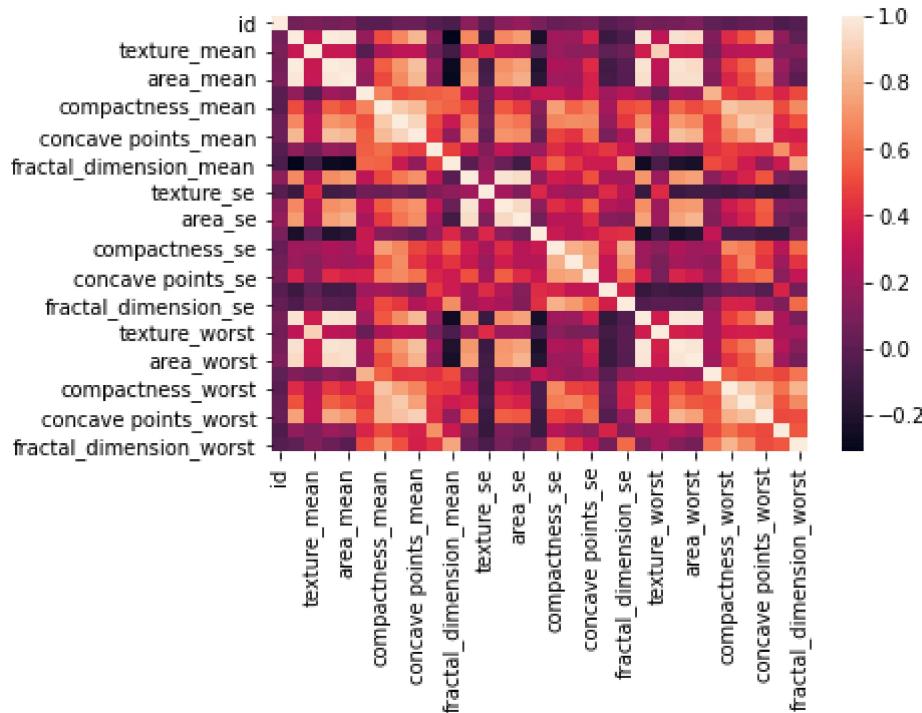
```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning:  
`distplot` is a deprecated function and will be removed in a future version. Please adapt  
your code to use either `displot` (a figure-level function with similar flexibility)  
or `histplot` (an axes-level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

```
Out[11]: <AxesSubplot:xlabel='fractal_dimension_worst', ylabel='Density'>
```



```
In [12]: sns.heatmap(df1.corr())
```

Out[12]: <AxesSubplot:>



## TO TRAIN THE MODEL AND MODEL BUILDING

```
In [13]: x=df[['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', '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']]
y=df['fractal_dimension_worst']
```

```
In [14]: from sklearn.model_selection import train_test_split  
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

```
In [15]: from sklearn.linear_model import LinearRegression  
lr=LinearRegression()  
lr.fit(x_train,y_train)
```

```
Out[15]: LinearRegression()
```

```
In [16]: lr.intercept_
```

```
Out[16]: -0.02356960802040628
```

```
In [17]: coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])  
coeff
```

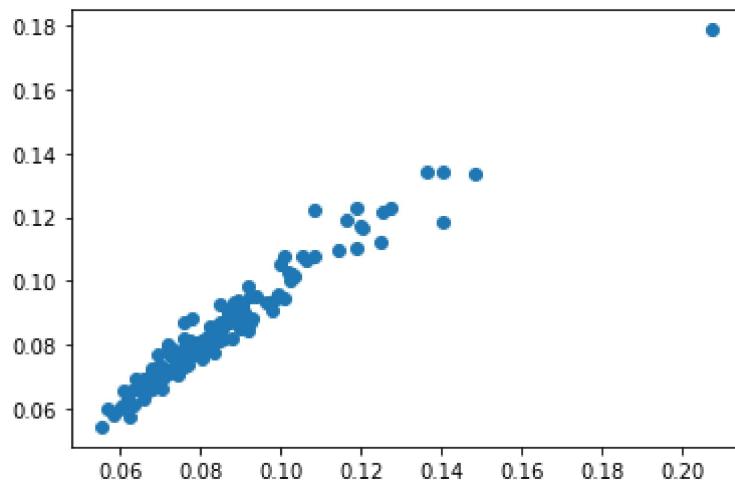
	Co-efficient
<b>id</b>	-1.669436e-12
<b>radius_mean</b>	-3.945456e-03
<b>texture_mean</b>	1.133743e-04
<b>perimeter_mean</b>	5.318178e-04
<b>area_mean</b>	5.081494e-07
<b>smoothness_mean</b>	-8.416602e-02
<b>compactness_mean</b>	-1.216304e-01
<b>concavity_mean</b>	-1.237901e-02
<b>concave points_mean</b>	1.127890e-02
<b>symmetry_mean</b>	-6.885636e-04
<b>fractal_dimension_mean</b>	1.291360e+00
<b>radius_se</b>	-1.821690e-02
<b>texture_se</b>	5.467393e-05
<b>perimeter_se</b>	1.483691e-03
<b>area_se</b>	5.175133e-05
<b>smoothness_se</b>	-6.036573e-02
<b>compactness_se</b>	-1.751672e-01
<b>concavity_se</b>	-6.920498e-02
<b>concave points_se</b>	-8.579859e-03
<b>symmetry_se</b>	-1.333066e-01
<b>fractal_dimension_se</b>	2.310160e+00

**Co-efficient**

<b>radius_worst</b>	3.868798e-03
<b>texture_worst</b>	-5.688160e-05
<b>perimeter_worst</b>	-3.760635e-04
<b>area_worst</b>	-8.021631e-06
<b>smoothness_worst</b>	8.456933e-02
<b>compactness_worst</b>	8.123840e-02
<b>concavity_worst</b>	1.378207e-02
<b>concave points_worst</b>	-5.144287e-03
<b>symmetry_worst</b>	1.825495e-02

```
In [18]: prediction =lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[18]: <matplotlib.collections.PathCollection at 0x2d2826533a0>



## ACCURACY

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

Out[19]: 0.941879713808282

```
In [20]: lr.score(x_train,y_train)
```

Out[20]: 0.9449331611748284

```
In [21]: from sklearn.linear_model import Ridge,Lasso
```

In [22]:

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

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_ridge.py:147: LinAlgWarning: Ill-conditioned matrix (rcond=1.06752e-18): result may not be accurate.
    return linalg.solve(A, Xy, sym_pos=True,
```

Out[22]: Ridge(alpha=10)

In [23]:

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

Out[23]: 0.6570885730458311

In [24]:

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

Out[24]: 0.7119103564527788

In [25]:

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

Out[25]: Lasso(alpha=10)

In [26]:

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

Out[26]: 0.004771619360980028

In [27]:

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

Out[27]: -0.012815401943230942