To Predict The breast cancer 0n various features of the dataset

importing Libraries

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

Data Collection

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

Out[2]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	...
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	...
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	...
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	...
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	...
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	...
...	...	...	...	...	...	...	...	...	...	...	...
564	926424	M	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	...
565	926682	M	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	...
566	926954	M	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	...
567	927241	M	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	...
568	92751	B	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	...

569 rows × 33 columns

Data Preprocessing

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

Out[3]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	...	t
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	...	
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	...	
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	...	
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	...	
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	...	

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	concave points_mean	...	t
564	926424	M	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	...	
565	926682	M	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	...	
566	926954	M	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	...	
567	927241	M	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	...	
568	92751	B	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	...	

5 rows × 33 columns

In [5]:

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

Out[5]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	...
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	...
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	...
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	...
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	...
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	...
...	...	...	...	...	...	...	...	...	...	...	...
564	926424	M	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	...
565	926682	M	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	...
566	926954	M	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	...
567	927241	M	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	...
568	92751	B	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	...

569 rows × 32 columns

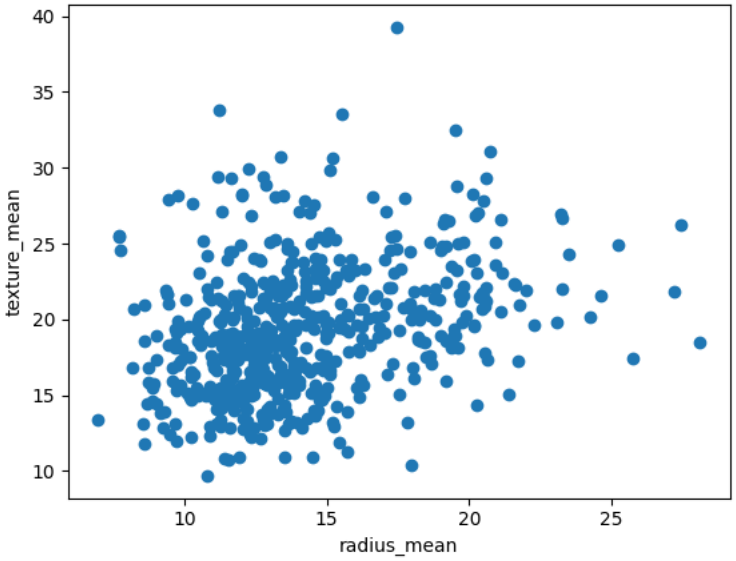
Exploratory Data Analysis

In [6]:

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

Out[6]:

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



K-means Clutstering

In [7]:

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

Out[7]:

```
▾ KMeans
KMeans()
```

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

```
C:\Users\anu\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The defa
ult 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[8]:
```
array([3, 0, 0, 5, 0, 3, 7, 2, 2, 2, 2, 7, 6, 2, 2, 4, 7, 7, 0, 3, 3, 1,
       3, 0, 7, 7, 2, 7, 2, 3, 6, 5, 6, 6, 7, 7, 2, 5, 2, 2, 2, 2, 6, 5,
       2, 7, 1, 5, 1, 2, 2, 3, 5, 7, 2, 5, 0, 2, 5, 1, 1, 5, 2, 1, 2, 2,
       5, 5, 5, 3, 0, 1, 6, 3, 5, 7, 3, 7, 6, 5, 2, 3, 6, 6, 1, 7, 2, 6,
       2, 3, 2, 2, 3, 5, 7, 0, 5, 5, 1, 7, 2, 1, 5, 5, 5, 3, 5, 5, 0, 2,
       5, 2, 7, 5, 1, 2, 1, 3, 2, 7, 1, 7, 0, 3, 3, 3, 2, 0, 3, 6, 1, 7,
       7, 3, 0, 2, 5, 1, 7, 1, 1, 7, 5, 3, 1, 1, 5, 7, 3, 5, 2, 5, 1, 1,
       3, 5, 7, 7, 1, 1, 5, 7, 0, 2, 0, 7, 1, 7, 6, 3, 1, 5, 3, 1, 1, 1,
       5, 7, 2, 3, 0, 6, 7, 1, 2, 1, 7, 5, 5, 3, 2, 2, 5, 4, 2, 3, 2, 7,
       0, 7, 5, 7, 6, 2, 5, 3, 5, 7, 2, 3, 0, 5, 0, 6, 2, 3, 5, 5, 0, 6,
       3, 3, 5, 7, 5, 3, 1, 3, 2, 2, 7, 4, 4, 6, 1, 2, 6, 0, 4, 4, 3, 3,
       5, 2, 6, 5, 5, 3, 2, 1, 6, 5, 0, 7, 0, 3, 6, 7, 2, 4, 6, 7, 7, 7,
       7, 6, 5, 2, 3, 5, 3, 1, 0, 1, 6, 5, 1, 0, 5, 3, 6, 1, 0, 7, 3, 5,
       2, 3, 5, 5, 7, 7, 3, 5, 3, 3, 1, 5, 5, 2, 0, 5, 6, 5, 5, 2, 3, 1,
       3, 3, 5, 3, 3, 1, 5, 5, 1, 7, 5, 5, 1, 0, 3, 0, 1, 5, 3, 5, 7, 7,
       3, 5, 5, 1, 5, 7, 3, 0, 5, 6, 3, 5, 1, 0, 1, 1, 5, 3, 1, 1, 5, 7,
       0, 2, 1, 5, 5, 3, 1, 5, 5, 2, 5, 7, 3, 0, 6, 5, 0, 0, 2, 3, 0, 0,
       3, 3, 5, 4, 3, 5, 1, 1, 2, 5, 3, 2, 1, 3, 1, 6, 1, 1, 7, 0, 5, 3,
       5, 5, 1, 5, 7, 1, 5, 3, 3, 5, 3, 2, 7, 5, 5, 5, 2, 2, 4, 2, 2, 7,
       1, 2, 5, 3, 1, 5, 5, 5, 1, 2, 5, 5, 2, 5, 0, 0, 3, 5, 5, 3, 5, 3,
       5, 6, 3, 5, 7, 2, 6, 7, 7, 0, 2, 6, 4, 3, 5, 4, 4, 2, 2, 4, 6, 6,
       4, 5, 5, 5, 2, 5, 6, 5, 5, 4, 3, 4, 1, 3, 2, 3, 1, 7, 5, 5, 3, 5,
       3, 3, 3, 0, 1, 7, 2, 3, 7, 1, 2, 7, 5, 5, 7, 0, 3, 2, 3, 0, 1, 1,
       5, 5, 3, 2, 1, 3, 2, 3, 7, 5, 7, 0, 5, 5, 1, 0, 5, 5, 1, 1, 5, 1,
       3, 1, 5, 5, 3, 0, 5, 0, 2, 2, 2, 2, 1, 2, 2, 4, 2, 2, 1, 5, 5, 2,
       2, 2, 4, 2, 4, 4, 5, 4, 2, 2, 4, 4, 4, 6, 0, 6, 6, 6, 2])
```

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

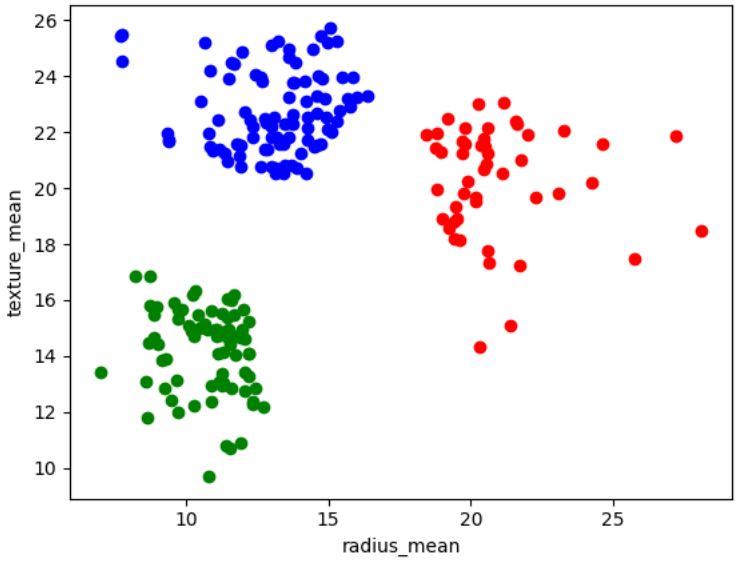
Out[9]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	...
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	...
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	...
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	...
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	...
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	...

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="red")
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[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_mean	compactness_mean	concavity_mean	concave points_mean	...
0	842302	M	17.99	0.022658	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	...
1	842517	M	20.57	0.272574	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	...
2	84300903	M	19.69	0.390260	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	...
3	84348301	M	11.42	0.360839	77.58	386.1	0.14250	0.28390	0.2414	0.10520	...
4	84358402	M	20.29	0.156578	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	...

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_mean	compactness_mean	concavity_mean	concave points_mean	...
0	842302	M	0.521037	0.022658	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	...
1	842517	M	0.643144	0.272574	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	...
2	84300903	M	0.601496	0.390260	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	...
3	84348301	M	0.210090	0.360839	77.58	386.1	0.14250	0.28390	0.2414	0.10520	...
4	84358402	M	0.629893	0.156578	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	...

5 rows × 34 columns

In [13]:

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

```
C:\Users\anu\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The defa
ult 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[13]:

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

In [14]:

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

Out[14]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	...
0	842302	M	0.521037	0.022658	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	...
1	842517	M	0.643144	0.272574	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	...
2	84300903	M	0.601496	0.390260	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	...
3	84348301	M	0.210090	0.360839	77.58	386.1	0.14250	0.28390	0.2414	0.10520	...
4	84358402	M	0.629893	0.156578	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	...

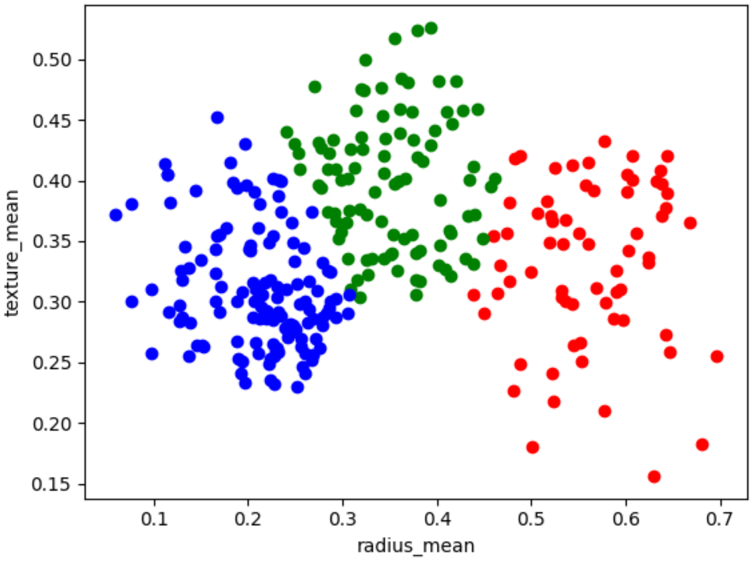
5 rows × 35 columns

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="red")
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[15]:

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



In [16]:

```
km.cluster_centers_
```

Out[16]:

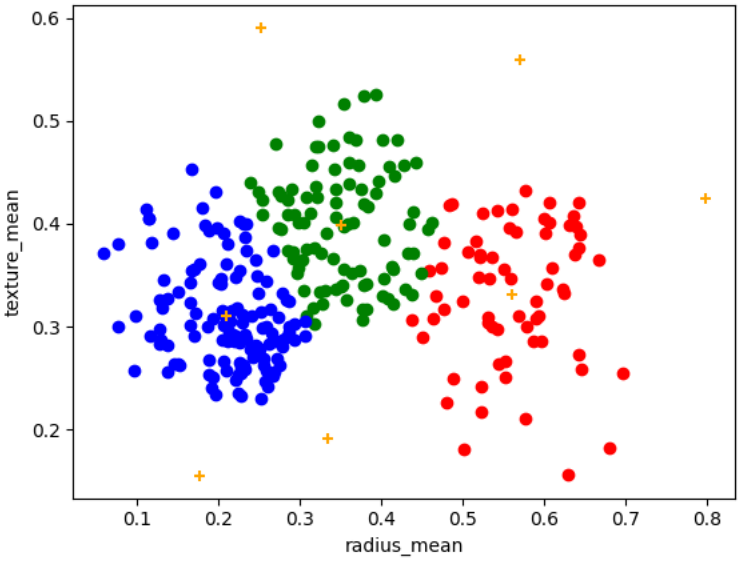
```
array([[0.56101927, 0.3314624 ],
       [0.3513457 , 0.39855639],
       [0.21015104, 0.31104952],
       [0.57132058, 0.55893025],
       [0.25179611, 0.59077612],
       [0.79840767, 0.42469846],
       [0.17694105, 0.15527139],
       [0.33489471, 0.19101622]])
```

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="red")
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[17]:

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



In [19]:

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

In [23]:

```
for k in k_rng:
    km=KMeans(n_clusters=k)
    km.fit(df[["radius_mean","texture_mean"]])
    sse.append(km.inertia_)
```

```
C:\Users\anu\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The defa
ult 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\anu\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The defa
ult 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\anu\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The defa
ult 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\anu\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The defa
ult 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\anu\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The defa
ult 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\anu\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The defa
ult 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\anu\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The defa
ult 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\anu\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The defa
ult 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\anu\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The defa
ult 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(
```

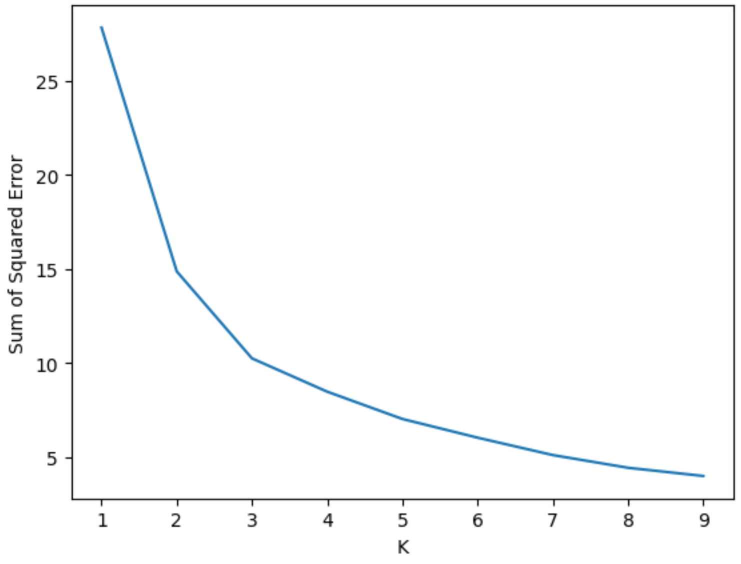
In [24]:

```
print(sse)
plt.plot(k_rng,sse)
plt.xlabel("K")
plt.ylabel("Sum of Squared Error")
```

```
[27.81750759504307, 14.872032958271172, 10.252751496105198, 8.484725277027607, 7.035012847498983, 6.041110774701765, 5.1173791
10317932, 4.442873133317689, 4.007280556362495]
```

Out[24]:

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



CONCLUSION:BY USING K-MEANS CLUSTERING MODEL WE GET BEST ACCURACY

In [ ]: