Problem Statement:

```
Breast cancer prediction
```

In [31]:

```
import pandas as pd
from matplotlib import pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")
```

In [32]:

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

Out[32]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothne
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 [33]:

```
df.head()
```

Out[33]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness
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 × 33 columns

In [34]:

```
df.tail()
```

Out[34]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness
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	

5 rows × 33 columns

In [35]:

```
df.describe()
```

Out[35]:

	id	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_
count	5.690000e+02	569.000000	569.000000	569.000000	569.000000	569.0
mean	3.037183e+07	14.127292	19.289649	91.969033	654.889104	0.0
std	1.250206e+08	3.524049	4.301036	24.298981	351.914129	0.0
min	8.670000e+03	6.981000	9.710000	43.790000	143.500000	0.0
25%	8.692180e+05	11.700000	16.170000	75.170000	420.300000	0.0
50%	9.060240e+05	13.370000	18.840000	86.240000	551.100000	0.0
75%	8.813129e+06	15.780000	21.800000	104.100000	782.700000	0.1
max	9.113205e+08	28.110000	39.280000	188.500000	2501.000000	0.1

8 rows × 32 columns

In [36]:

```
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 [37]:

```
df.shape
```

Out[37]:

```
(569, 33)
```

In [38]:

```
df.isnull().any()
```

Out[38]:

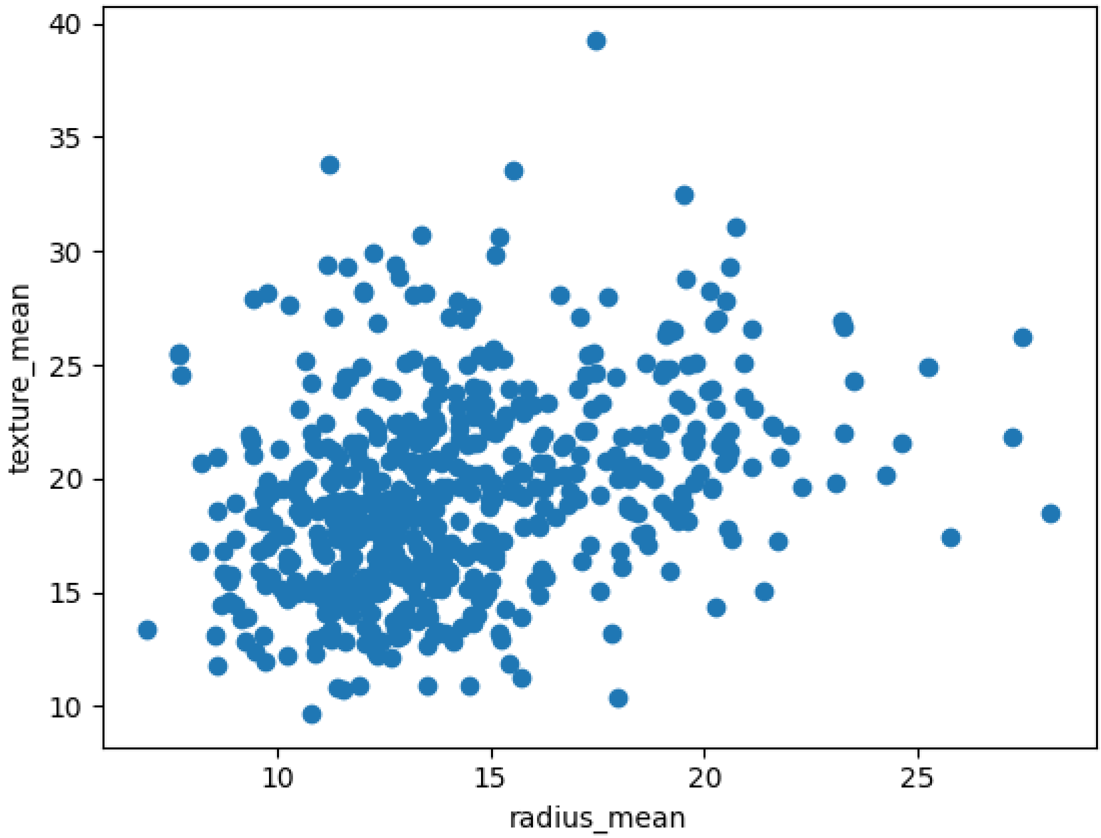
```
id                          False
diagnosis                   False
radius_mean                 False
texture_mean                False
perimeter_mean              False
area_mean                   False
smoothness_mean             False
compactness_mean            False
concavity_mean              False
concave points_mean         False
symmetry_mean               False
fractal_dimension_mean      False
radius_se                   False
texture_se                  False
perimeter_se                False
area_se                     False
smoothness_se               False
compactness_se              False
concavity_se                False
concave points_se           False
symmetry_se                 False
fractal_dimension_se        False
radius_worst                False
texture_worst               False
perimeter_worst             False
area_worst                  False
smoothness_worst            False
compactness_worst           False
concavity_worst             False
concave points_worst        False
symmetry_worst              False
fractal_dimension_worst     False
Unnamed: 32                  True
dtype: bool
```

In [39]:

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

Out[39]:

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



In [40]:

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

Out[40]:

```
▾ KMeans
KMeans()
```

In [41]:

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

Out[41]:

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

In [42]:

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

Out[42]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness
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	

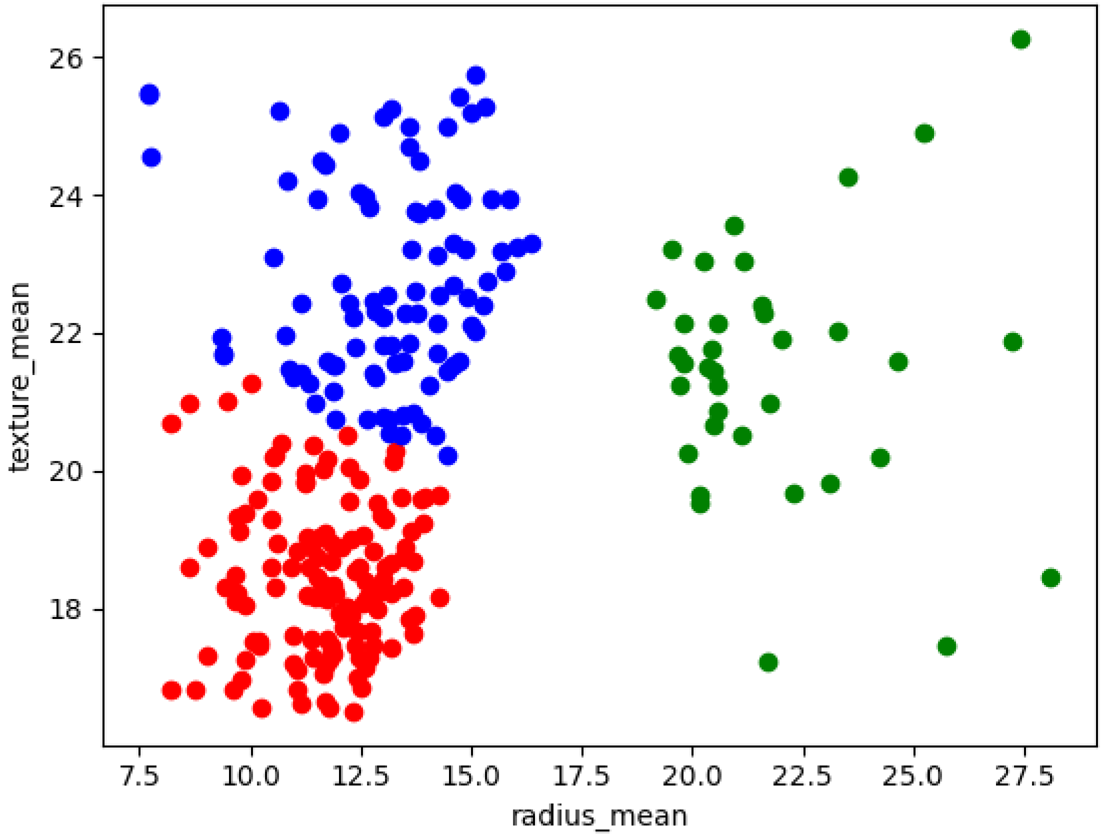
5 rows × 34 columns

In [43]:

```
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[43]:

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



In [44]:

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

Out[44]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness
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 [45]:

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

Out[45]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness
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 [46]:

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

Out[46]:

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

In [47]:

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

Out[47]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness
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	

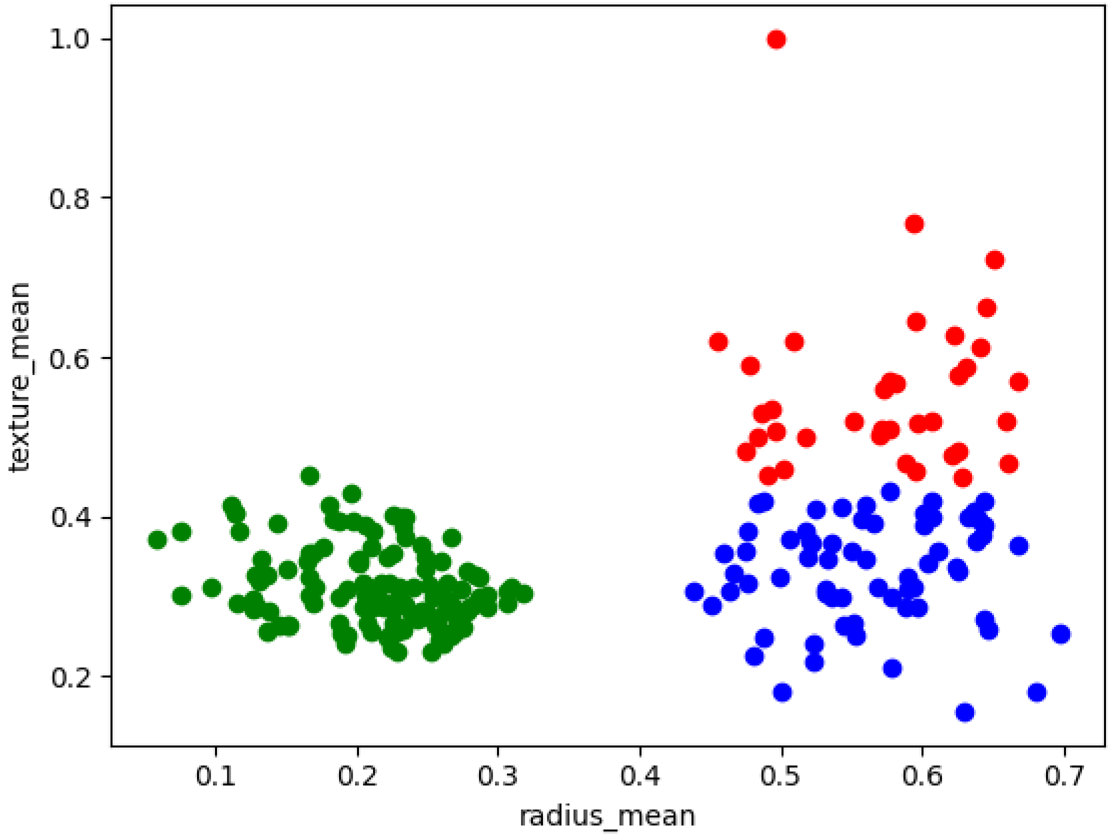
5 rows × 35 columns

In [48]:

```
df1=df[df["New Cluster"]==0]
df2=df[df["New Cluster"]==2]
df3=df[df["New Cluster"]==3]
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[48]:

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



In [49]:

```
km.cluster_centers_
```

Out[49]:

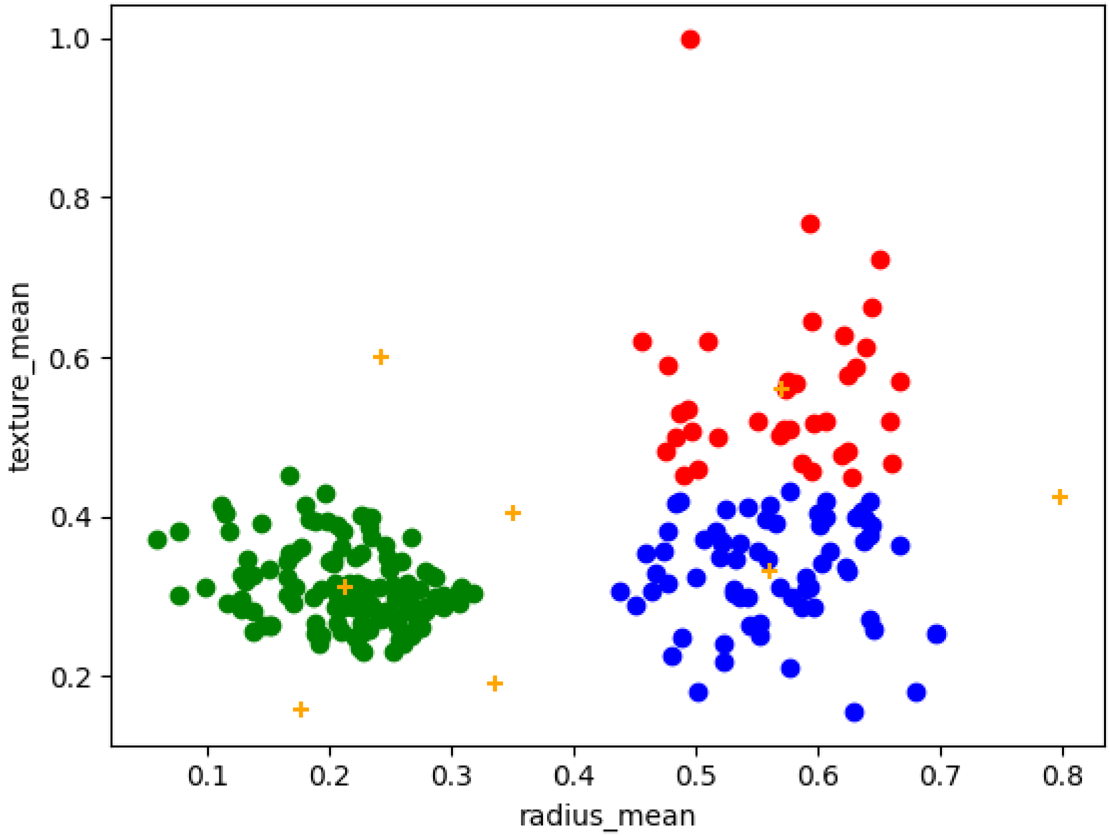
```
array([[0.57132058, 0.55893025],
       [0.17620217, 0.15747668],
       [0.21306768, 0.31137257],
       [0.56101927, 0.3314624 ],
       [0.24279689, 0.59913388],
       [0.33570532, 0.19063107],
       [0.35135576, 0.40520246],
       [0.79840767, 0.42469846]])
```

In [50]:

```
df1=df[df["New Cluster"]==0]
df2=df[df["New Cluster"]==2]
df3=df[df["New Cluster"]==3]
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[50]:

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



In [51]:

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

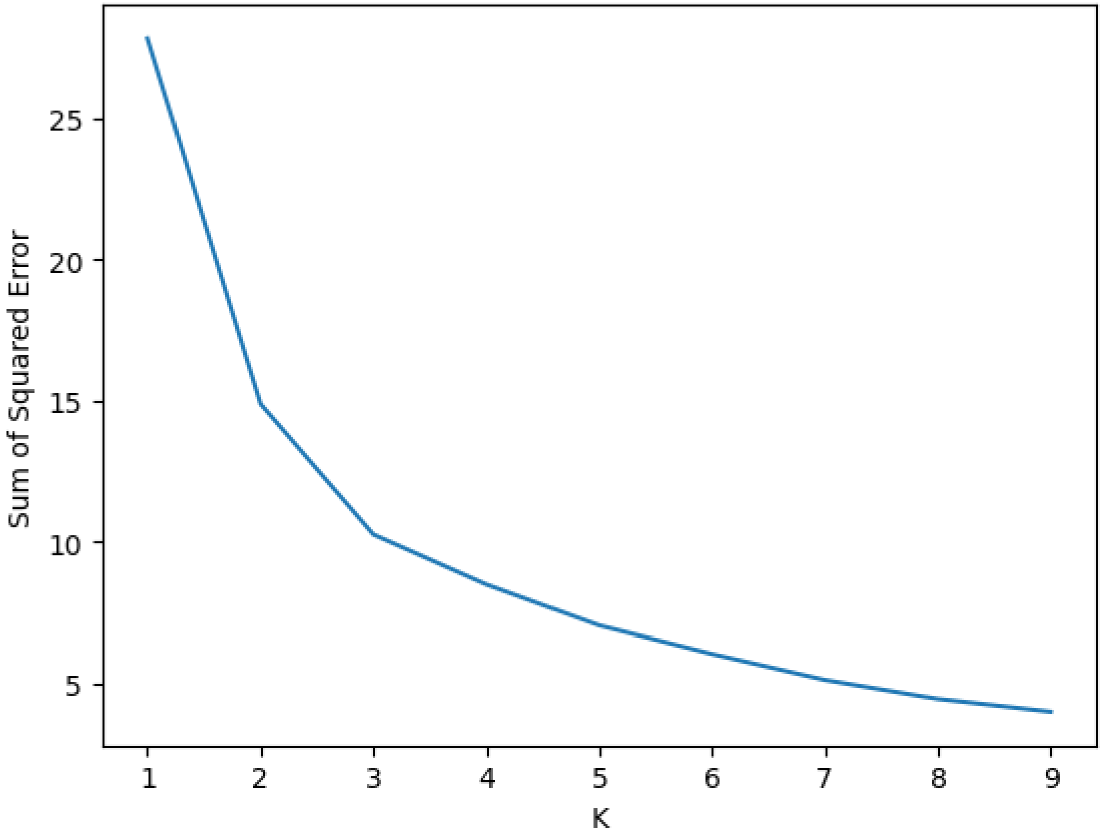
In [52]:

```
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")
```

```
[27.81750759504307, 14.872032958271172, 10.252751496105196, 8.488875784807
327, 7.041107262889024, 6.0313468782509725, 5.117379110317932, 4.444287355
881231, 3.9916276477713915]
```

Out[52]:

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



Conclusion:

For the given dataset we can do prediction by various models,but accuracy from that models are not good.So we prefer K-Means Clustering for this dataset.

In [ ]: