PROBLEM STATEMENT:- TO SEGREGATE DATA INTO SEVARAL CLUSTERS USING K-MEANS

In [26]:
```
import pandas as pd
from matplotlib import pyplot as plt
%matplotlib inline
```

LOADING THE DATASET

In [27]:
```
df=pd.read_csv(r"/content/BreastCancerPrediction.csv")
df
```

Out[27]:

	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
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 × 33 columns

DATA PREPROCESSING:-

In [28]:
```
df.head()
```

Out[28]:

	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
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 rows × 33 columns

In [29]:
```
df.tail()
```

Out[29]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean
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

5 rows × 33 columns

In [30]:
```
df.describe()
```

Out[30]:

	id	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	cc
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 × 32 columns

In [31]:
```
df.drop(['Unnamed: 32'],axis=1)
```

Out[31]:

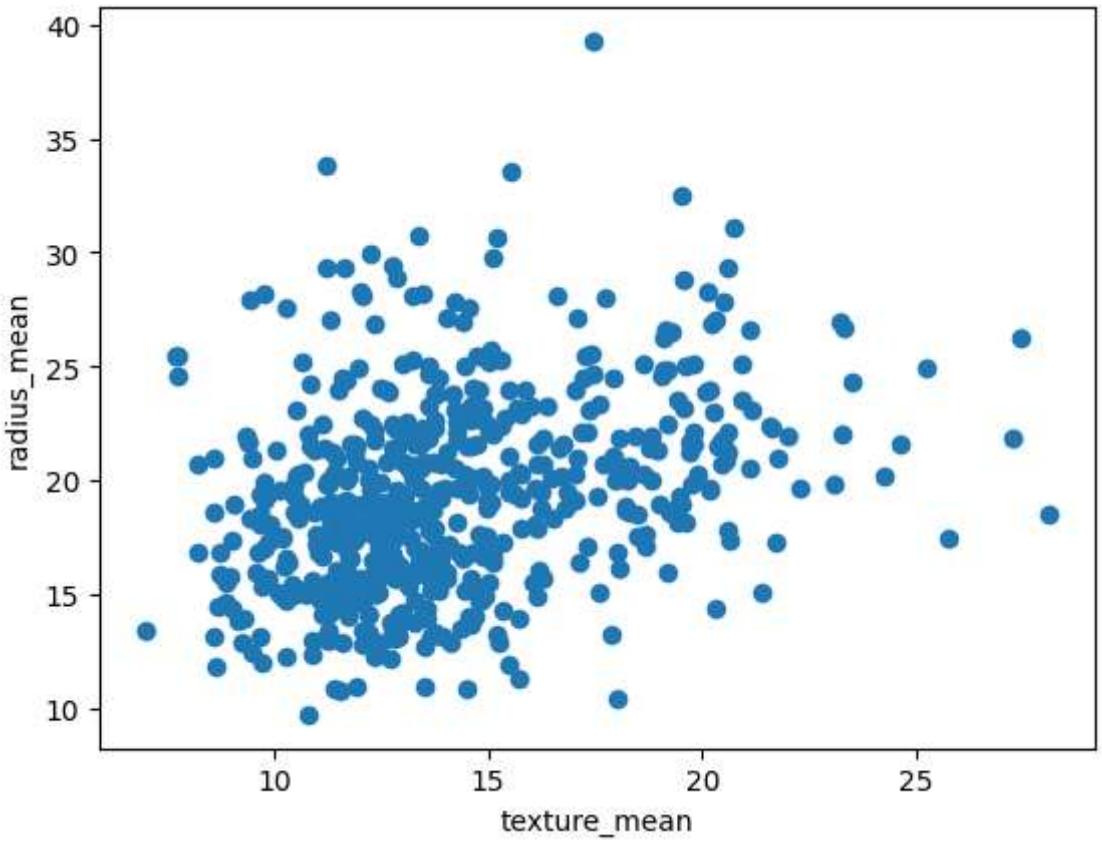
	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean
0	842302	M	17.99	10.38	122.80	1001.0	0.11840
1	842517	M	20.57	17.77	132.90	1326.0	0.08474
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960
3	84348301	M	11.42	20.38	77.58	386.1	0.14250
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030
...	...	...	...	...	...	...	...
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

DATA VISUVALIZATION

In [32]:
```
plt.scatter(df["radius_mean"],df["texture_mean"])
plt.xlabel("texture_mean")
plt.ylabel("radius_mean")
```

Out[32]:
```
Text(0, 0.5, 'radius_mean')
```



In [33]:
```
from sklearn.cluster import KMeans
km=KMeans()
km
```

Out[33]:
```
KMeans()
```

In [34]:
```
y_predicted=km.fit_predict(df[["texture_mean","radius_mean"]])
y_predicted
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
g: 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(
```

Out[34]:
```
array([7, 6, 6, 5, 6, 7, 6, 0, 1, 1, 0, 0, 3, 1, 1, 4, 0, 0, 6, 7, 7, 2,
       7, 3, 0, 7, 0, 6, 1, 7, 3, 5, 3, 3, 0, 0, 0, 5, 1, 0, 1, 1, 3, 0,
       1, 6, 5, 5, 2, 1, 1, 7, 5, 6, 0, 5, 6, 0, 5, 2, 2, 5, 1, 2, 1, 1,
       5, 5, 5, 7, 6, 2, 3, 7, 5, 0, 2, 7, 3, 5, 1, 7, 3, 3, 2, 6, 0, 3,
       1, 7, 1, 0, 7, 5, 0, 3, 5, 5, 2, 0, 1, 2, 5, 5, 5, 7, 5, 5, 6, 1,
       5, 1, 0, 5, 2, 1, 2, 7, 0, 6, 2, 6, 6, 7, 7, 7, 1, 6, 7, 3, 2, 0,
       0, 7, 6, 1, 5, 2, 7, 2, 2, 0, 5, 7, 2, 2, 5, 0, 7, 5, 1, 5, 2, 2,
       7, 5, 0, 0, 2, 2, 5, 6, 6, 1, 6, 0, 2, 0, 3, 7, 2, 0, 7, 2, 2, 2,
       5, 0, 1, 2, 6, 3, 0, 2, 0, 2, 6, 5, 5, 7, 1, 1, 5, 4, 1, 7, 1, 6,
       6, 0, 5, 0, 3, 1, 5, 7, 5, 0, 1, 7, 6, 5, 6, 3, 1, 7, 5, 5, 6, 3,
       7, 7, 5, 0, 7, 7, 2, 7, 1, 1, 0, 4, 4, 3, 2, 0, 3, 6, 4, 4, 7, 2,
       5, 1, 3, 5, 5, 7, 1, 2, 3, 5, 6, 7, 6, 7, 3, 7, 0, 4, 3, 0, 0, 0,
       0, 3, 5, 1, 7, 5, 7, 2, 6, 2, 3, 5, 2, 6, 5, 7, 3, 2, 6, 0, 7, 5,
       1, 2, 5, 5, 0, 0, 7, 5, 2, 7, 2, 5, 0, 1, 6, 5, 3, 5, 5, 1, 7, 2,
       7, 7, 5, 7, 2, 2, 5, 5, 2, 6, 5, 5, 2, 6, 2, 6, 2, 5, 7, 5, 0, 0,
       7, 5, 5, 2, 5, 0, 7, 6, 5, 3, 7, 5, 2, 6, 2, 2, 5, 7, 2, 2, 5, 0,
       6, 1, 2, 5, 5, 7, 2, 5, 5, 1, 5, 0, 7, 6, 3, 5, 6, 6, 0, 7, 6, 6,
       7, 7, 5, 4, 7, 5, 2, 2, 1, 5, 7, 1, 2, 7, 2, 3, 2, 5, 0, 6, 5, 7,
       5, 5, 2, 5, 6, 2, 5, 7, 2, 5, 7, 1, 6, 5, 5, 5, 1, 0, 4, 1, 1, 0,
       2, 1, 5, 7, 2, 0, 5, 1, 2, 1, 5, 5, 0, 5, 6, 6, 7, 0, 5, 7, 0, 7,
       5, 3, 7, 5, 6, 1, 3, 7, 0, 6, 1, 3, 4, 7, 5, 4, 4, 1, 1, 4, 3, 3,
       4, 5, 5, 0, 0, 5, 3, 5, 5, 4, 7, 4, 2, 7, 0, 7, 2, 0, 5, 0, 7, 5,
       7, 7, 7, 6, 5, 0, 1, 7, 6, 2, 0, 0, 5, 5, 6, 6, 7, 1, 7, 6, 2, 2,
       5, 5, 7, 1, 2, 7, 0, 7, 0, 5, 6, 6, 5, 7, 2, 6, 5, 5, 2, 2, 5, 2,
       7, 2, 5, 5, 7, 6, 5, 6, 1, 1, 1, 1, 2, 1, 1, 4, 0, 1, 5, 5, 5, 1,
       1, 1, 4, 1, 4, 4, 5, 4, 1, 1, 4, 4, 4, 3, 6, 3, 4, 3, 1],
      dtype=int32)
```

In [35]:
```
df["cluster"]=y_predicted
df.head()
```

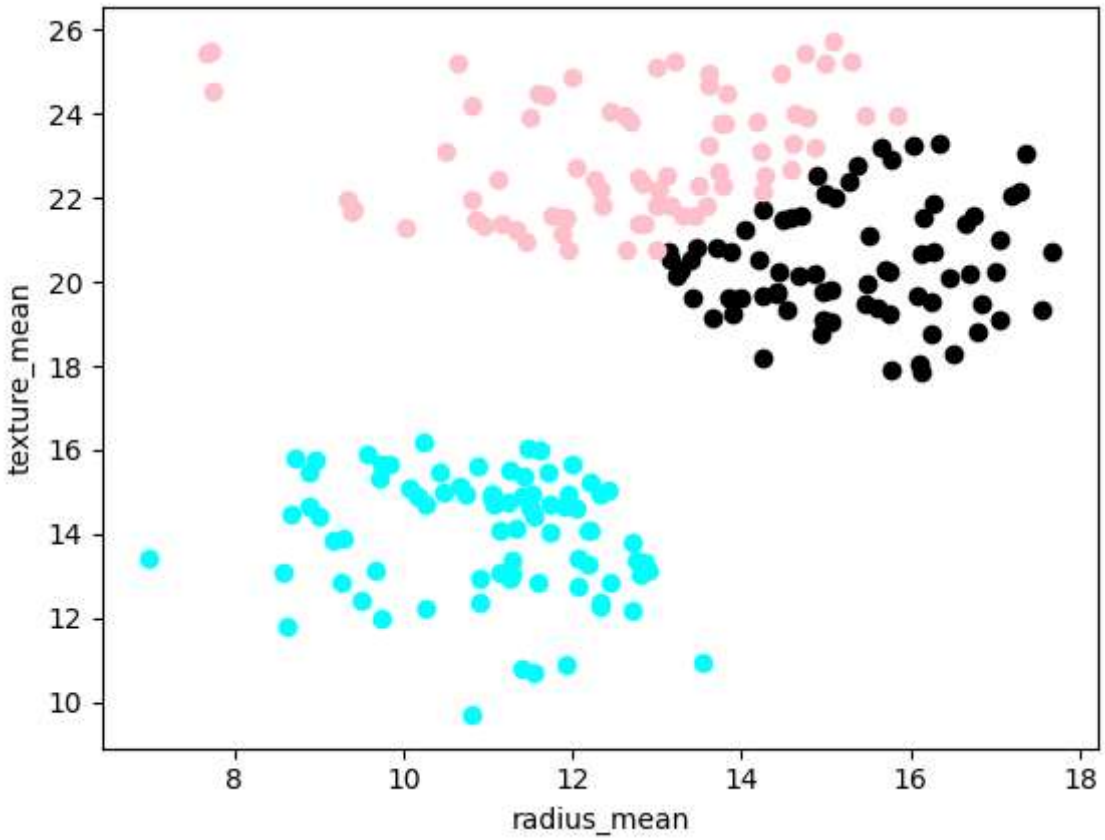
Out[35]:

	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
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 rows × 34 columns

In [36]:
```
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[36]:
```
Text(0, 0.5, 'texture_mean')
```



In [37]:
```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
scaler.fit(df[["texture_mean"]])
df["texture_mean"]=scaler.transform(df[["texture_mean"]])
df.head()
```

Out[37]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean
0	842302	M	17.99	0.022658	122.80	1001.0	0.11840
1	842517	M	20.57	0.272574	132.90	1326.0	0.08474
2	84300903	M	19.69	0.390260	130.00	1203.0	0.10960
3	84348301	M	11.42	0.360839	77.58	386.1	0.14250
4	84358402	M	20.29	0.156578	135.10	1297.0	0.10030

5 rows × 34 columns

In [38]:
```
scaler.fit(df[["radius_mean"]])
df["radius_mean"]=scaler.transform(df[["radius_mean"]])
df.head()
```

Out[38]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean
0	842302	M	0.521037	0.022658	122.80	1001.0	0.11840
1	842517	M	0.643144	0.272574	132.90	1326.0	0.08474
2	84300903	M	0.601496	0.390260	130.00	1203.0	0.10960
3	84348301	M	0.210090	0.360839	77.58	386.1	0.14250
4	84358402	M	0.629893	0.156578	135.10	1297.0	0.10030

5 rows × 34 columns

In [39]:
```
y_predicted=km.fit_predict(df[["radius_mean","texture_mean"]])
y_predicted
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
g: 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(
```

Out[39]:
```
array([1, 2, 2, 4, 2, 1, 2, 7, 7, 0, 7, 1, 6, 7, 7, 0, 7, 7, 2, 1, 1, 5,
       1, 3, 7, 2, 7, 2, 7, 2, 6, 4, 6, 6, 1, 7, 7, 4, 7, 7, 7, 4, 6, 7,
       7, 2, 5, 4, 5, 7, 4, 1, 4, 2, 7, 4, 2, 7, 4, 5, 5, 4, 7, 5, 0, 7,
       4, 4, 4, 1, 2, 5, 6, 1, 4, 7, 1, 2, 6, 4, 4, 1, 3, 6, 5, 2, 7, 6,
       7, 1, 7, 7, 1, 4, 7, 6, 4, 4, 5, 7, 0, 5, 4, 4, 4, 1, 4, 4, 3, 4,
       5, 4, 7, 4, 5, 4, 5, 1, 7, 2, 5, 2, 3, 1, 1, 1, 0, 2, 1, 6, 5, 7,
       7, 1, 2, 7, 4, 5, 1, 5, 5, 1, 4, 1, 5, 5, 4, 7, 1, 1, 7, 4, 5, 5,
       1, 4, 2, 2, 5, 5, 4, 2, 2, 7, 3, 7, 5, 2, 6, 1, 5, 7, 1, 5, 5, 5,
       4, 7, 7, 1, 3, 6, 7, 5, 7, 5, 2, 4, 4, 1, 7, 7, 4, 0, 7, 1, 7, 2,
       2, 7, 4, 2, 3, 7, 4, 1, 4, 2, 7, 1, 2, 4, 3, 6, 7, 1, 4, 4, 2, 6,
       1, 1, 4, 7, 1, 1, 5, 1, 0, 7, 2, 0, 0, 6, 5, 7, 3, 2, 0, 6, 1, 1,
       4, 7, 6, 4, 1, 1, 0, 5, 6, 4, 2, 2, 2, 1, 6, 1, 7, 0, 6, 6, 2, 7,
       2, 6, 4, 7, 1, 4, 1, 5, 3, 5, 6, 4, 5, 2, 1, 1, 6, 5, 2, 7, 1, 4,
       4, 1, 4, 4, 7, 7, 1, 4, 1, 1, 5, 4, 1, 4, 2, 4, 6, 4, 4, 0, 1, 5,
       1, 1, 4, 1, 1, 5, 4, 4, 5, 2, 4, 4, 5, 2, 1, 2, 5, 4, 1, 4, 7, 7,
       1, 4, 4, 5, 4, 2, 1, 2, 4, 3, 1, 5, 5, 2, 5, 5, 4, 1, 5, 5, 4, 7,
       3, 0, 5, 4, 4, 1, 5, 4, 4, 7, 4, 2, 1, 2, 6, 4, 2, 3, 7, 1, 2, 2,
       1, 1, 4, 0, 1, 4, 5, 5, 7, 4, 1, 7, 5, 1, 5, 6, 5, 5, 7, 3, 4, 1,
       7, 4, 5, 4, 2, 5, 4, 1, 5, 4, 1, 7, 2, 4, 4, 4, 4, 7, 0, 4, 4, 7,
       5, 4, 4, 1, 5, 7, 4, 4, 5, 4, 4, 4, 7, 4, 2, 2, 1, 7, 4, 1, 7, 1,
       4, 6, 1, 4, 2, 0, 6, 1, 7, 2, 4, 6, 0, 1, 4, 0, 0, 0, 0, 0, 6, 3,
       0, 4, 4, 7, 7, 4, 6, 4, 4, 0, 1, 0, 5, 1, 7, 1, 5, 7, 4, 7, 1, 1,
       1, 1, 1, 2, 5, 2, 7, 1, 2, 5, 7, 7, 4, 4, 2, 2, 1, 0, 1, 3, 5, 5,
       4, 4, 1, 7, 5, 1, 7, 1, 7, 4, 2, 2, 4, 1, 5, 3, 4, 7, 5, 5, 4, 5,
       1, 5, 4, 4, 1, 2, 4, 2, 7, 0, 0, 0, 5, 0, 0, 0, 7, 7, 5, 5, 4, 0,
       4, 4, 0, 4, 0, 0, 4, 0, 7, 0, 0, 0, 0, 6, 3, 6, 6, 6, 0],
      dtype=int32)
```

In [40]:
```
df["New Cluster"]=y_predicted
df.head()
```

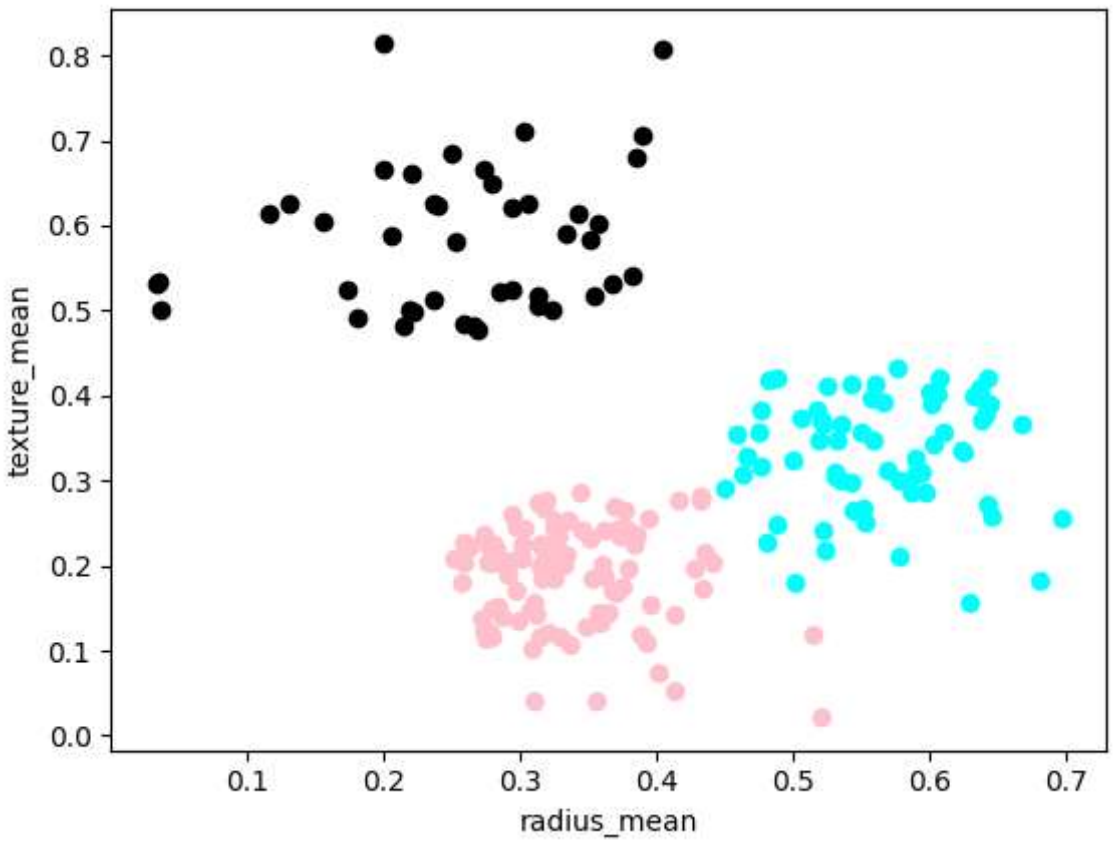
Out[40]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean
0	842302	M	0.521037	0.022658	122.80	1001.0	0.11840
1	842517	M	0.643144	0.272574	132.90	1326.0	0.08474
2	84300903	M	0.601496	0.390260	130.00	1203.0	0.10960
3	84348301	M	0.210090	0.360839	77.58	386.1	0.14250
4	84358402	M	0.629893	0.156578	135.10	1297.0	0.10030

5 rows × 35 columns

In [41]:
```
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[41]:
```
Text(0, 0.5, 'texture_mean')
```

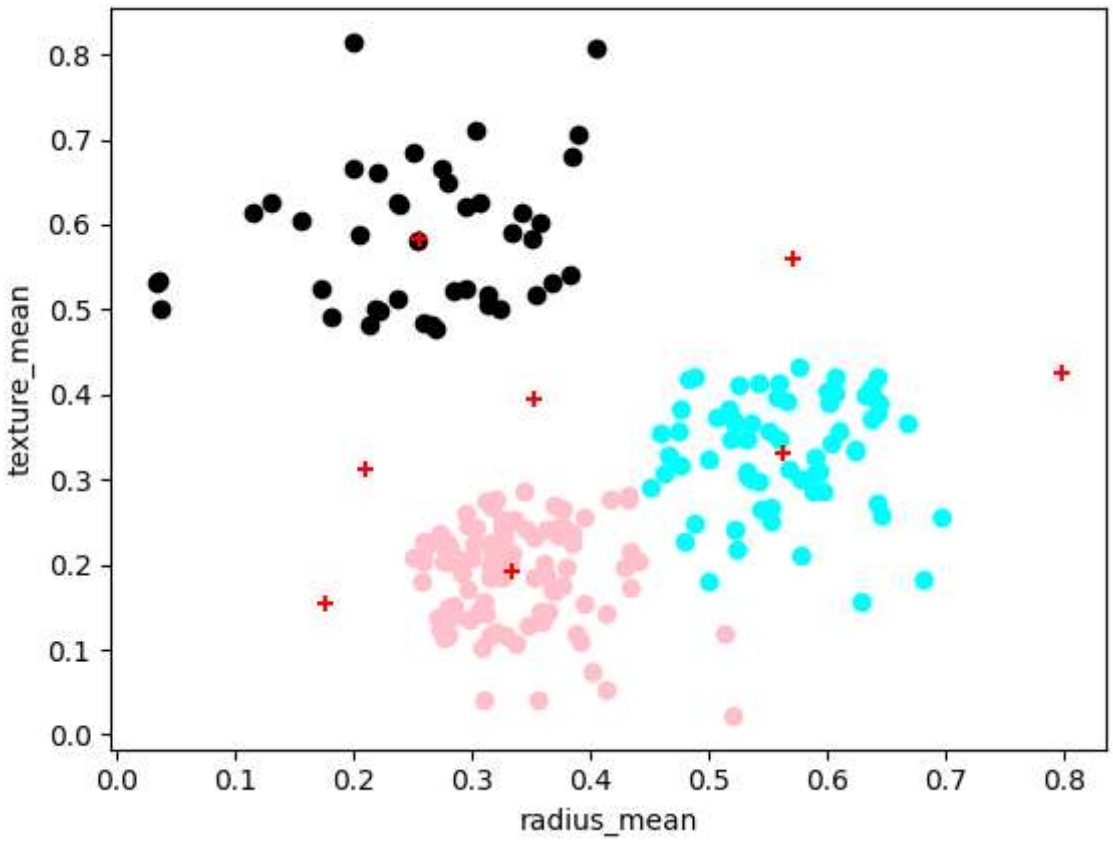


In [42]:
```
km.cluster_centers_
```

Out[42]:
```
array([[0.25627183, 0.58431314],
       [0.33489471, 0.19101622],
       [0.56287997, 0.33184226],
       [0.79840767, 0.42469846],
       [0.21015104, 0.31104952],
       [0.17694105, 0.15527139],
       [0.57132058, 0.55893025],
       [0.35339953, 0.39439771]])
```

In [43]:
```
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",marker="+")
plt.xlabel("radius_mean")
plt.ylabel("texture_mean")
```

Out[43]:
```
Text(0, 0.5, 'texture_mean')
```



In [44]:
```
k_rng=range(1,10)
sse=[]
```

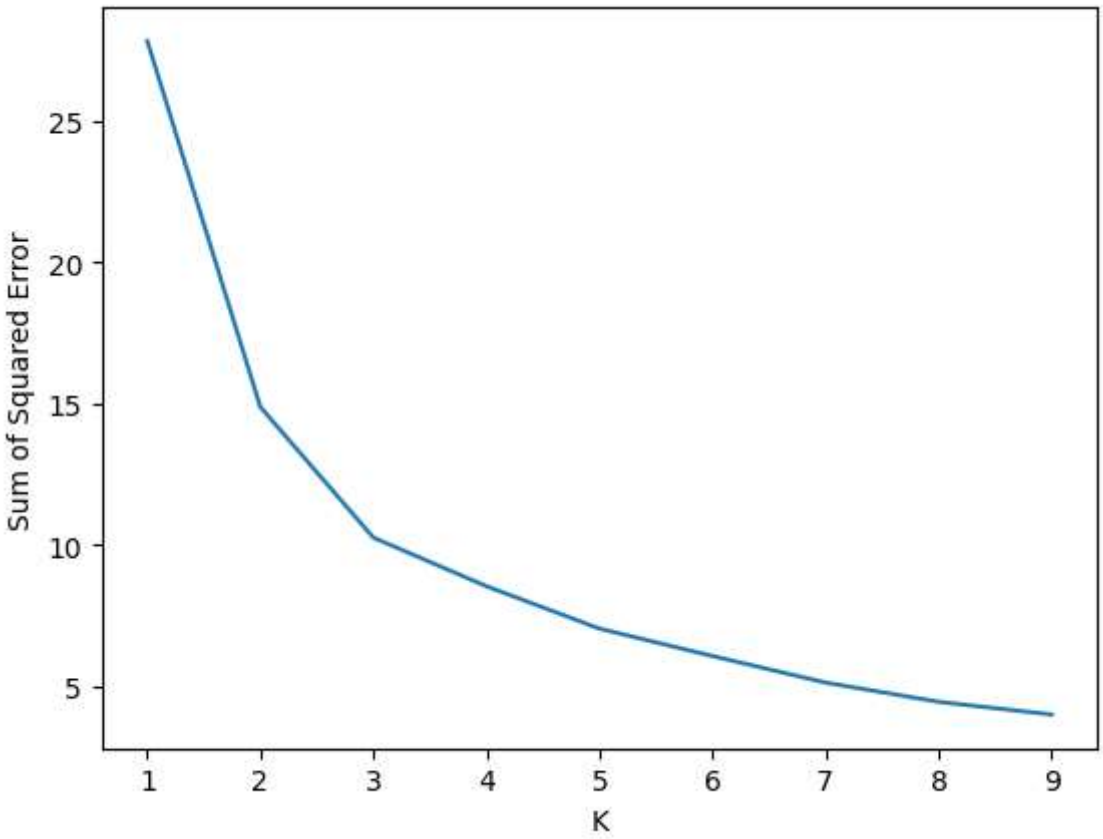
ELBOW METHOD:-

In [45]:
```
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")
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
g: 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(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
g: 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(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
g: 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(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
g: 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(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
g: 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(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
g: 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(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
g: 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(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
g: 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(
[27.81750759504307, 14.872032958271165, 10.252751496105201, 8.533546029699322, 7.0342
60811831776, 6.064527806713256, 5.125301203101715, 4.44413586208949, 3.99610143201293
4]
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
g: 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(
```

Out[45]:
```
Text(0, 0.5, 'Sum of Squared Error')
```



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