Problem Statement

Linear Regression

Import Libraries

In [20]:

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

In [21]:

```
a=pd.read_csv("cancer.csv")
a
```

Out[21]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mea
0	842302	M	17.99	10.38	122.80	1001.0	0.1184
1	842517	M	20.57	17.77	132.90	1326.0	0.0847
2	84300903	M	19.69	21.25	130.00	1203.0	0.1096
3	84348301	M	11.42	20.38	77.58	386.1	0.1425
4	84358402	M	20.29	14.34	135.10	1297.0	0.1003
...	...	...	...	...	...	...	
564	926424	M	21.56	22.39	142.00	1479.0	0.1110
565	926682	M	20.13	28.25	131.20	1261.0	0.0978
566	926954	M	16.60	28.08	108.30	858.1	0.0845
567	927241	M	20.60	29.33	140.10	1265.0	0.1178
568	92751	B	7.76	24.54	47.92	181.0	0.0526

569 rows × 32 columns

To display top 10 rows

In [22]:

```
c=a.head(15)
c
```

Out[22]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean
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

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean
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
5	843786	M	12.45	15.70	82.57	477.1	0.12780
6	844359	M	18.25	19.98	119.60	1040.0	0.09463
7	84458202	M	13.71	20.83	90.20	577.9	0.11890
8	844981	M	13.00	21.82	87.50	519.8	0.12730
9	84501001	M	12.46	24.04	83.97	475.9	0.11860
10	845636	M	16.02	23.24	102.70	797.8	0.08206
11	84610002	M	15.78	17.89	103.60	781.0	0.09710
12	846226	M	19.17	24.80	132.40	1123.0	0.09740
13	846381	M	15.85	23.95	103.70	782.7	0.08401
14	84667401	M	13.73	22.61	93.60	578.3	0.11310

15 rows × 32 columns

To find Missing values

In [23]:

```
c.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15 entries, 0 to 14
Data columns (total 32 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   id                       15 non-null     int64  
 1   diagnosis                15 non-null     object 
 2   radius_mean              15 non-null     float64
 3   texture_mean             15 non-null     float64
 4   perimeter_mean           15 non-null     float64
 5   area_mean                15 non-null     float64
 6   smoothness_mean          15 non-null     float64
 7   compactness_mean         15 non-null     float64
 8   concavity_mean           15 non-null     float64
 9   concave points_mean      15 non-null     float64
 10  symmetry_mean            15 non-null     float64
 11  fractal_dimension_mean   15 non-null     float64
 12  radius_se                15 non-null     float64
 13  texture_se               15 non-null     float64
 14  perimeter_se             15 non-null     float64
 15  area_se                  15 non-null     float64
 16  smoothness_se            15 non-null     float64
 17  compactness_se           15 non-null     float64
 18  concavity_se             15 non-null     float64
 19  concave points_se        15 non-null     float64
 20  symmetry_se              15 non-null     float64
 21  fractal_dimension_se     15 non-null     float64
 22  radius_worst             15 non-null     float64
 23  texture_worst            15 non-null     float64
 24  perimeter_worst          15 non-null     float64
 25  area_worst               15 non-null     float64
 26  smoothness_worst         15 non-null     float64
 27  compactness_worst        15 non-null     float64
 28  concavity_worst          15 non-null     float64
 29  concave points_worst     15 non-null     float64
 30  symmetry_worst           15 non-null     float64
 31  fractal_dimension_worst  15 non-null     float64
dtypes: float64(30), int64(1), object(1)
memory usage: 3.9+ KB
```

To display summary of statistics

In [24]:

```
a.describe()
```

Out[24]:

	id	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean
count	5.690000e+02	569.000000	569.000000	569.000000	569.000000	569.000000
mean	3.037183e+07	14.127292	19.289649	91.969033	654.889104	0.096360
std	1.250206e+08	3.524049	4.301036	24.298981	351.914129	0.014064
min	8.670000e+03	6.981000	9.710000	43.790000	143.500000	0.052630
25%	8.692180e+05	11.700000	16.170000	75.170000	420.300000	0.086370
50%	9.060240e+05	13.370000	18.840000	86.240000	551.100000	0.095870
75%	8.813129e+06	15.780000	21.800000	104.100000	782.700000	0.105300
max	9.113205e+08	28.110000	39.280000	188.500000	2501.000000	0.163400

8 rows × 31 columns

To display column heading

In [25]:

```
a.columns
```

Out[25]:

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

In [ ]:

```
s=a.dropna(axis=1)
s
```

In [27]:

```
s.columns
```

Out[27]:

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

Distribution Plot

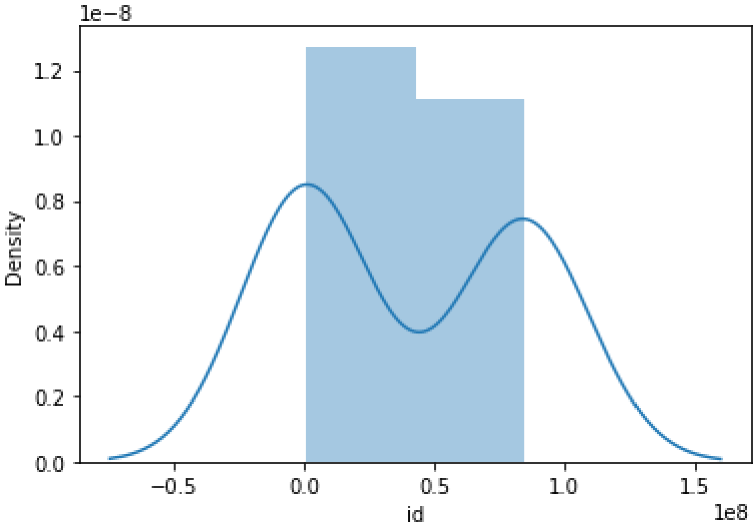
In [33]:

```
sns.distplot(c['id'])
```

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

Out[33]:

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



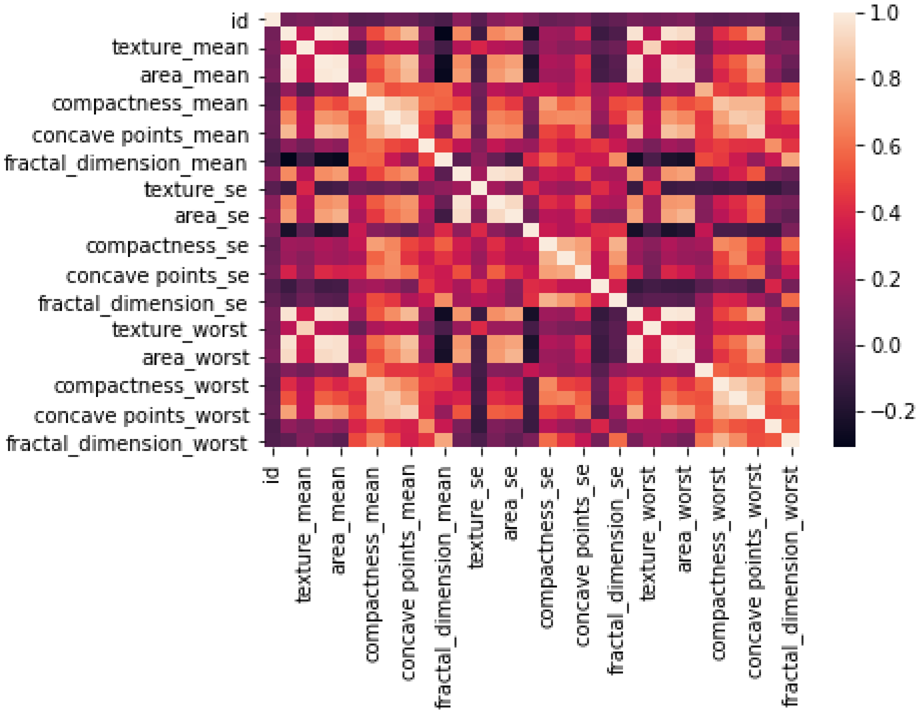
Correlation

In [34]:

```
b=a[['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']]
sns.heatmap(b.corr())
```

Out[34]:

```
<AxesSubplot:>
```



Train the model - Model Building

In [35]:

```
g=c[['id']]
h=c['id']
```

To split dataset into training end test

In [36]:

```
from sklearn.model_selection import train_test_split
g_train,g_test,h_train,h_test=train_test_split(g,h,test_size=0.6)
```

To run the model

In [37]:

```
from sklearn.linear_model import LinearRegression
```

In [38]:

```
lr=LinearRegression()
lr.fit(g_train,h_train)
```

Out[38]:

```
LinearRegression()
```

In [39]:

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

```
1.862645149230957e-09
```

Coeffecient

In [40]:

```
coeff=pd.DataFrame(lr.coef_,g.columns,columns=['Co-effecient'])
coeff
```

Out[40]:

	Co-effecient
id	1.0

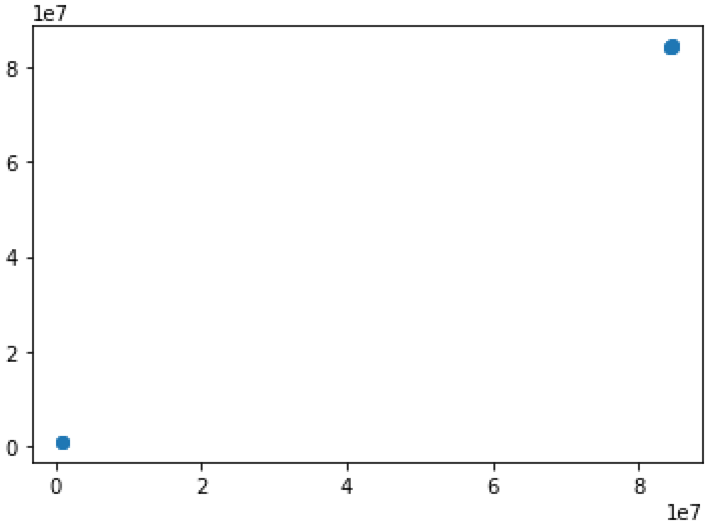
Best Fit line

In [41]:

```
prediction=lr.predict(g_test)
plt.scatter(h_test,prediction)
```

Out[41]:

```
<matplotlib.collections.PathCollection at 0x11e80381970>
```



To find score

In [42]:

```
print(lr.score(g_test,h_test))
```

```
1.0
```