Breast cancer prediction

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

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

Out[2]:

	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

Data preprocessing

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

Out[3]:
```
<bound method NDFrame.head of            id diagnosis  radius_mean  texture_m
ean  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 [4]:
```
df.tail()
```

Out[4]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_m
564	926424	M	21.56	22.39	142.00	1479.0	0.1
565	926682	M	20.13	28.25	131.20	1261.0	0.0
566	926954	M	16.60	28.08	108.30	858.1	0.0
567	927241	M	20.60	29.33	140.10	1265.0	0.1
568	92751	B	7.76	24.54	47.92	181.0	0.0

5 rows × 33 columns

In [5]:
```
df.drop(['Unnamed: 32'],axis=1)
```

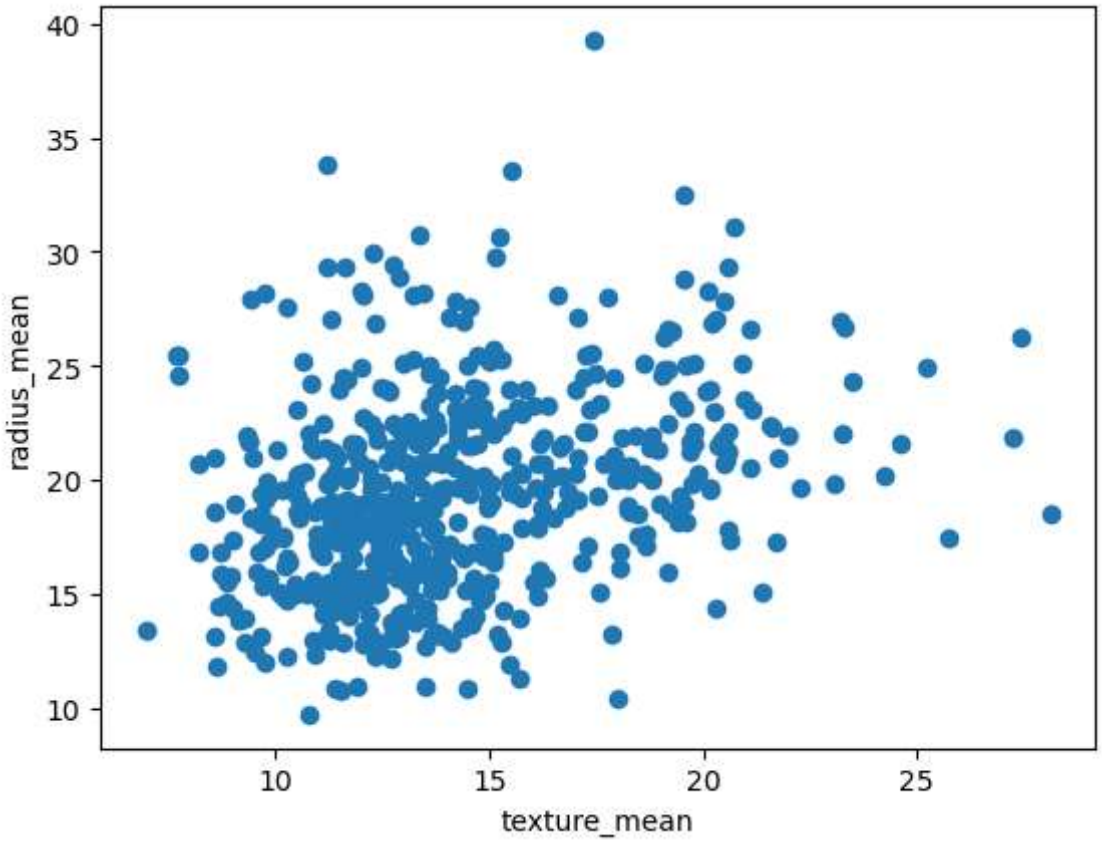
Out[5]:

is	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mea
M	17.99	10.38	122.80	1001.0	0.11840	0.2776
M	20.57	17.77	132.90	1326.0	0.08474	0.0786
M	19.69	21.25	130.00	1203.0	0.10960	0.1599
M	11.42	20.38	77.58	386.1	0.14250	0.2839
M	20.29	14.34	135.10	1297.0	0.10030	0.1328
...	...	...	...	...	...	.
M	21.56	22.39	142.00	1479.0	0.11100	0.1159
M	20.13	28.25	131.20	1261.0	0.09780	0.1034
M	16.60	28.08	108.30	858.1	0.08455	0.1023
M	20.60	29.33	140.10	1265.0	0.11780	0.2770
B	7.76	24.54	47.92	181.0	0.05263	0.0436

Data visualization

In [6]:
```
plt.scatter(df["radius_mean"],df["texture_mean"])
plt.xlabel("texture_mean")
plt.ylabel("radius_mean")
```

Out[6]:
```
Text(0, 0.5, 'radius_mean')
```



In [7]:
```
from sklearn.cluster import KMeans
km=KMeans()
km
```

Out[7]:
```
KMeans
KMeans()
```

In [8]:
```
y_predicted=km.fit_predict(df[["texture_mean","radius_mean"]])
y_predicted
```

```
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:87
0: FutureWarning: The default value of `n_init` will change from 10 to 'auto'
in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:138
2: UserWarning: KMeans is known to have a memory leak on Windows with MKL, wh
en there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
```

Out[8]:
```
array([4, 6, 6, 1, 6, 4, 0, 5, 5, 5, 5, 0, 7, 5, 5, 2, 0, 0, 6, 4, 4, 3,
       4, 6, 0, 4, 5, 0, 5, 4, 7, 1, 7, 7, 0, 0, 5, 1, 5, 5, 5, 5, 7, 1,
       5, 0, 1, 1, 3, 5, 5, 4, 1, 0, 5, 1, 6, 5, 1, 3, 3, 1, 5, 3, 5, 5,
       1, 1, 1, 4, 6, 3, 7, 4, 1, 0, 3, 0, 7, 1, 1, 4, 7, 7, 3, 0, 5, 7,
       5, 4, 5, 5, 4, 1, 0, 7, 1, 1, 3, 0, 5, 3, 1, 1, 1, 4, 1, 1, 6, 5,
       1, 5, 0, 1, 3, 5, 3, 4, 5, 0, 3, 0, 6, 4, 4, 4, 5, 0, 4, 7, 3, 0,
       0, 4, 0, 5, 1, 3, 4, 3, 3, 0, 1, 4, 3, 3, 1, 0, 4, 1, 5, 1, 3, 3,
       4, 1, 0, 0, 3, 3, 1, 0, 6, 5, 6, 0, 3, 0, 7, 4, 3, 1, 4, 3, 3, 3,
       1, 0, 5, 3, 6, 7, 0, 3, 5, 3, 0, 1, 1, 4, 5, 5, 1, 2, 5, 4, 5, 0,
       6, 0, 1, 0, 7, 5, 1, 4, 1, 0, 5, 4, 6, 1, 6, 7, 5, 4, 1, 1, 6, 7,
       4, 4, 1, 0, 4, 4, 3, 4, 5, 5, 0, 2, 2, 7, 3, 5, 7, 6, 2, 2, 4, 3,
       1, 5, 7, 1, 1, 4, 5, 3, 7, 1, 6, 0, 6, 4, 7, 4, 5, 2, 7, 0, 0, 0,
       0, 7, 1, 5, 4, 1, 4, 3, 6, 3, 7, 1, 3, 0, 1, 4, 7, 3, 6, 0, 4, 1,
       1, 3, 1, 1, 0, 0, 4, 1, 3, 4, 3, 1, 1, 5, 6, 1, 7, 1, 1, 5, 4, 3,
       4, 4, 1, 4, 3, 3, 1, 1, 3, 0, 1, 1, 3, 6, 3, 6, 3, 1, 4, 1, 0, 0,
       4, 1, 1, 3, 1, 0, 4, 0, 1, 7, 4, 1, 3, 6, 3, 3, 1, 4, 3, 3, 1, 0,
       6, 5, 3, 1, 1, 4, 3, 1, 1, 5, 1, 0, 4, 6, 7, 1, 6, 6, 5, 4, 6, 6,
       4, 4, 1, 2, 4, 1, 3, 3, 5, 1, 4, 5, 3, 4, 3, 7, 3, 1, 0, 6, 1, 4,
       1, 1, 3, 1, 0, 3, 1, 4, 3, 1, 4, 5, 0, 1, 1, 1, 1, 5, 2, 5, 1, 0,
       3, 5, 1, 4, 3, 1, 1, 1, 3, 5, 1, 1, 5, 1, 6, 6, 4, 1, 1, 4, 1, 4,
       1, 7, 4, 1, 0, 5, 7, 4, 0, 6, 5, 7, 2, 4, 1, 2, 2, 5, 5, 2, 7, 7,
       2, 1, 1, 1, 5, 1, 7, 1, 1, 2, 4, 2, 3, 4, 5, 4, 3, 0, 1, 1, 4, 1,
       4, 4, 4, 6, 3, 0, 5, 4, 0, 3, 5, 0, 1, 1, 0, 6, 4, 5, 4, 6, 3, 3,
       1, 1, 4, 5, 3, 4, 5, 4, 0, 1, 0, 6, 1, 4, 3, 6, 1, 1, 3, 3, 1, 3,
       4, 3, 1, 1, 4, 6, 1, 6, 5, 5, 5, 5, 3, 5, 5, 2, 5, 5, 3, 1, 1, 5,
       5, 5, 2, 5, 2, 2, 1, 2, 5, 5, 2, 2, 2, 7, 6, 7, 2, 7, 5])
```

In [9]:
```
df["cluster"]=y_predicted
df.head()
```

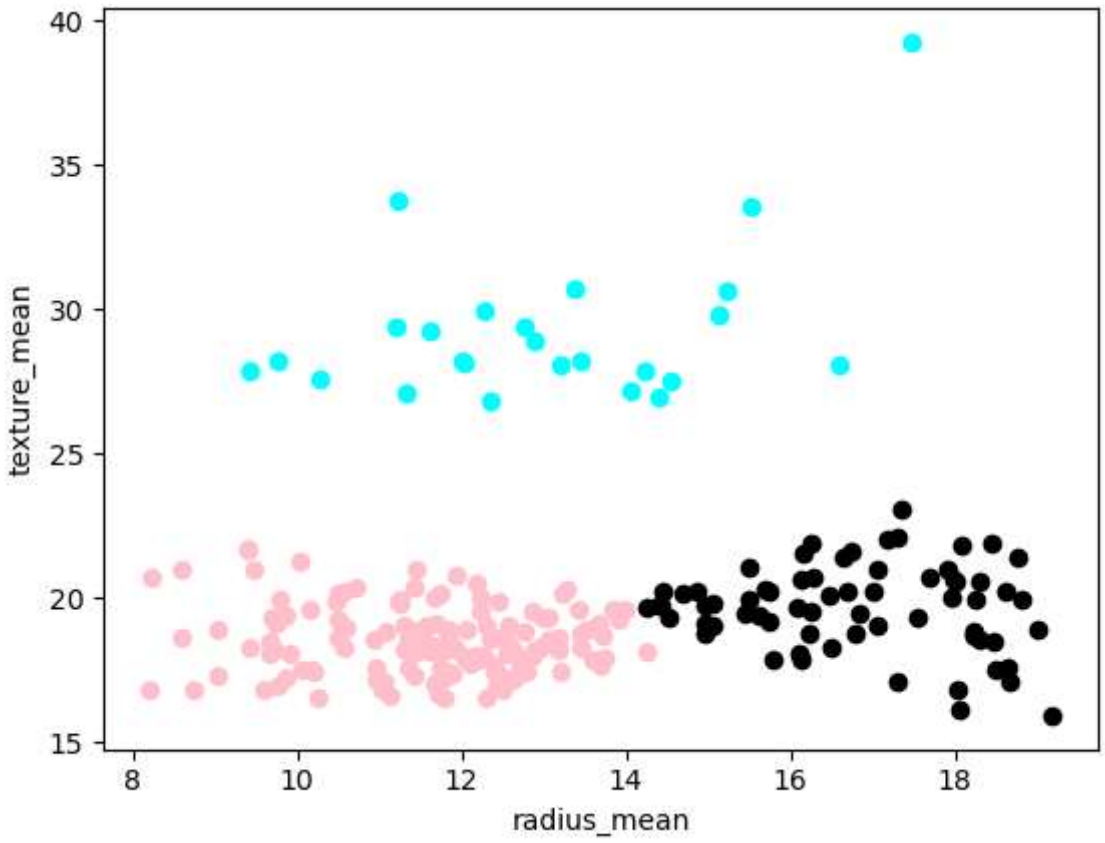
Out[9]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_m
0	842302	M	17.99	10.38	122.80	1001.0	0.1
1	842517	M	20.57	17.77	132.90	1326.0	0.08
2	84300903	M	19.69	21.25	130.00	1203.0	0.1(
3	84348301	M	11.42	20.38	77.58	386.1	0.14
4	84358402	M	20.29	14.34	135.10	1297.0	0.1(

5 rows × 34 columns

In [10]:
```
df1=df[df.cluster==0]
df2=df[df.cluster==1]
df3=df[df.cluster==2]
plt.scatter(df1["radius_mean"],df1["texture_mean"],color="black")
plt.scatter(df2["radius_mean"],df2["texture_mean"],color="pink")
plt.scatter(df3["radius_mean"],df3["texture_mean"],color="cyan")
plt.xlabel("radius_mean")
plt.ylabel("texture_mean")
```

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



In [11]:
```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
scaler.fit(df[["texture_mean"]])
df["texture_mean"]=scaler.transform(df[["texture_mean"]])
df.head()
```

Out[11]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_m
0	842302	M	17.99	0.022658	122.80	1001.0	0.1
1	842517	M	20.57	0.272574	132.90	1326.0	0.0
2	84300903	M	19.69	0.390260	130.00	1203.0	0.1
3	84348301	M	11.42	0.360839	77.58	386.1	0.1
4	84358402	M	20.29	0.156578	135.10	1297.0	0.1

5 rows × 34 columns

In [12]:
```
scaler.fit(df[["radius_mean"]])
df["radius_mean"]=scaler.transform(df[["radius_mean"]])
df.head()
```

Out[12]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_m
0	842302	M	0.521037	0.022658	122.80	1001.0	0.1
1	842517	M	0.643144	0.272574	132.90	1326.0	0.0
2	84300903	M	0.601496	0.390260	130.00	1203.0	0.1
3	84348301	M	0.210090	0.360839	77.58	386.1	0.1
4	84358402	M	0.629893	0.156578	135.10	1297.0	0.1

5 rows × 34 columns

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

```
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:87
0: FutureWarning: The default value of `n_init` will change from 10 to 'auto'
in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:138
2: UserWarning: KMeans is known to have a memory leak on Windows with MKL, wh
en there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
```

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

In [14]:
```
df["New Cluster"]=y_predicted
df.head()
```

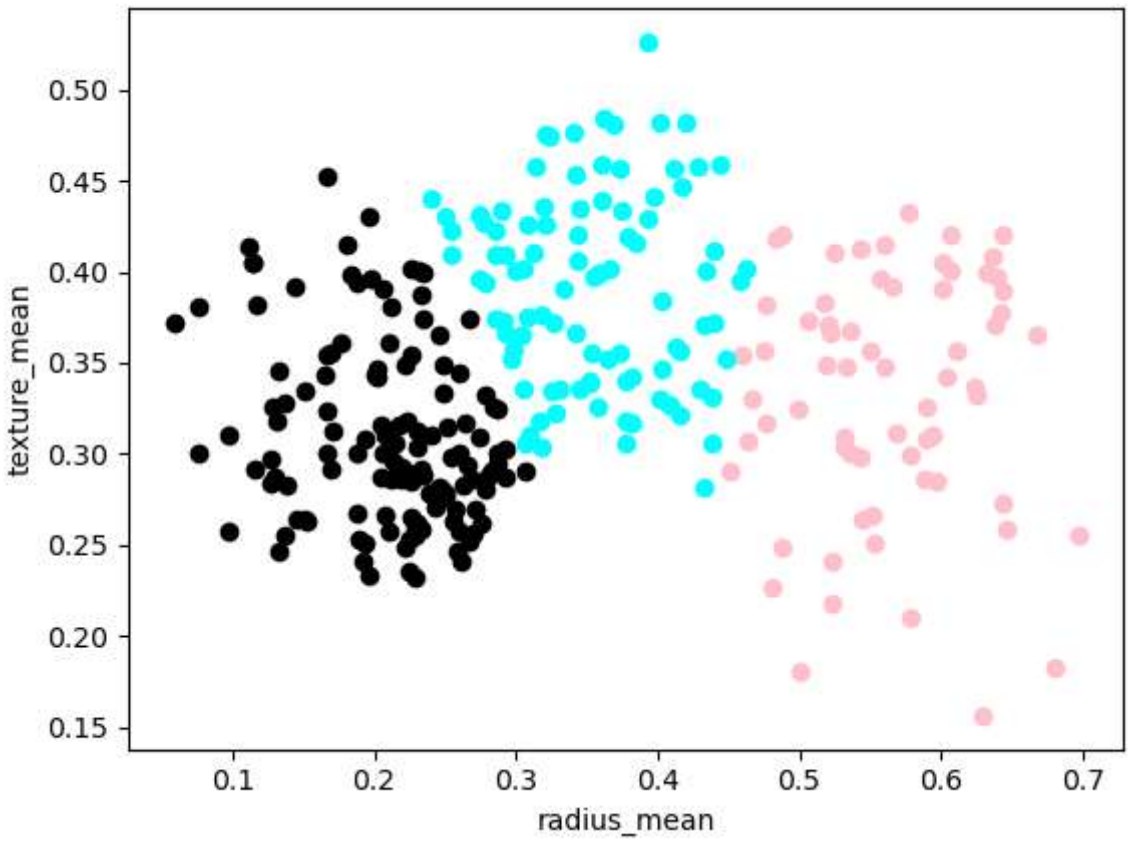
Out[14]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_m
0	842302	M	0.521037	0.022658	122.80	1001.0	0.1
1	842517	M	0.643144	0.272574	132.90	1326.0	0.08
2	84300903	M	0.601496	0.390260	130.00	1203.0	0.1(
3	84348301	M	0.210090	0.360839	77.58	386.1	0.14
4	84358402	M	0.629893	0.156578	135.10	1297.0	0.1(

5 rows × 35 columns

In [15]:
```
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="black")
plt.scatter(df2["radius_mean"],df2["texture_mean"],color="pink")
plt.scatter(df3["radius_mean"],df3["texture_mean"],color="cyan")
plt.xlabel("radius_mean")
plt.ylabel("texture_mean")
```

Out[15]:
```
Text(0, 0.5, 'texture_mean')
```

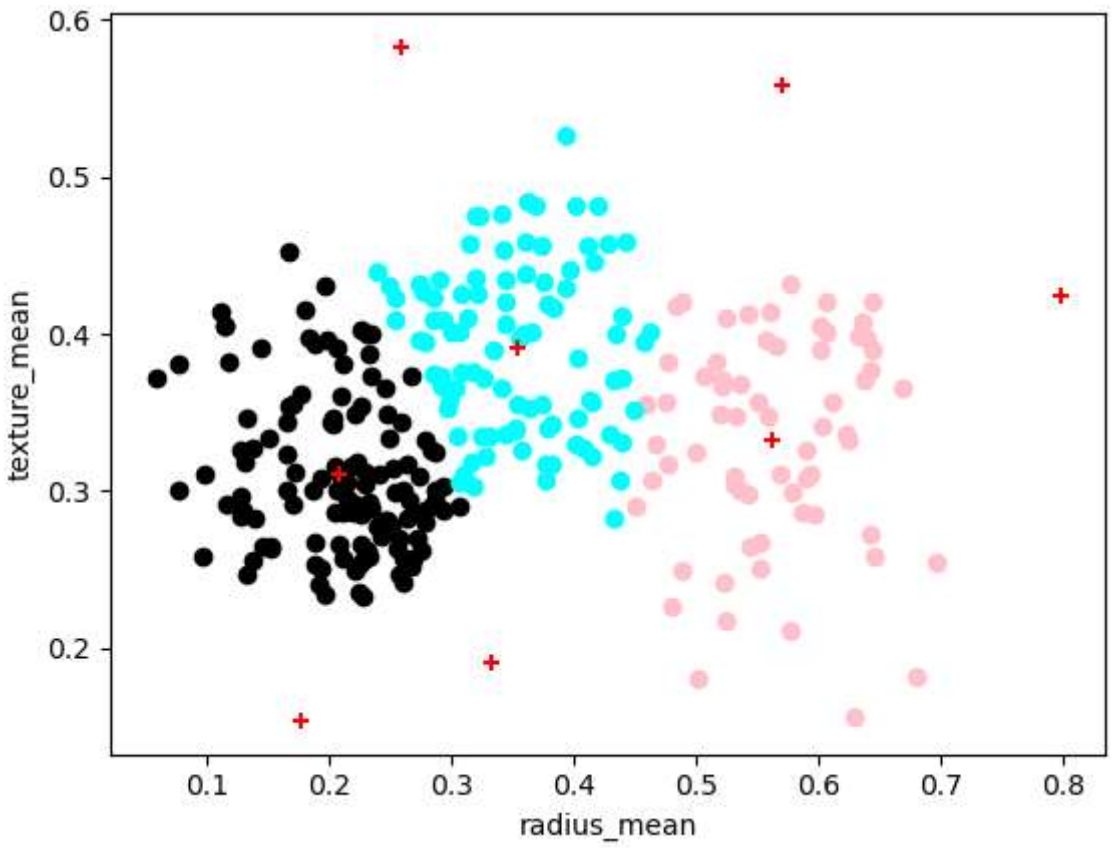


In [16]:
```
km.cluster_centers_
```

Out[16]:
```
array([[0.20878924, 0.31058452],
       [0.56287997, 0.33184226],
       [0.3534653 , 0.39091896],
       [0.3331624 , 0.18999839],
       [0.2590623 , 0.58293879],
       [0.17652977, 0.15382448],
       [0.79840767, 0.42469846],
       [0.57132058, 0.55893025]])
```

In [17]:
```
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="black")
plt.scatter(df2["radius_mean"],df2["texture_mean"],color="pink")
plt.scatter(df3["radius_mean"],df3["texture_mean"],color="cyan")
plt.scatter(km.cluster_centers_[:,0],km.cluster_centers_[:,1],color="red",mark
plt.xlabel("radius_mean")
plt.ylabel("texture_mean")
```

Out[17]:
```
Text(0, 0.5, 'texture_mean')
```



In [18]:
```
k_rng=range(1,10)
sse=[]
```

Elbow method

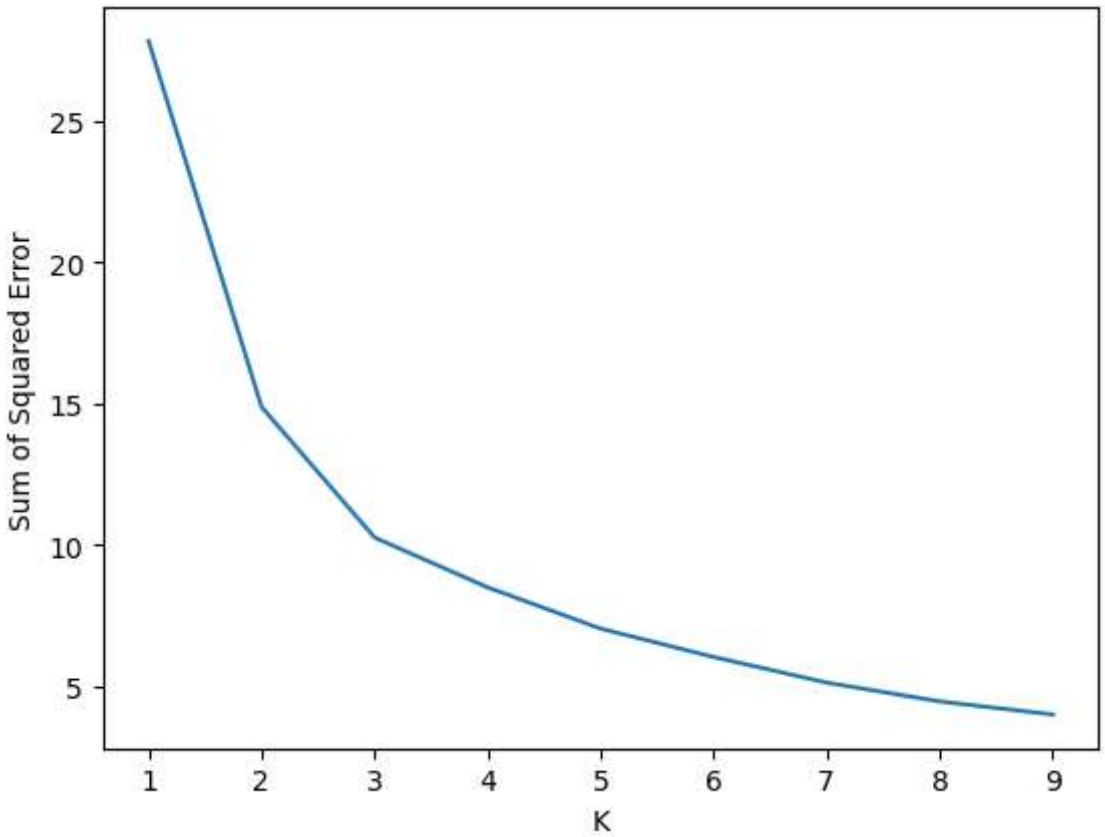
In [20]:
```
for k in k_rng:
    km=KMeans(n_clusters=k)
    km.fit(df[["radius_mean","texture_mean"]])
    sse.append(km.inertia_)
print(sse)
plt.plot(k_rng,sse)
plt.xlabel("K")
plt.ylabel("Sum of Squared Error")
```

```
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:87
0: FutureWarning: The default value of `n_init` will change from 10 to 'auto'
in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:138
2: UserWarning: KMeans is known to have a memory leak on Windows with MKL, wh
en there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:87
0: FutureWarning: The default value of `n_init` will change from 10 to 'auto'
in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:138
2: UserWarning: KMeans is known to have a memory leak on Windows with MKL, wh
en there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:87
0: FutureWarning: The default value of `n_init` will change from 10 to 'auto'
in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:138
2: UserWarning: KMeans is known to have a memory leak on Windows with MKL, wh
en there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:87
0: FutureWarning: The default value of `n_init` will change from 10 to 'auto'
in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:138
2: UserWarning: KMeans is known to have a memory leak on Windows with MKL, wh
en there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:87
0: FutureWarning: The default value of `n_init` will change from 10 to 'auto'
in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:138
2: UserWarning: KMeans is known to have a memory leak on Windows with MKL, wh
en there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:87
0: FutureWarning: The default value of `n_init` will change from 10 to 'auto'
in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:138
2: UserWarning: KMeans is known to have a memory leak on Windows with MKL, wh
en there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:87
0: FutureWarning: The default value of `n_init` will change from 10 to 'auto'
in 1.4. Set the value of `n_init` explicitly to suppress the warning
```

```
  warnings.warn(
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:138
2: UserWarning: KMeans is known to have a memory leak on Windows with MKL, wh
en there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:87
0: FutureWarning: The default value of `n_init` will change from 10 to 'auto'
in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:138
2: UserWarning: KMeans is known to have a memory leak on Windows with MKL, wh
en there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:87
0: FutureWarning: The default value of `n_init` will change from 10 to 'auto'
in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
C:\Users\Y.Saranya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:138
2: UserWarning: KMeans is known to have a memory leak on Windows with MKL, wh
en there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(

[27.817507595043075, 14.872296449956036, 10.252751496105198, 8.48893568609588
9, 7.030611949327338, 6.025284267841283, 5.118942965866768, 4.45468255928802
7, 3.991752934887513]
```

Out[20]: Text(0, 0.5, 'Sum of Squared Error')



CONCLUSION:- BASED ON THE ABOVE PROGRAM THE DATA HAS DIVIDED INTO SEVARAL THE DATA HAS DIVIDED INTO SEVARAL

In [ ]: