Problem Statement: To Predict How Best the Datafits and To predict the BreastCancer based on the given Features

KMeans Clustering

1)Data collection

In [1]:
```
#Importing Libraries
import pandas as pd
from matplotlib import pyplot as plt
%matplotlib inline
```

In [2]:
```
df=pd.read_csv(r"C:\Users\Jayadeep\Downloads\BreastCancerPrediction.csv")
df
```

Out[2]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	poin
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	
...	...	...	...	...	...	...	...	...	...	
564	926424	M	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	
565	926682	M	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	
566	926954	M	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	
567	927241	M	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	
568	92751	B	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	

569 rows × 33 columns

2)Data cleaning and preprocessing

In [3]:
```
df.head()
```

Out[3]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	co points_
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0

5 rows × 33 columns

In [4]:
```
df.tail()
```

Out[4]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	co points_
564	926424	M	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0
565	926682	M	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0
566	926954	M	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0
567	927241	M	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0
568	92751	B	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0

5 rows × 33 columns

In [5]:
```
df.shape
```

Out[5]:
```
(569, 33)
```

In [6]:
```
df.describe
```

Out[6]:
```
<bound method NDFrame.describe of            id diagnosis  radius_mean  texture_mean  perimeter_mean  area_mean  \
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

     smoothness_mean  compactness_mean  concavity_mean  concave points_mean  \
0            0.11840           0.27760         0.30010              0.14710
1            0.08474           0.07864         0.08690              0.07017
2            0.10960           0.15990         0.19740              0.12790
3            0.14250           0.28390         0.24140              0.10520
4            0.10030           0.13280         0.19800              0.10430
..               ...               ...             ...                  ...
564          0.11100           0.11590         0.24390              0.13890
565          0.09780           0.10340         0.14400              0.09791
566          0.08455           0.10230         0.09251              0.05302
567          0.11780           0.27700         0.35140              0.15200
568          0.05263           0.04362         0.00000              0.00000

     ...  texture_worst  perimeter_worst  area_worst  smoothness_worst  \
0    ...          17.33           184.60      2019.0           0.16220
1    ...          23.41           158.80      1956.0           0.12380
2    ...          25.53           152.50      1709.0           0.14440
3    ...          26.50            98.87       567.7           0.20980
4    ...          16.67           152.20      1575.0           0.13740
..   ...            ...              ...         ...               ...
564  ...          26.40           166.10      2027.0           0.14100
565  ...          38.25           155.00      1731.0           0.11660
566  ...          34.12           126.70      1124.0           0.11390
567  ...          39.42           184.60      1821.0           0.16500
568  ...          30.37            59.16       268.6           0.08996

     compactness_worst  concavity_worst  concave points_worst  symmetry_worst  \
0              0.66560           0.7119                0.2654          0.4601
```

```
1              0.18660          0.2416                0.1860          0.2750
2              0.42450          0.4504                0.2430          0.3613
3              0.86630          0.6869                0.2575          0.6638
4              0.20500          0.4000                0.1625          0.2364
..                 ...             ...                   ...             ...
564            0.21130          0.4107                0.2216          0.2060
565            0.19220          0.3215                0.1628          0.2572
566            0.30940          0.3403                0.1418          0.2218
567            0.86810          0.9387                0.2650          0.4087
568            0.06444          0.0000                0.0000          0.2871

     fractal_dimension_worst  Unnamed: 32
0                    0.11890          NaN
1                    0.08902          NaN
2                    0.08758          NaN
3                    0.17300          NaN
4                    0.07678          NaN
..                       ...          ...
564                  0.07115          NaN
565                  0.06637          NaN
566                  0.07820          NaN
567                  0.12400          NaN
568                  0.07039          NaN

[569 rows x 33 columns]>
```

In [7]:
```
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 [8]:
```
df.isnull().sum()
```

Out[8]:
```
id                          0
diagnosis                   0
radius_mean                 0
texture_mean                0
perimeter_mean              0
area_mean                   0
smoothness_mean             0
compactness_mean            0
concavity_mean              0
concave points_mean         0
symmetry_mean               0
fractal_dimension_mean      0
radius_se                   0
texture_se                  0
perimeter_se                0
area_se                     0
smoothness_se               0
compactness_se              0
concavity_se                0
concave points_se           0
symmetry_se                 0
fractal_dimension_se        0
radius_worst                0
texture_worst               0
perimeter_worst             0
area_worst                  0
smoothness_worst            0
compactness_worst           0
concavity_worst             0
concave points_worst        0
symmetry_worst              0
fractal_dimension_worst     0
Unnamed: 32               569
dtype: int64
```

In [9]:
```
df.drop(['Unnamed: 32'],axis=1)
```

Out[9]:

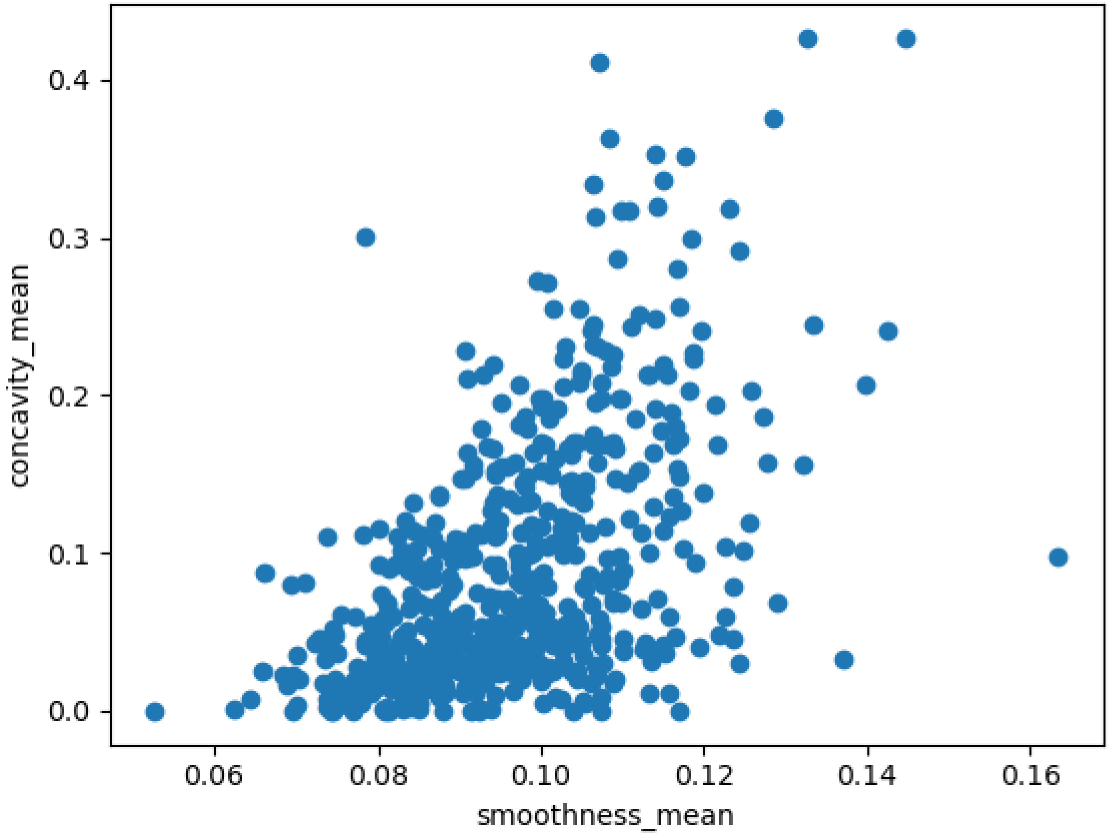
	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	poin
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	
...	...	...	...	...	...	...	...	...	...	
564	926424	M	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	
565	926682	M	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	
566	926954	M	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	
567	927241	M	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	
568	92751	B	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	

569 rows × 32 columns

3)Exploratory data Analysis

In [10]:
```
plt.scatter(df["smoothness_mean"],df["concavity_mean"])
plt.xlabel("smoothness_mean")
plt.ylabel("concavity_mean")
```

Out[10]:
```
Text(0, 0.5, 'concavity_mean')
```



4)Training our model

In [11]:
```
from sklearn.cluster import KMeans
```

In [12]:
```
km=KMeans()
km
```

Out[12]:
```
KMeans()
```

In [13]:
```
y_predicted=km.fit_predict(df[["smoothness_mean","concavity_mean"]])
y_predicted
```

Out[13]:
```
array([3, 0, 1, 1, 1, 5, 0, 0, 1, 1, 4, 0, 1, 0, 1, 5, 7, 5, 5, 7, 4, 4,
       1, 0, 5, 1, 5, 5, 5, 0, 1, 0, 1, 5, 5, 5, 5, 4, 4, 0, 4, 0, 1, 0,
       7, 1, 2, 0, 7, 4, 2, 2, 2, 5, 7, 4, 5, 5, 2, 4, 2, 4, 1, 7, 0, 5,
       4, 4, 3, 4, 0, 0, 5, 7, 4, 0, 7, 5, 6, 4, 4, 0, 3, 1, 4, 5, 0, 5,
       7, 0, 4, 0, 4, 4, 5, 5, 4, 2, 4, 0, 0, 2, 2, 7, 4, 1, 7, 4, 6, 7,
       4, 0, 3, 7, 4, 4, 0, 5, 1, 7, 4, 5, 6, 0, 7, 2, 7, 0, 0, 1, 4, 5,
       0, 7, 0, 4, 4, 4, 5, 4, 2, 0, 4, 4, 2, 4, 5, 0, 0, 4, 4, 5, 6, 4,
       0, 4, 1, 7, 2, 2, 7, 0, 1, 7, 5, 2, 4, 7, 1, 4, 4, 7, 1, 4, 2, 2,
       5, 5, 2, 2, 3, 1, 7, 7, 7, 2, 7, 4, 4, 4, 1, 4, 2, 0, 5, 4, 5, 0,
       0, 0, 4, 0, 6, 5, 7, 7, 4, 7, 0, 7, 5, 4, 3, 5, 0, 0, 7, 7, 0, 0,
       4, 7, 4, 0, 4, 7, 2, 7, 7, 5, 1, 2, 2, 5, 2, 2, 1, 0, 0, 5, 4, 2,
       5, 4, 1, 4, 4, 5, 4, 4, 3, 4, 1, 0, 5, 0, 5, 1, 3, 5, 5, 4, 0, 4,
       0, 5, 7, 4, 4, 0, 2, 4, 1, 2, 7, 7, 2, 7, 2, 4, 1, 2, 5, 1, 0, 2,
       7, 2, 0, 2, 5, 7, 7, 4, 2, 2, 2, 4, 2, 4, 1, 7, 1, 4, 4, 2, 2, 2,
       2, 2, 2, 2, 4, 2, 2, 2, 2, 0, 1, 2, 7, 0, 4, 1, 2, 4, 2, 2, 5, 5,
       0, 7, 2, 2, 2, 5, 4, 0, 4, 1, 7, 7, 7, 1, 4, 4, 2, 4, 4, 4, 2, 3,
       3, 0, 4, 0, 0, 2, 4, 4, 2, 4, 4, 7, 2, 0, 5, 4, 0, 1, 5, 4, 1, 5,
       2, 7, 1, 2, 4, 5, 7, 4, 7, 7, 7, 7, 7, 2, 0, 1, 4, 2, 1, 3, 4, 2,
       0, 7, 2, 4, 3, 4, 4, 4, 2, 4, 7, 7, 0, 4, 4, 4, 4, 7, 4, 4, 4, 5,
       2, 2, 7, 5, 7, 0, 2, 2, 7, 4, 2, 2, 3, 7, 1, 5, 4, 0, 2, 4, 2, 2,
       0, 0, 2, 2, 0, 7, 5, 4, 0, 5, 0, 5, 4, 7, 4, 4, 7, 4, 2, 2, 0, 6,
       4, 4, 4, 0, 0, 2, 1, 0, 4, 2, 7, 2, 7, 7, 7, 2, 7, 1, 4, 4, 7, 4,
       0, 5, 7, 1, 4, 4, 2, 4, 0, 2, 2, 7, 7, 4, 5, 1, 7, 5, 7, 5, 0, 0,
       7, 4, 7, 1, 7, 4, 5, 7, 7, 4, 5, 5, 4, 4, 4, 1, 2, 7, 4, 4, 4, 4,
       0, 4, 7, 4, 2, 5, 7, 1, 5, 4, 2, 0, 7, 0, 4, 4, 4, 4, 2, 4, 2, 2,
       2, 4, 2, 4, 7, 7, 2, 2, 0, 0, 4, 2, 1, 3, 1, 5, 0, 6, 2])
```

In [14]:
```
df["cluster"]=y_predicted
```

In [15]:
```
df.head()
```

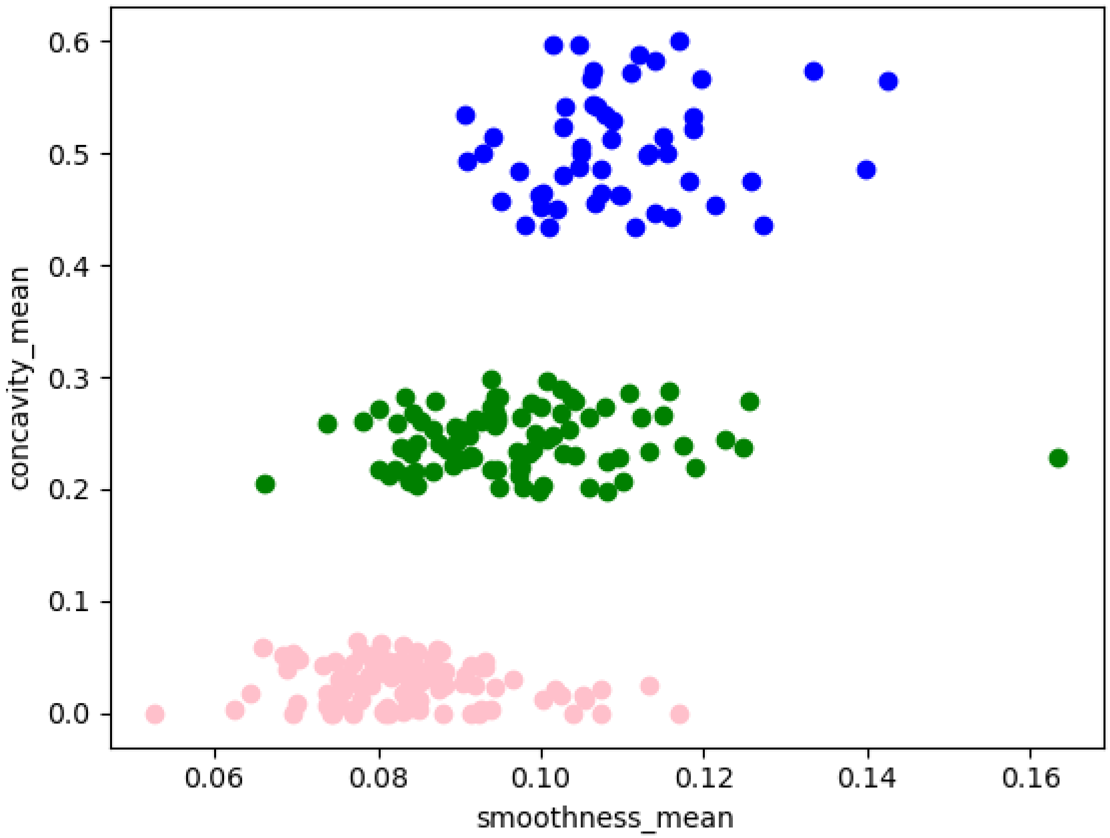
Out[15]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	co points_
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0

5 rows × 34 columns

In [23]:
```
df1=df[df.cluster==0]
df2=df[df.cluster==1]
df3=df[df.cluster==2]
plt.scatter(df1["smoothness_mean"],df1["concavity_mean"],color='green')
plt.scatter(df2["smoothness_mean"],df2["concavity_mean"],color='blue')
plt.scatter(df3["smoothness_mean"],df3["concavity_mean"],color='pink')
plt.xlabel("smoothness_mean")
plt.ylabel("concavity_mean")
```

Out[23]:
```
Text(0, 0.5, 'concavity_mean')
```



In [20]:
```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
scaler.fit(df[["concavity_mean"]])
df["concavity_mean"]=scaler.transform(df[["concavity_mean"]])
df.head()
```

Out[20]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	co points_
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.703140	0
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.203608	0
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.462512	0
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.565604	0
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.463918	0

5 rows × 34 columns

In [24]:
```
scaler=MinMaxScaler()
scaler.fit(df[["smoothness_mean"]])
df["smoothness_mean"]=scaler.transform(df[["smoothness_mean"]])
df.head()
```

Out[24]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	co points_
0	842302	M	17.99	10.38	122.80	1001.0	0.593753	0.27760	0.703140	0
1	842517	M	20.57	17.77	132.90	1326.0	0.289880	0.07864	0.203608	0
2	84300903	M	19.69	21.25	130.00	1203.0	0.514309	0.15990	0.462512	0
3	84348301	M	11.42	20.38	77.58	386.1	0.811321	0.28390	0.565604	0
4	84358402	M	20.29	14.34	135.10	1297.0	0.430351	0.13280	0.463918	0

5 rows × 34 columns

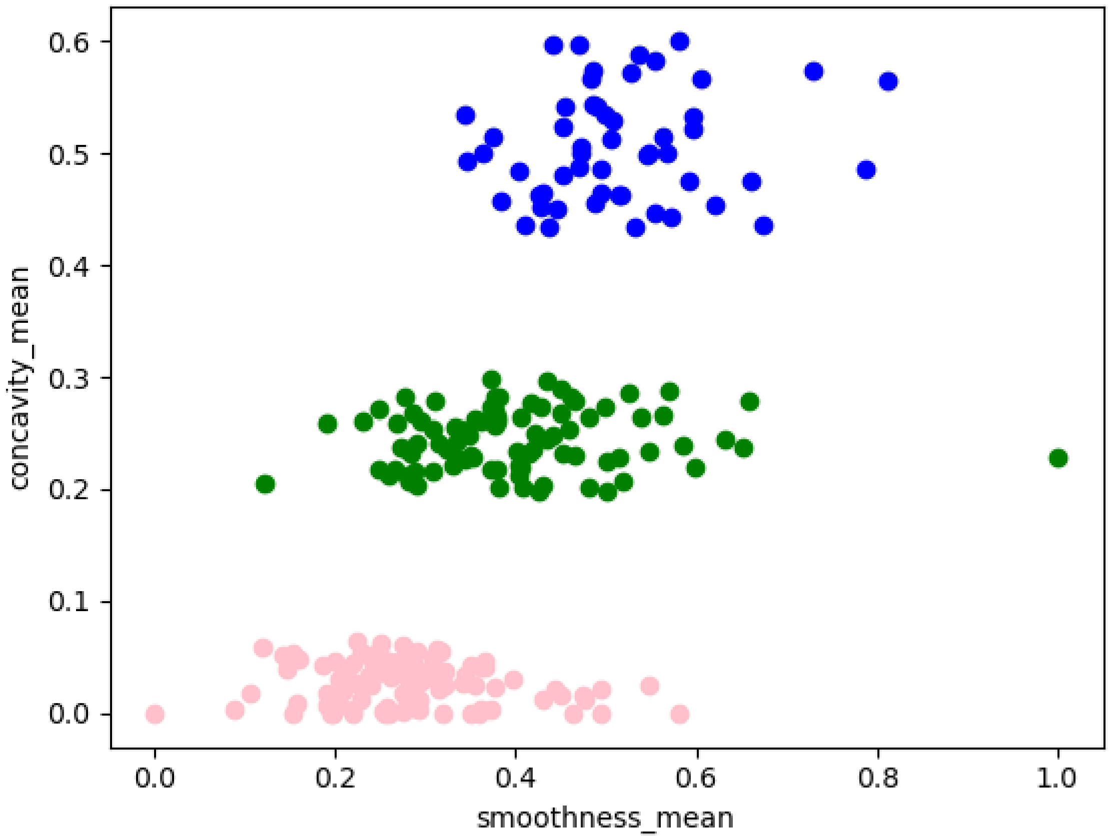
In [25]:
```
km=KMeans()
```

In [26]:
```
y_predicted=km.fit_predict(df[["smoothness_mean","concavity_mean"]])
y_predicted
```

Out[26]:
```
array([2, 3, 7, 5, 7, 5, 3, 4, 5, 7, 6, 3, 7, 3, 7, 5, 3, 5, 0, 3, 4, 4,
       7, 3, 5, 7, 0, 0, 0, 3, 7, 0, 7, 0, 0, 0, 0, 1, 1, 0, 6, 5, 7, 0,
       3, 7, 1, 5, 4, 1, 1, 6, 6, 5, 1, 1, 0, 5, 6, 1, 4, 4, 7, 6, 0, 5,
       4, 6, 2, 1, 3, 3, 0, 4, 1, 3, 4, 0, 2, 4, 4, 0, 2, 5, 1, 0, 3, 0,
       1, 4, 1, 3, 6, 4, 0, 0, 4, 4, 1, 3, 3, 4, 6, 4, 4, 5, 4, 1, 2, 1,
       4, 0, 7, 4, 4, 1, 3, 5, 7, 3, 1, 0, 2, 4, 6, 1, 1, 3, 5, 7, 4, 0,
       0, 3, 3, 1, 4, 1, 5, 4, 1, 3, 4, 1, 6, 4, 0, 3, 3, 6, 4, 0, 2, 1,
       3, 1, 7, 6, 1, 6, 4, 3, 7, 4, 3, 6, 1, 3, 7, 1, 4, 1, 5, 4, 1, 1,
       0, 0, 6, 1, 2, 7, 3, 1, 1, 1, 3, 1, 4, 6, 7, 1, 6, 0, 0, 6, 5, 3,
       3, 0, 1, 3, 2, 5, 3, 3, 4, 3, 3, 6, 0, 1, 2, 0, 3, 0, 3, 6, 0, 3,
       1, 4, 4, 0, 1, 4, 4, 3, 6, 0, 5, 6, 6, 0, 1, 1, 7, 3, 3, 0, 1, 6,
       0, 6, 7, 4, 6, 3, 1, 4, 7, 1, 7, 0, 0, 4, 0, 5, 2, 0, 0, 1, 3, 6,
       3, 0, 3, 6, 1, 4, 6, 1, 7, 1, 1, 4, 1, 3, 6, 1, 7, 6, 0, 7, 3, 6,
       3, 6, 3, 1, 3, 1, 4, 6, 1, 1, 1, 1, 6, 4, 7, 3, 7, 4, 1, 6, 1, 6,
       6, 6, 1, 6, 1, 1, 4, 1, 6, 0, 7, 6, 4, 3, 4, 7, 1, 4, 1, 6, 5, 5,
       0, 3, 4, 6, 6, 5, 1, 3, 4, 7, 3, 3, 4, 0, 4, 1, 6, 1, 1, 4, 6, 2,
       2, 0, 6, 3, 4, 1, 6, 4, 6, 1, 1, 1, 6, 3, 0, 1, 3, 7, 0, 6, 7, 0,
       6, 4, 7, 6, 6, 5, 4, 6, 6, 4, 1, 3, 3, 6, 3, 7, 1, 4, 5, 2, 4, 6,
       4, 3, 6, 1, 2, 1, 6, 1, 1, 4, 1, 6, 0, 1, 1, 4, 6, 3, 6, 1, 4, 5,
       1, 4, 1, 0, 4, 3, 4, 6, 4, 1, 6, 6, 7, 4, 7, 0, 1, 0, 1, 1, 1, 6,
       3, 3, 1, 6, 3, 4, 0, 1, 3, 0, 6, 0, 1, 4, 1, 1, 3, 1, 6, 6, 3, 2,
       6, 1, 6, 3, 3, 6, 7, 5, 1, 1, 6, 6, 4, 1, 1, 6, 4, 7, 1, 6, 4, 4,
       0, 0, 1, 7, 4, 6, 6, 6, 0, 6, 6, 3, 4, 1, 0, 7, 3, 5, 4, 0, 5, 5,
       4, 4, 1, 5, 3, 1, 0, 3, 3, 4, 0, 0, 4, 4, 4, 7, 1, 4, 1, 4, 4, 1,
       5, 4, 4, 4, 1, 0, 1, 7, 0, 4, 6, 3, 4, 3, 6, 1, 1, 1, 1, 1, 1, 6,
       6, 1, 6, 1, 6, 1, 1, 6, 3, 3, 4, 6, 7, 2, 7, 0, 3, 2, 6])
```

In [27]:
```
df1=df[df.cluster==0]
df2=df[df.cluster==1]
df3=df[df.cluster==2]
plt.scatter(df1["smoothness_mean"],df1["concavity_mean"],color='green')
plt.scatter(df2["smoothness_mean"],df2["concavity_mean"],color='blue')
plt.scatter(df3["smoothness_mean"],df3["concavity_mean"],color='pink')
plt.xlabel("smoothness_mean")
plt.ylabel("concavity_mean")
```

Out[27]:
```
Text(0, 0.5, 'concavity_mean')
```

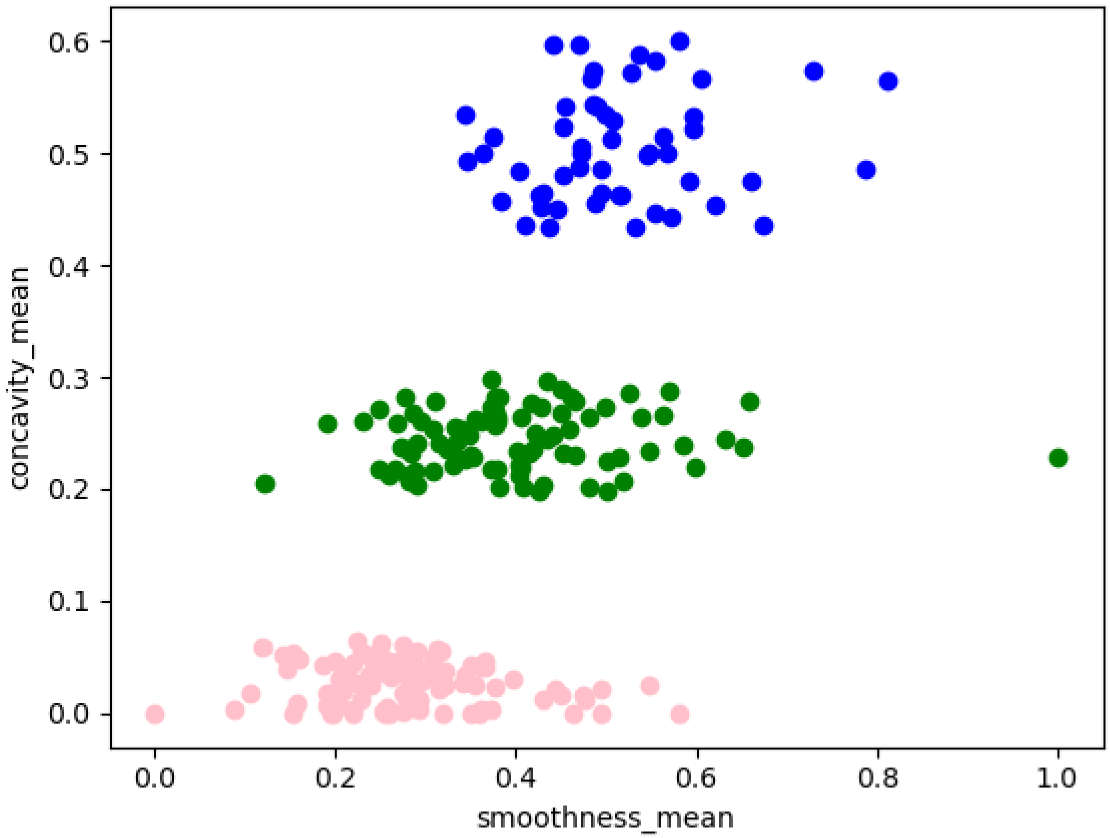


In [28]:
```
km.cluster_centers_
```

Out[28]:
```
array([[0.43398859, 0.33623868],
       [0.34564593, 0.07266997],
       [0.5821171 , 0.79713371],
       [0.33571948, 0.21763548],
       [0.4973293 , 0.10747708],
       [0.62466591, 0.37283803],
       [0.22278576, 0.05977613],
       [0.47534256, 0.52183591]])
```

In [29]:
```
df1=df[df.cluster==0]
df2=df[df.cluster==1]
df3=df[df.cluster==2]
plt.scatter(df1["smoothness_mean"],df1["concavity_mean"],color='green')
plt.scatter(df2["smoothness_mean"],df2["concavity_mean"],color='blue')
plt.scatter(df3["smoothness_mean"],df3["concavity_mean"],color='pink')
plt.xlabel("smoothness_mean")
plt.ylabel("concavity_mean")
```

Out[29]:
```
Text(0, 0.5, 'concavity_mean')
```



In [30]:
```
k_rng=range(1,10)
sse=[]
```

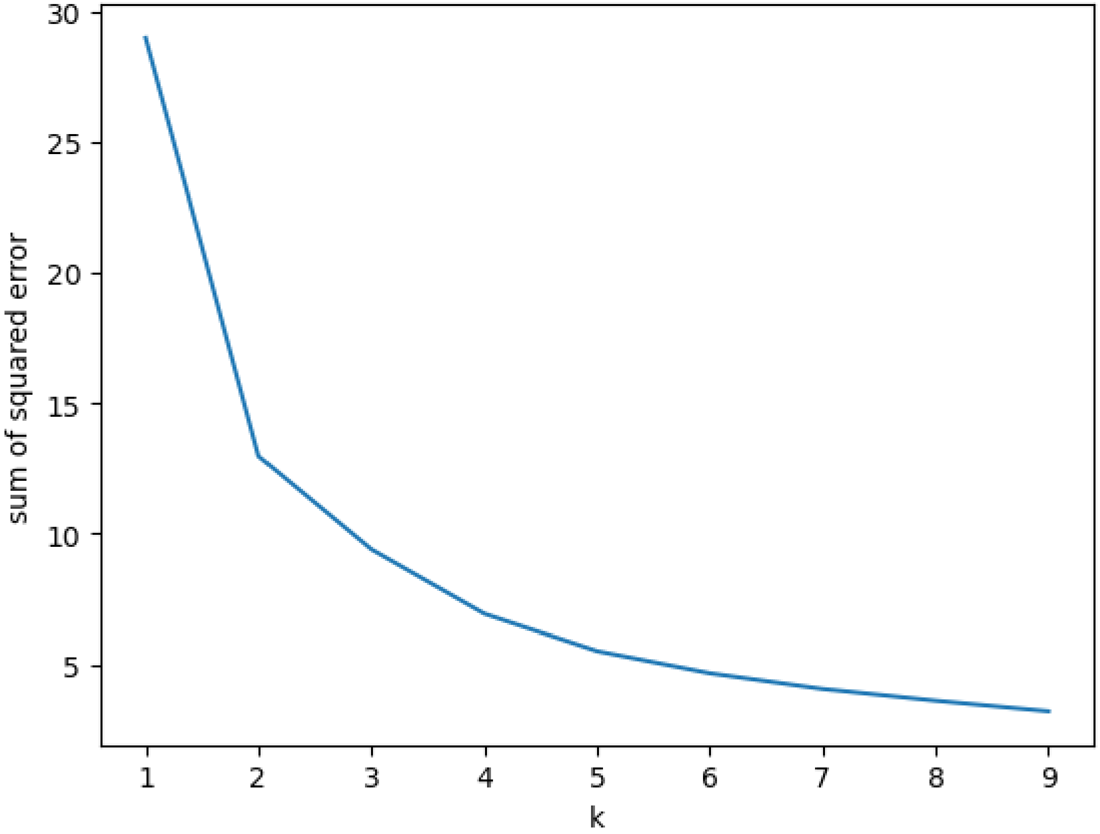
In [32]:
```
for k in k_rng:
    km=KMeans(n_clusters=k)
    km.fit(df[["smoothness_mean","concavity_mean"]])
    sse.append(km.inertia_)
sse
```

```
C:\Users\Jayadeep\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1036: UserWarning: KMeans is known to have
a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting th
e environment variable OMP_NUM_THREADS=3.
  warnings.warn(
```

Out[32]:
```
[28.97323405387539,
 12.96263525857166,
 9.392190505398347,
 6.936972751219077,
 5.506884287188131,
 4.665441269342259,
 4.065031707454045,
 3.614839704521038,
 3.210208570550766]
```

In [33]:
```
plt.plot(k_rng,sse)
plt.xlabel("k")
plt.ylabel("sum of squared error")
```

Out[33]:
```
Text(0, 0.5, 'sum of squared error')
```



Conclusion: The given data set is "BreastCancer Prediction".For this Dataset we have used Kmeans model.Based on the given data set we have divided Dataset in to Different Clusters.