Breast Cancer Prediction

In [1]:

```
#importing libraries and data reading,cleaning
```

In [2]:

```
import pandas as pd
from matplotlib import pyplot as plt
%matplotlib inline
```

In [3]:

```
df=pd.read_csv(r"C:\Users\shaik\Downloads\BreastCancerPrediction.csv")
df
```

Out[3]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothn
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 [21]:

```
df.head()
```

Out[21]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothnes
0	842302	M	0.521037	0.022658	122.80	1001.0	
1	842517	M	0.643144	0.272574	132.90	1326.0	
2	84300903	M	0.601496	0.390260	130.00	1203.0	
3	84348301	M	0.210090	0.360839	77.58	386.1	
4	84358402	M	0.629893	0.156578	135.10	1297.0	

5 rows × 35 columns

In [22]:

```
df.tail()
```

Out[22]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothnes
564	926424	M	0.690000	0.428813	142.00	1479.0	
565	926682	M	0.622320	0.626987	131.20	1261.0	
566	926954	M	0.455251	0.621238	108.30	858.1	
567	927241	M	0.644564	0.663510	140.10	1265.0	
568	92751	B	0.036869	0.501522	47.92	181.0	

5 rows × 35 columns

In [6]:

```
#dropping the null column
df.drop(['Unnamed: 32'],axis=1)
```

Out[6]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothn
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	

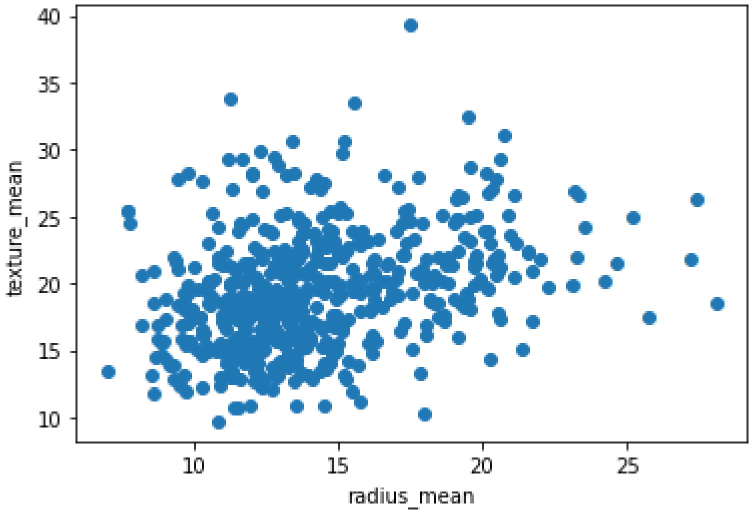
569 rows × 32 columns

In [7]:

```
plt.scatter(df["radius_mean"],df["texture_mean"])
plt.xlabel("radius_mean")
plt.ylabel("texture_mean")
```

Out[7]:

```
Text(0, 0.5, 'texture_mean')
```



In [8]:

```
from sklearn.cluster import KMeans
km=KMeans()
km
```

Out[8]:

```
KMeans()
```

In [9]:

```
y_predicted=km.fit_predict(df[["radius_mean","texture_mean"]])
y_predicted
```

Out[9]:

```
array([3, 2, 0, 1, 2, 3, 2, 6, 6, 6, 6, 2, 4, 6, 6, 5, 2, 2, 0, 3, 3, 7,
       3, 0, 2, 2, 6, 2, 6, 3, 4, 1, 4, 4, 2, 2, 6, 1, 6, 6, 6, 6, 4, 1,
       6, 2, 7, 1, 7, 6, 6, 3, 1, 2, 6, 1, 2, 6, 1, 7, 7, 1, 6, 7, 6, 6,
       1, 1, 1, 3, 2, 7, 4, 3, 1, 2, 7, 2, 4, 1, 6, 3, 0, 4, 7, 2, 6, 4,
       6, 3, 6, 6, 3, 1, 2, 0, 1, 1, 7, 1, 6, 7, 1, 1, 1, 3, 1, 1, 0, 6,
       1, 6, 1, 1, 7, 6, 7, 3, 6, 2, 7, 2, 0, 3, 3, 3, 6, 2, 3, 4, 7, 2,
       2, 3, 2, 6, 1, 7, 3, 7, 7, 2, 1, 3, 7, 7, 1, 2, 3, 1, 6, 1, 7, 7,
       3, 1, 2, 2, 7, 7, 1, 2, 2, 6, 0, 2, 7, 2, 4, 3, 7, 1, 3, 7, 7, 7,
       1, 2, 6, 7, 0, 4, 2, 7, 6, 7, 2, 1, 1, 3, 6, 6, 1, 5, 6, 3, 6, 2,
       0, 6, 1, 2, 4, 6, 1, 3, 1, 2, 6, 3, 0, 1, 0, 4, 6, 3, 1, 1, 0, 4,
       3, 3, 1, 2, 3, 3, 7, 3, 6, 6, 2, 5, 5, 4, 7, 6, 4, 0, 5, 5, 3, 3,
       1, 6, 4, 1, 1, 3, 6, 7, 0, 1, 2, 2, 2, 3, 4, 3, 6, 5, 4, 4, 2, 2,
       2, 4, 1, 6, 3, 1, 3, 7, 0, 7, 4, 1, 7, 2, 1, 3, 4, 7, 2, 2, 3, 1,
       1, 7, 1, 1, 1, 2, 3, 1, 7, 3, 7, 1, 1, 6, 2, 1, 4, 1, 1, 6, 3, 7,
       3, 3, 1, 3, 7, 7, 1, 1, 7, 2, 1, 1, 7, 2, 7, 0, 7, 1, 3, 1, 2, 2,
       3, 1, 1, 7, 1, 2, 3, 2, 1, 0, 3, 1, 7, 0, 7, 7, 1, 3, 7, 7, 1, 2,
       0, 6, 7, 1, 1, 3, 7, 1, 1, 6, 1, 2, 3, 0, 4, 1, 0, 0, 6, 3, 2, 2,
       3, 3, 1, 5, 3, 1, 7, 7, 6, 1, 3, 6, 7, 3, 7, 4, 7, 1, 2, 0, 1, 3,
       1, 1, 7, 1, 2, 7, 1, 3, 7, 1, 3, 6, 2, 1, 1, 1, 6, 6, 5, 6, 6, 2,
       7, 6, 1, 3, 7, 1, 1, 1, 7, 6, 1, 1, 6, 1, 2, 2, 3, 1, 1, 3, 1, 3,
       1, 4, 3, 1, 2, 6, 4, 3, 1, 0, 6, 4, 5, 3, 1, 5, 5, 6, 6, 5, 4, 0,
       5, 1, 1, 1, 6, 1, 4, 1, 1, 5, 3, 5, 7, 3, 6, 3, 7, 2, 1, 1, 3, 1,
       3, 3, 3, 2, 7, 2, 6, 3, 2, 7, 6, 2, 1, 1, 2, 0, 3, 6, 3, 0, 7, 7,
       1, 1, 3, 6, 7, 3, 6, 3, 2, 1, 2, 2, 1, 3, 7, 0, 1, 1, 7, 7, 1, 7,
       3, 7, 1, 1, 3, 0, 1, 0, 6, 6, 6, 6, 7, 6, 6, 5, 6, 6, 7, 1, 1, 6,
       6, 6, 5, 6, 5, 5, 1, 5, 6, 6, 5, 5, 5, 4, 0, 4, 4, 4, 6])
```

In [10]:

```
df["cluster"]=y_predicted
df.head()
```

Out[10]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothnes
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	

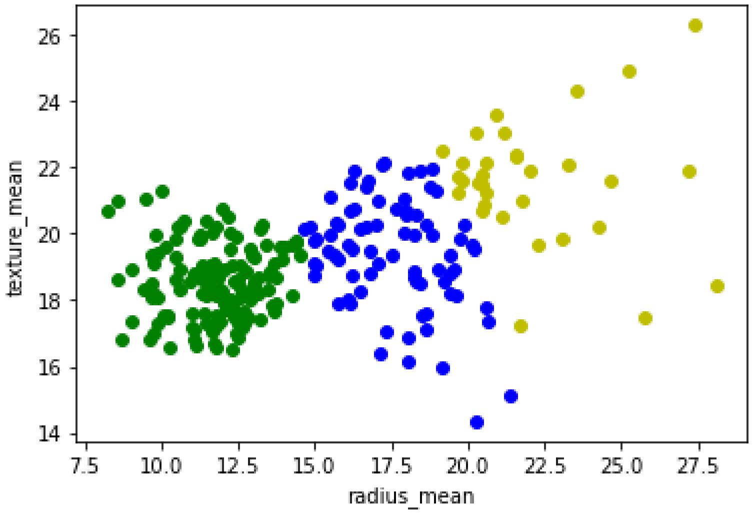
5 rows × 34 columns

In [11]:

```
df1=df[df.cluster==0]
df2=df[df.cluster==1]
df3=df[df.cluster==2]
plt.scatter(df1["radius_mean"],df1["texture_mean"],color="y")
plt.scatter(df2["radius_mean"],df2["texture_mean"],color="green")
plt.scatter(df3["radius_mean"],df3["texture_mean"],color="blue")
plt.xlabel("radius_mean")
plt.ylabel("texture_mean")
```

Out[11]:

```
Text(0, 0.5, 'texture_mean')
```



In [12]:

```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
scaler.fit(df[["texture_mean"]])
df["texture_mean"]=scaler.transform(df[["texture_mean"]])
df.head()
```

Out[12]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothnes
0	842302	M	17.99	0.022658	122.80	1001.0	
1	842517	M	20.57	0.272574	132.90	1326.0	
2	84300903	M	19.69	0.390260	130.00	1203.0	
3	84348301	M	11.42	0.360839	77.58	386.1	
4	84358402	M	20.29	0.156578	135.10	1297.0	

5 rows × 34 columns

In [13]:

```
scaler.fit(df[["radius_mean"]])
df["radius_mean"]=scaler.transform(df[["radius_mean"]])
df.head()
```

Out[13]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothnes
0	842302	M	0.521037	0.022658	122.80	1001.0	
1	842517	M	0.643144	0.272574	132.90	1326.0	
2	84300903	M	0.601496	0.390260	130.00	1203.0	
3	84348301	M	0.210090	0.360839	77.58	386.1	
4	84358402	M	0.629893	0.156578	135.10	1297.0	

5 rows × 34 columns

In [14]:

```
y_predicted=km.fit_predict(df[["radius_mean","texture_mean"]])
y_predicted
```

Out[14]:

```
array([7, 4, 4, 5, 4, 7, 4, 3, 3, 0, 3, 7, 2, 3, 3, 0, 3, 3, 4, 7, 7, 1,
       7, 6, 3, 4, 3, 4, 3, 4, 2, 5, 2, 2, 7, 3, 3, 5, 0, 3, 3, 5, 2, 3,
       3, 4, 1, 5, 1, 3, 5, 7, 5, 4, 3, 5, 4, 3, 5, 1, 1, 5, 3, 1, 0, 3,
       5, 5, 5, 7, 4, 1, 2, 7, 5, 3, 7, 4, 2, 5, 5, 7, 6, 2, 1, 4, 3, 2,
       3, 7, 3, 3, 7, 5, 3, 2, 5, 5, 1, 3, 0, 1, 5, 5, 5, 7, 5, 5, 6, 5,
       5, 3, 3, 5, 1, 5, 1, 7, 3, 4, 1, 4, 6, 7, 7, 7, 0, 4, 7, 2, 1, 3,
       3, 7, 4, 3, 5, 1, 7, 1, 1, 7, 5, 7, 1, 1, 5, 3, 7, 7, 3, 5, 1, 1,
       7, 5, 4, 4, 1, 1, 5, 4, 4, 3, 6, 3, 1, 4, 2, 7, 1, 3, 7, 1, 1, 1,
       5, 3, 3, 7, 6, 2, 3, 1, 3, 1, 4, 5, 5, 7, 3, 3, 5, 0, 3, 7, 3, 4,
       4, 3, 5, 4, 6, 3, 5, 7, 5, 4, 3, 7, 4, 5, 6, 2, 3, 7, 5, 5, 4, 2,
       7, 7, 5, 3, 7, 7, 1, 7, 0, 3, 4, 0, 0, 2, 1, 3, 6, 4, 0, 2, 7, 7,
       5, 3, 2, 5, 7, 7, 0, 1, 2, 5, 4, 4, 4, 7, 2, 7, 3, 0, 2, 2, 4, 3,
       4, 2, 5, 3, 7, 5, 7, 1, 6, 1, 2, 5, 1, 4, 7, 7, 2, 1, 4, 3, 7, 5,
       5, 7, 5, 5, 3, 3, 7, 5, 7, 7, 1, 5, 7, 5, 4, 5, 2, 5, 5, 0, 7, 1,
       7, 7, 5, 7, 7, 1, 5, 5, 1, 4, 5, 5, 1, 4, 7, 4, 1, 5, 7, 5, 3, 3,
       7, 5, 5, 1, 5, 4, 7, 4, 5, 6, 7, 1, 1, 4, 1, 1, 5, 7, 1, 1, 5, 3,
       6, 0, 1, 5, 5, 7, 1, 5, 5, 3, 5, 4, 7, 4, 2, 5, 4, 6, 3, 7, 4, 4,
       7, 7, 5, 0, 7, 5, 1, 1, 3, 5, 7, 3, 1, 7, 1, 2, 1, 1, 3, 6, 5, 7,
       3, 5, 1, 5, 4, 1, 5, 7, 1, 5, 7, 3, 4, 5, 5, 5, 5, 3, 0, 5, 5, 3,
       1, 5, 5, 7, 1, 3, 5, 5, 1, 5, 5, 5, 3, 5, 4, 4, 7, 3, 5, 7, 3, 7,
       5, 2, 7, 5, 4, 0, 2, 7, 3, 4, 5, 2, 0, 7, 5, 0, 0, 0, 0, 0, 2, 6,
       0, 5, 5, 3, 3, 5, 2, 5, 5, 0, 7, 0, 1, 7, 3, 7, 1, 3, 5, 3, 7, 7,
       7, 7, 7, 4, 1, 4, 3, 7, 4, 1, 3, 3, 5, 5, 4, 4, 7, 0, 7, 6, 1, 1,
       5, 5, 7, 3, 1, 7, 3, 7, 3, 5, 4, 4, 5, 7, 1, 6, 5, 3, 1, 1, 3, 1,
       7, 1, 5, 5, 7, 4, 5, 4, 3, 0, 0, 0, 1, 0, 0, 0, 3, 3, 1, 1, 5, 0,
       5, 5, 0, 5, 0, 0, 5, 0, 3, 0, 0, 0, 0, 2, 6, 2, 2, 2, 0])
```

In [15]:

```
df["New Cluster"]=y_predicted
df.head()
```

Out[15]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothnes
0	842302	M	0.521037	0.022658	122.80	1001.0	
1	842517	M	0.643144	0.272574	132.90	1326.0	
2	84300903	M	0.601496	0.390260	130.00	1203.0	
3	84348301	M	0.210090	0.360839	77.58	386.1	
4	84358402	M	0.629893	0.156578	135.10	1297.0	

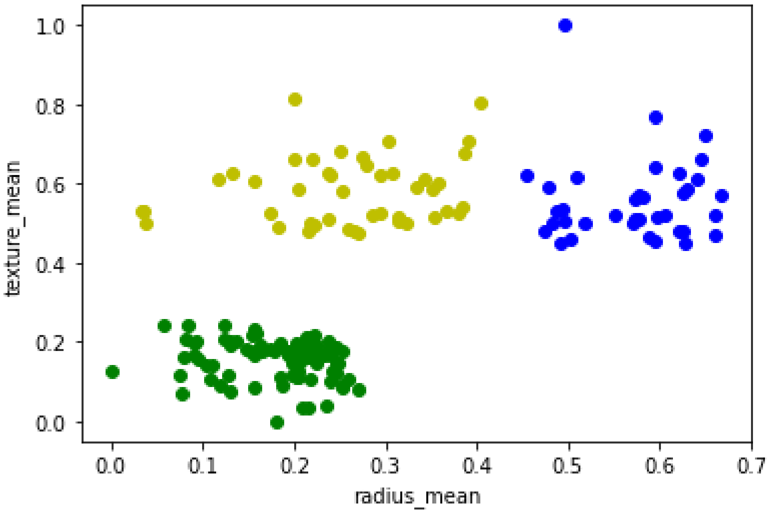
5 rows × 35 columns

In [16]:

```
df1=df[df["New Cluster"]==0]
df2=df[df["New Cluster"]==1]
df3=df[df["New Cluster"]==2]
plt.scatter(df1["radius_mean"],df1["texture_mean"],color="y")
plt.scatter(df2["radius_mean"],df2["texture_mean"],color="green")
plt.scatter(df3["radius_mean"],df3["texture_mean"],color="blue")
plt.xlabel("radius_mean")
plt.ylabel("texture_mean")
```

Out[16]:

```
Text(0, 0.5, 'texture_mean')
```



In [17]:

```
km.cluster_centers_
```

Out[17]:

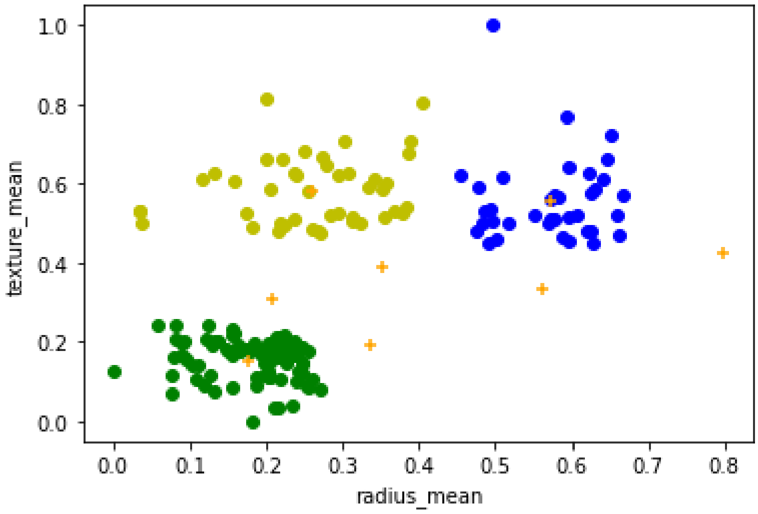
```
array([[0.2590623 , 0.58293879],
       [0.17750575, 0.15412045],
       [0.57132058, 0.55893025],
       [0.35173159, 0.39188367],
       [0.56287997, 0.33184226],
       [0.20867092, 0.3094643 ],
       [0.79840767, 0.42469846],
       [0.33570532, 0.19063107]])
```

In [18]:

```
df1=df[df["New Cluster"]==0]
df2=df[df["New Cluster"]==1]
df3=df[df["New Cluster"]==2]
plt.scatter(df1["radius_mean"],df1["texture_mean"],color="y")
plt.scatter(df2["radius_mean"],df2["texture_mean"],color="green")
plt.scatter(df3["radius_mean"],df3["texture_mean"],color="blue")
plt.scatter(km.cluster_centers_[:,0],km.cluster_centers_[:,1],color="orange",marker="+")
plt.xlabel("radius_mean")
plt.ylabel("texture_mean")
```

Out[18]:

```
Text(0, 0.5, 'texture_mean')
```



In [19]:

```
k_rng=range(1,10)
sse=[]
```

In [20]:

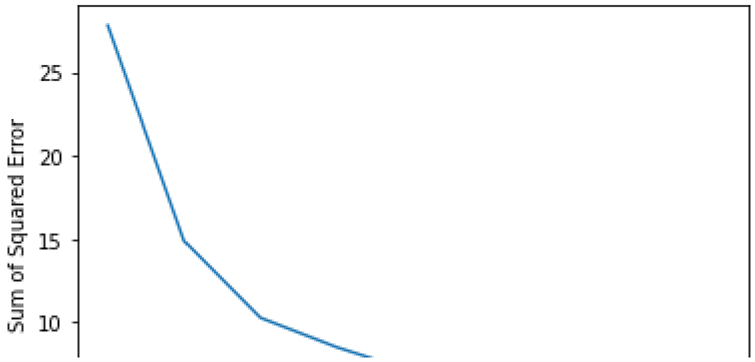
```
for k in k_rng:
 km=KMeans(n_clusters=k)
 km.fit(df[["radius_mean","texture_mean"]])
 sse.append(km.inertia_)#km.inertia_ will gives  the value of sum of square error
print(sse)
plt.plot(k_rng,sse)
plt.xlabel("K")
plt.ylabel("Sum of Squared Error")
```

```
  warnings.warn(

[27.817507595043075, 14.872296449956036, 10.252751496105198, 8.48472527
7027607, 7.030773895811419, 6.043115625877609, 5.143113512343714, 4.443
01570025843, 4.018064982712662]
```

Out[20]:

```
Text(0, 0.5, 'Sum of Squared Error')
```



CONCLUSION

we can use multiple models but we get different types of accuracies so,that's why we will take it as a clustering and done with K-Means Clustering