PROBLEM STATEMENT: To predict and study using

the breast cancer diagnostic dataset.

In [1]:

```
import pandas as pd
from matplotlib import pyplot as plt
%matplotlib inline
```

In [2]:

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

Out[2]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothi
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 [3]:

```
df.head()
```

Out[3]:

	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	

5 rows × 33 columns

In [4]:

```
df.tail()
```

Out[4]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothne
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 [5]:

```
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 [6]:

```
df.isnull().sum()
```

Out[6]:

```
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 [7]:

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

Out[7]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothi
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	

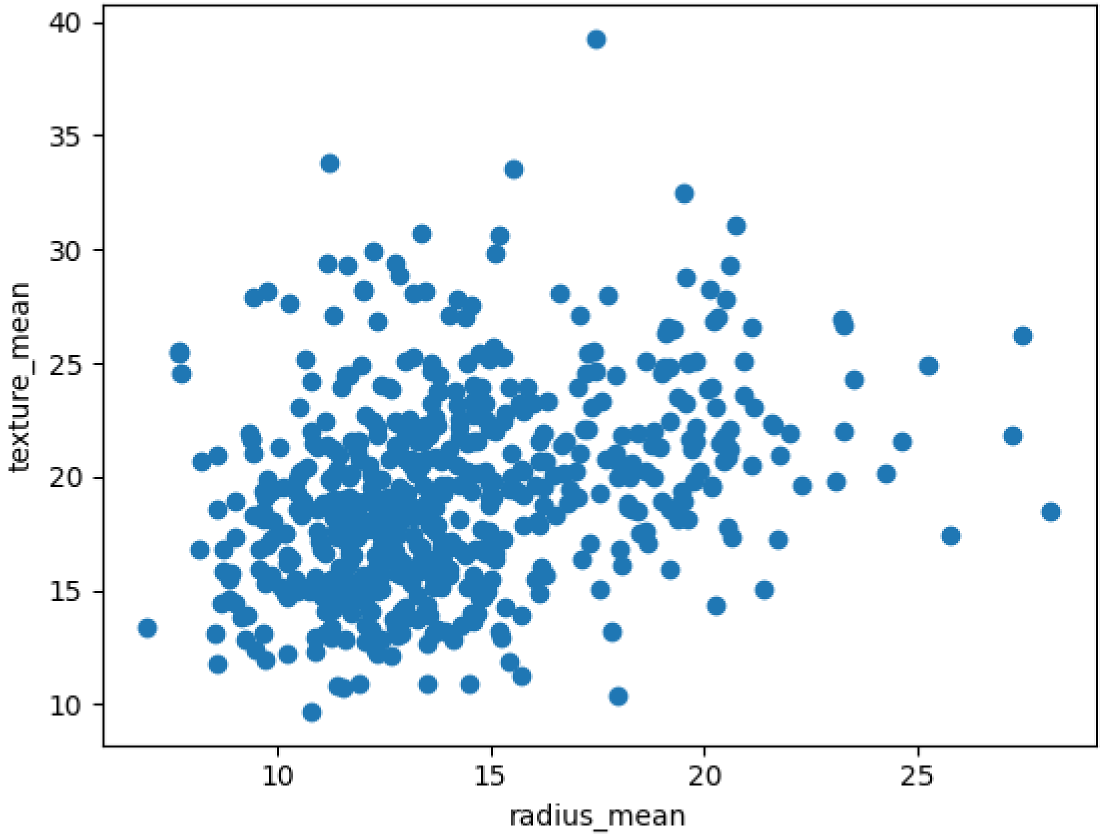
569 rows × 32 columns

In [8]:

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

Out[8]:

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



In [9]:

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

Out[9]:

```
▾ KMeans
KMeans()
```

In [10]:

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

```
C:\Users\jyothi reddy\AppData\Local\Programs\Python\Python311\Lib\site-pa
ckages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `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[10]:

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

In [11]:

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

Out[11]:

	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	

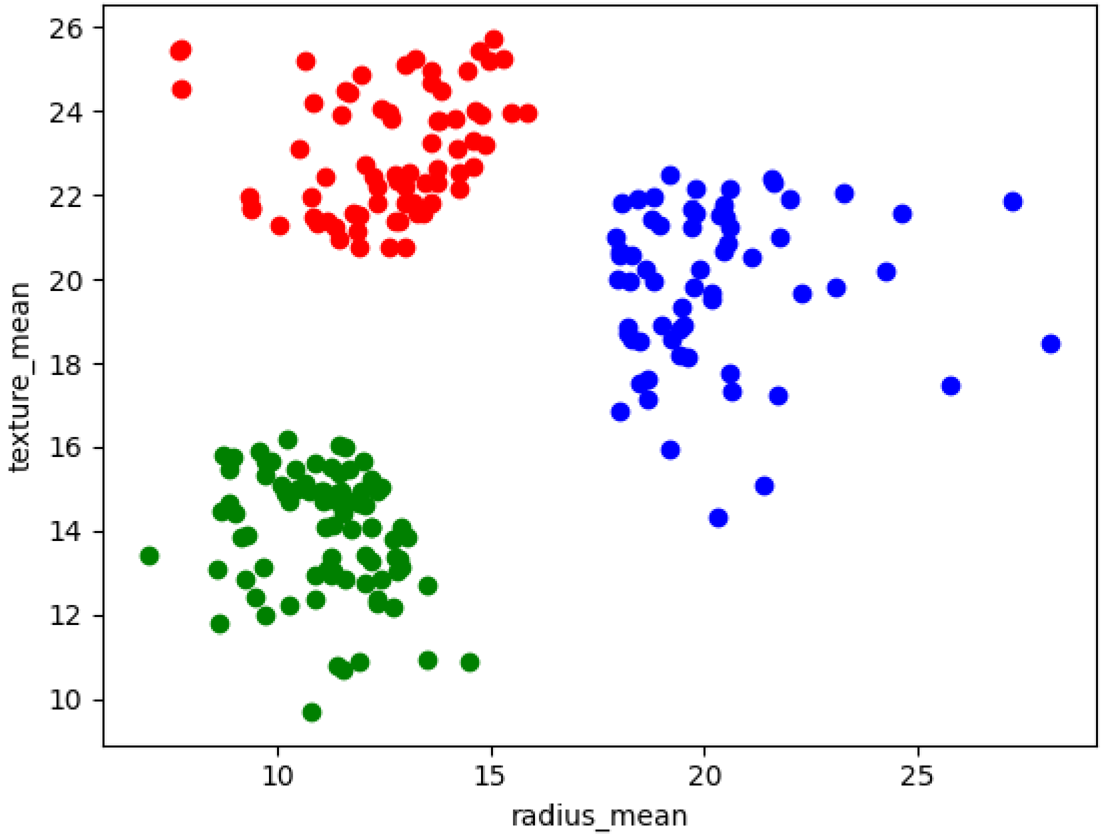
5 rows × 34 columns

In [12]:

```
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[12]:

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



In [13]:

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

Out[13]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothne
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 [14]:

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

Out[14]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothne
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 [15]:

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

```
C:\Users\jyothi reddy\AppData\Local\Programs\Python\Python311\Lib\site-pa
ckages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `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[15]:

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

In [16]:

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

Out[16]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothne
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	

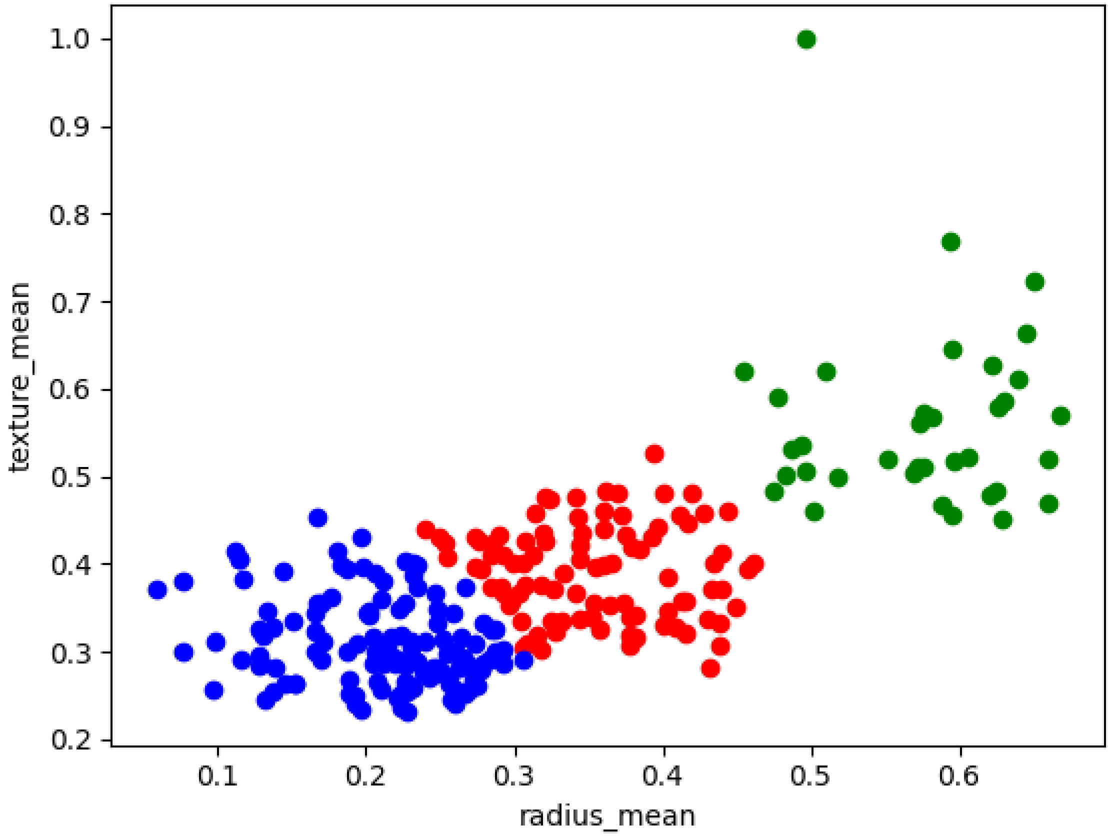
5 rows × 35 columns

In [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.xlabel("radius_mean")
plt.ylabel("texture_mean")
```

Out[17]:

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



In [18]:

```
km.cluster_centers_
```

Out[18]:

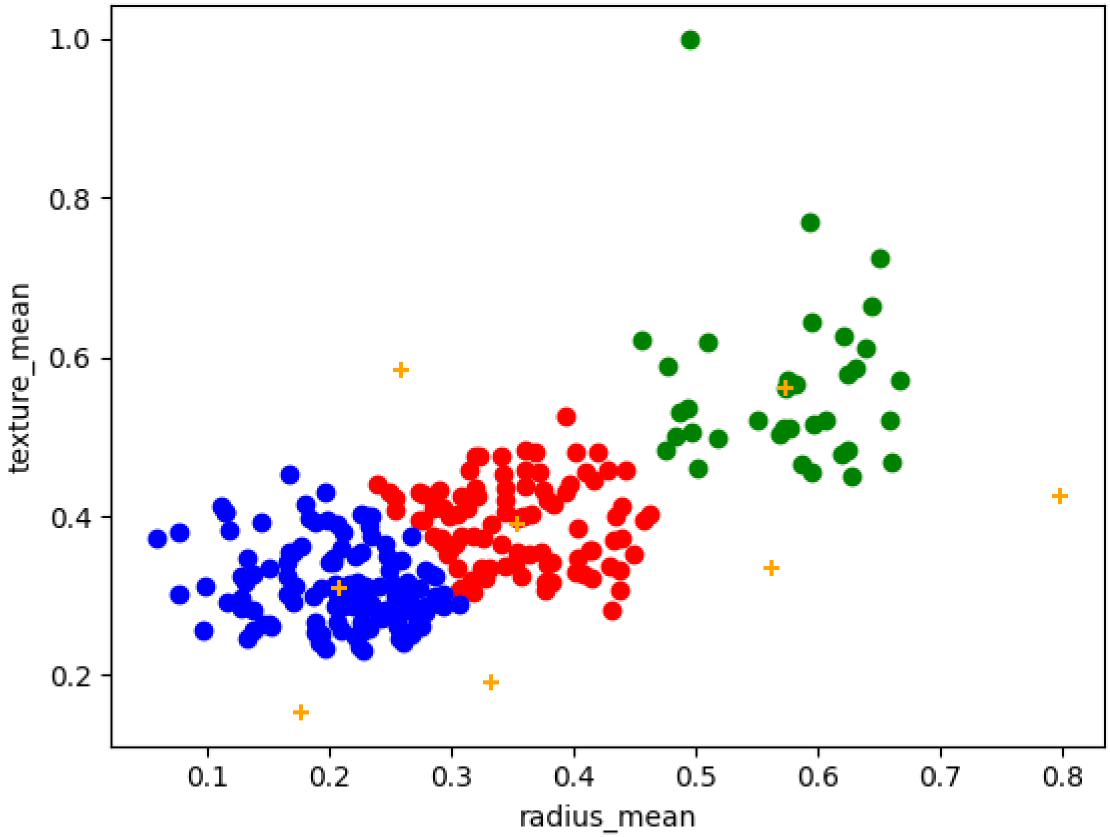
```
array([[0.3534653 , 0.39091896],
       [0.57355872, 0.56191523],
       [0.20878924, 0.31058452],
       [0.3331624 , 0.18999839],
       [0.56180336, 0.33362777],
       [0.2590623 , 0.58293879],
       [0.17652977, 0.15382448],
       [0.79840767, 0.42469846]])
```

In [19]:

```
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[19]:

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



In [20]:

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

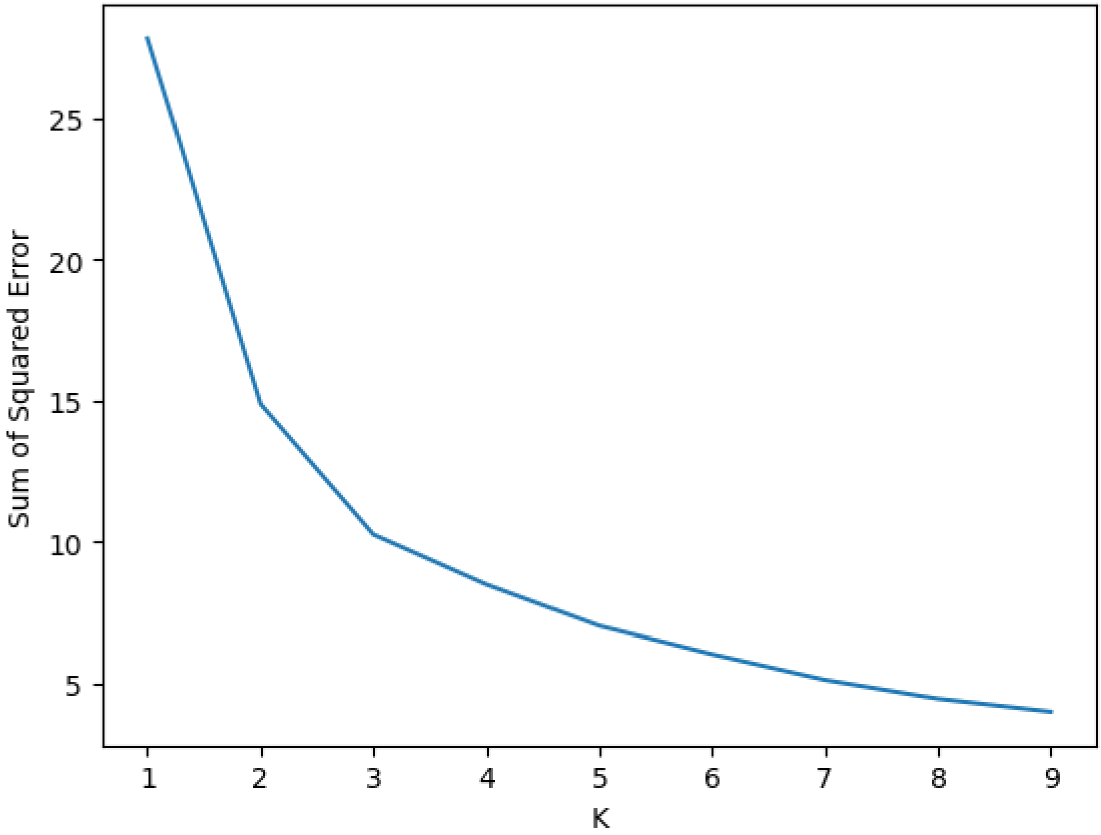
In [21]:

```
for k in k_rng:
 km=KMeans(n_clusters=k)
 km.fit(df[["radius_mean","texture_mean"]])
 sse.append(km.inertia_)
#km.inertia_ will give you the value of sum of square error
print(sse)
plt.plot(k_rng,sse)
plt.xlabel("K")
plt.ylabel("Sum of Squared Error")
```

```
C:\Users\jyothi reddy\AppData\Local\Programs\Python\Python311\Lib\site-pa
ckages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `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\jyothi reddy\AppData\Local\Programs\Python\Python311\Lib\site-pa
ckages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `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\jyothi reddy\AppData\Local\Programs\Python\Python311\Lib\site-pa
ckages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `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\jyothi reddy\AppData\Local\Programs\Python\Python311\Lib\site-pa
ckages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `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\jyothi reddy\AppData\Local\Programs\Python\Python311\Lib\site-pa
ckages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `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\jyothi reddy\AppData\Local\Programs\Python\Python311\Lib\site-pa
ckages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `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\jyothi reddy\AppData\Local\Programs\Python\Python311\Lib\site-pa
ckages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `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\jyothi reddy\AppData\Local\Programs\Python\Python311\Lib\site-pa
ckages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `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\jyothi reddy\AppData\Local\Programs\Python\Python311\Lib\site-pa
ckages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `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, 14.87203295827117, 10.252751496105196, 8.490050221511
442, 7.0300218442414915, 6.024392230917021, 5.119889988004201, 4.45348648
0782376, 3.993531470856821]
```

Out[21]:

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



Conclusion

Based on accuracy of all models that were implemented we can conclude that "K -Means Clustering" is the best model for the given dataset

In [ ]: