PROJECT 4

Problem Statement: Which Model is suitable for BreastCancerPredictionDataset

Importing Libraries

In [2]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

Reading Data ¶

In [3]:
```
df=pd.read_csv(r"C:\Users\P. VIJAY KUMAR\Downloads\BreastCancerPrediction
df
```

Out[3]:

	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 Preprocedding

In [4]:
```
1 df.head()
```

Out[4]:

	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 × 33 columns

In [5]:
```
1 df.tail()
```

Out[5]:

	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	0.09
566	926954	M	16.60	28.08	108.30	858.1	0.08
567	927241	M	20.60	29.33	140.10	1265.0	0.1
568	92751	B	7.76	24.54	47.92	181.0	0.05

5 rows × 33 columns

In [6]:
```
1 df.describe()
```

Out[6]:

	id	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_me
count	5.690000e+02	569.000000	569.000000	569.000000	569.000000	569.0000
mean	3.037183e+07	14.127292	19.289649	91.969033	654.889104	0.0963
std	1.250206e+08	3.524049	4.301036	24.298981	351.914129	0.0140
min	8.670000e+03	6.981000	9.710000	43.790000	143.500000	0.0526
25%	8.692180e+05	11.700000	16.170000	75.170000	420.300000	0.0863
50%	9.060240e+05	13.370000	18.840000	86.240000	551.100000	0.0958
75%	8.813129e+06	15.780000	21.800000	104.100000	782.700000	0.1053
max	9.113205e+08	28.110000	39.280000	188.500000	2501.000000	0.1634

8 rows × 32 columns

In [7]:
```
1 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 [8]:
```
1 df.isnull().sum()
```

Out[8]:
```
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 [9]:
```
1 df.columns
```

Out[9]:
```
Index(['id', 'diagnosis', 'radius_mean', 'texture_mean', 'perimeter_mean',
       'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mean',
       'concave points_mean', 'symmetry_mean', 'fractal_dimension_mean',
       'radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_se',
       'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_se',
       'fractal_dimension_se', 'radius_worst', 'texture_worst',
       'perimeter_worst', 'area_worst', 'smoothness_worst',
       'compactness_worst', 'concavity_worst', 'concave points_worst',
       'symmetry_worst', 'fractal_dimension_worst', 'Unnamed: 32'],
      dtype='object')
```

In [10]:
```
1 df.shape
```

Out[10]:
```
(569, 33)
```

In [11]:
```
df=df.drop("Unnamed: 32",axis=1)
```

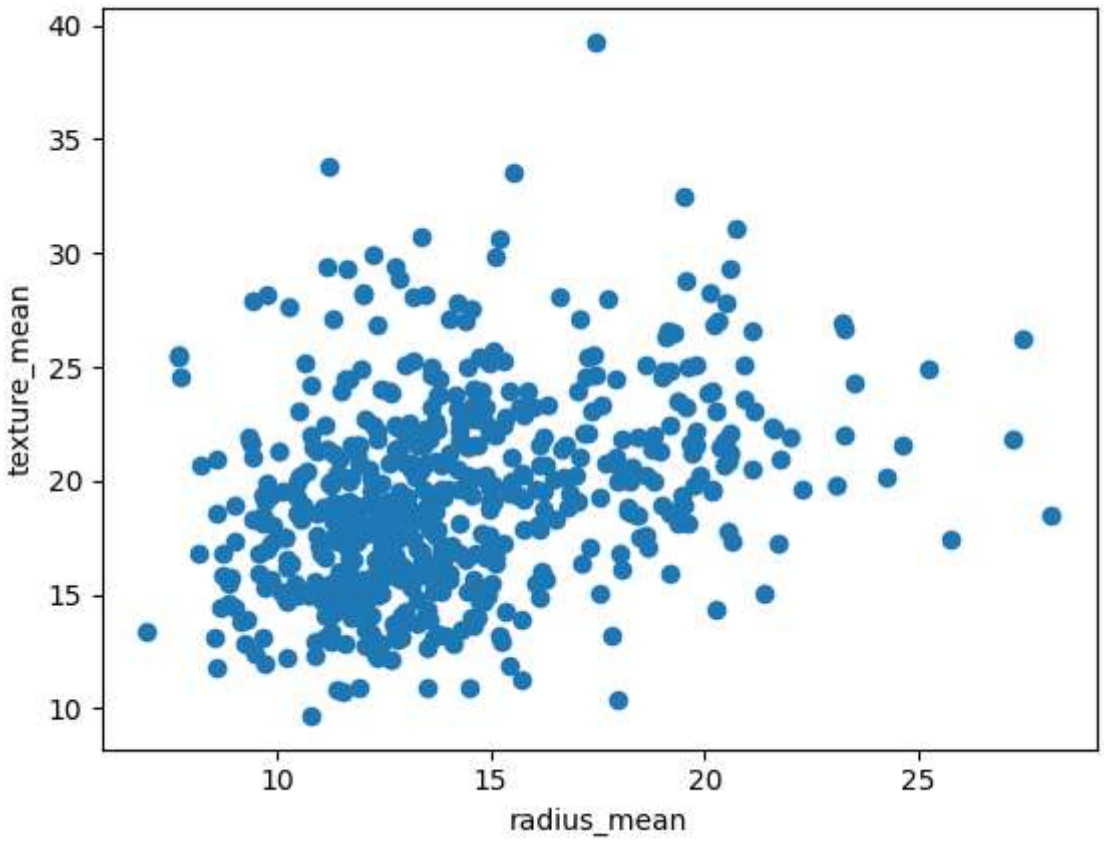
In [12]:
```
df.shape
```

Out[12]: (569, 32)

Data Visualization

In [13]:
```
plt.scatter(df["radius_mean"],df["texture_mean"])
plt.xlabel("radius_mean")
plt.ylabel("texture_mean")
```

Out[13]: Text(0, 0.5, 'texture_mean')



In [14]:
```
from sklearn.cluster import KMeans
km=KMeans()
km

```

Out[14]:
```
KMeans
KMeans()
```

In [15]:
```
y_predicted=km.fit_predict(df[["radius_mean","texture_mean"]])
y_predicted
```

```
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(
```

Out[15]:
```
array([5, 7, 7, 1, 7, 5, 7, 3, 2, 2, 3, 3, 0, 2, 2, 6, 3, 3, 7, 5, 5, 4,
       5, 0, 3, 5, 3, 7, 2, 5, 0, 1, 0, 0, 3, 3, 3, 1, 2, 3, 2, 2, 0, 3,
       2, 7, 1, 1, 4, 2, 2, 5, 1, 7, 3, 1, 7, 3, 1, 4, 4, 1, 2, 4, 2, 2,
       1, 1, 1, 5, 7, 4, 0, 5, 1, 3, 4, 5, 0, 1, 2, 5, 0, 0, 4, 7, 3, 0,
       2, 5, 2, 3, 5, 1, 3, 0, 1, 1, 4, 3, 2, 4, 1, 1, 1, 5, 1, 1, 7, 2,
       1, 2, 3, 1, 4, 2, 4, 5, 3, 7, 4, 7, 7, 5, 5, 5, 2, 7, 5, 0, 4, 3,
       3, 5, 7, 2, 1, 4, 5, 4, 4, 3, 1, 5, 4, 4, 1, 3, 5, 1, 2, 1, 4, 4,
       5, 1, 3, 3, 4, 4, 1, 7, 7, 2, 7, 3, 4, 3, 0, 5, 4, 3, 5, 4, 4, 4,
       1, 3, 2, 4, 7, 0, 3, 4, 3, 4, 7, 1, 1, 5, 2, 2, 1, 6, 2, 5, 2, 7,
       7, 3, 1, 3, 0, 2, 1, 5, 1, 3, 2, 5, 7, 1, 7, 0, 2, 5, 1, 1, 7, 0,
       5, 5, 1, 3, 5, 5, 4, 5, 2, 2, 3, 6, 6, 0, 4, 3, 0, 7, 6, 6, 5, 4,
       1, 2, 0, 1, 1, 5, 2, 4, 0, 1, 7, 5, 7, 5, 0, 5, 3, 6, 0, 3, 3, 3,
       3, 0, 1, 2, 5, 1, 5, 4, 7, 4, 0, 1, 4, 7, 1, 5, 0, 4, 7, 3, 5, 1,
       2, 4, 1, 1, 3, 3, 5, 1, 4, 5, 4, 1, 3, 2, 7, 1, 0, 1, 1, 2, 5, 4,
       5, 5, 1, 5, 4, 4, 1, 1, 4, 7, 1, 1, 4, 7, 4, 7, 4, 1, 5, 1, 3, 3,
       5, 1, 1, 4, 1, 3, 5, 7, 1, 0, 5, 1, 4, 7, 4, 4, 1, 5, 4, 4, 1, 3,
       7, 2, 4, 1, 1, 5, 4, 1, 1, 2, 1, 3, 5, 7, 0, 1, 7, 7, 3, 5, 7, 7,
       5, 5, 1, 6, 5, 1, 4, 4, 2, 1, 5, 2, 4, 5, 4, 0, 4, 1, 3, 7, 1, 5,
       1, 1, 4, 1, 7, 4, 1, 5, 4, 1, 5, 2, 7, 1, 1, 1, 2, 3, 6, 2, 2, 3,
       4, 2, 1, 5, 4, 3, 1, 2, 4, 2, 1, 1, 3, 1, 7, 7, 5, 3, 1, 5, 3, 5,
       1, 0, 5, 1, 7, 2, 0, 5, 3, 7, 2, 0, 6, 5, 1, 6, 6, 2, 2, 6, 0, 0,
       6, 1, 1, 3, 3, 1, 0, 1, 1, 6, 5, 6, 4, 5, 3, 5, 4, 3, 1, 3, 5, 1,
       5, 5, 5, 7, 1, 3, 2, 5, 7, 4, 3, 3, 1, 1, 7, 7, 5, 2, 5, 7, 4, 4,
       1, 1, 5, 2, 4, 5, 3, 5, 3, 1, 7, 7, 1, 5, 4, 7, 1, 1, 4, 4, 1, 4,
       5, 4, 1, 1, 5, 7, 1, 7, 2, 2, 2, 2, 4, 2, 2, 6, 3, 2, 1, 1, 1, 2,
       2, 2, 6, 2, 6, 6, 1, 6, 2, 2, 6, 6, 6, 0, 7, 0, 6, 0, 2])
```

In [16]:
```
df["cluster"]=y_predicted
df.head()
```

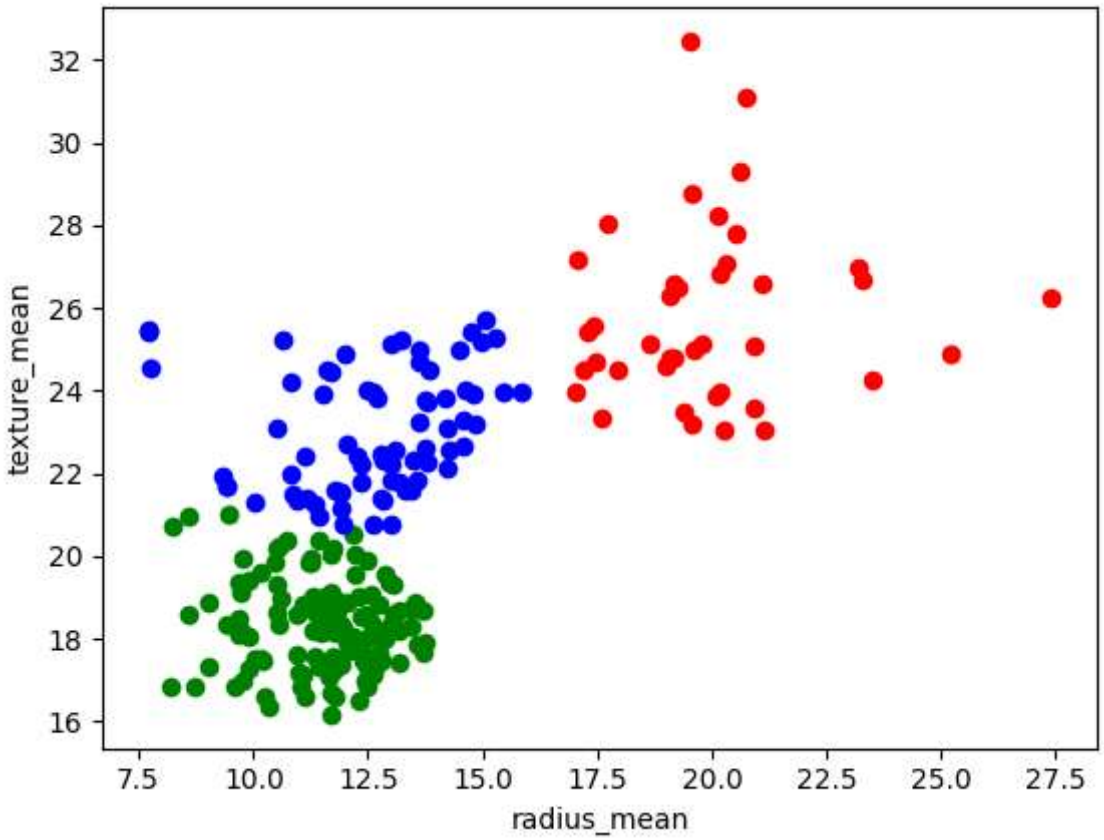
Out[16]:

	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	0.0
2	84300903	M	19.69	21.25	130.00	1203.0	0.1
3	84348301	M	11.42	20.38	77.58	386.1	0.1
4	84358402	M	20.29	14.34	135.10	1297.0	0.1

5 rows × 33 columns

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



In [18]:
```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
scaler.fit(df[["texture_mean"]])
df["texture_mean"]=scaler.transform(df[["texture_mean"]])
df.head()
```

Out[18]:

	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 × 33 columns

In [19]:
```
scaler.fit(df[["radius_mean"]])
df["radius_mean"]=scaler.transform(df[["radius_mean"]])
df.head()
```

Out[19]:

	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 × 33 columns

In [20]:
```
y_predicted=km.fit_predict(df[["radius_mean","texture_mean"]])
y_predicted
```

```
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklearn\cluster\_kmeans.py:870: 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(
```

Out[20]:
```
array([3, 7, 7, 4, 7, 3, 7, 6, 6, 0, 6, 3, 2, 6, 6, 0, 6, 6, 7, 3, 3, 1,
       3, 5, 6, 7, 6, 7, 6, 3, 2, 4, 2, 2, 3, 6, 6, 4, 6, 6, 6, 4, 2, 6,
       6, 7, 1, 4, 1, 6, 4, 3, 4, 7, 6, 4, 7, 6, 4, 1, 1, 4, 6, 1, 0, 6,
       4, 4, 4, 3, 7, 1, 2, 3, 4, 6, 3, 7, 2, 4, 4, 3, 5, 2, 1, 7, 6, 2,
       6, 3, 6, 6, 3, 4, 6, 2, 4, 4, 1, 6, 0, 1, 4, 4, 4, 3, 4, 4, 5, 4,
       4, 4, 6, 4, 1, 4, 1, 3, 6, 7, 1, 7, 5, 3, 3, 3, 0, 7, 3, 2, 1, 6,
       6, 3, 7, 6, 4, 1, 3, 1, 1, 3, 4, 3, 1, 1, 4, 6, 3, 3, 6, 4, 1, 1,
       3, 4, 7, 7, 1, 1, 4, 7, 7, 6, 5, 6, 1, 7, 2, 3, 1, 6, 3, 1, 1, 1,
       4, 6, 6, 3, 5, 2, 6, 1, 6, 1, 7, 4, 4, 3, 6, 6, 4, 0, 6, 3, 6, 7,
       7, 6, 4, 7, 5, 6, 4, 3, 4, 7, 6, 3, 7, 4, 5, 2, 6, 3, 4, 4, 7, 2,
       3, 3, 4, 6, 3, 3, 1, 3, 0, 6, 7, 0, 0, 2, 1, 6, 5, 7, 0, 2, 3, 3,
       4, 6, 2, 4, 3, 3, 0, 1, 2, 4, 7, 7, 7, 3, 2, 3, 6, 0, 2, 2, 7, 6,
       7, 2, 4, 6, 3, 4, 3, 1, 5, 1, 2, 4, 1, 7, 3, 3, 2, 1, 7, 6, 3, 4,
       4, 3, 4, 4, 6, 6, 3, 4, 3, 3, 1, 4, 3, 4, 7, 4, 2, 4, 4, 0, 3, 1,
       3, 3, 4, 3, 3, 1, 4, 4, 1, 7, 4, 4, 1, 7, 3, 7, 1, 4, 3, 4, 6, 6,
       3, 4, 4, 1, 4, 7, 3, 7, 4, 5, 3, 1, 1, 7, 1, 1, 4, 3, 1, 1, 4, 6,
       5, 0, 1, 4, 4, 3, 1, 4, 4, 6, 4, 7, 3, 7, 2, 4, 7, 5, 6, 3, 7, 7,
       3, 3, 4, 0, 3, 4, 1, 1, 6, 4, 3, 6, 1, 3, 1, 2, 1, 1, 6, 5, 4, 3,
       6, 4, 1, 4, 7, 1, 4, 3, 1, 4, 3, 6, 7, 4, 4, 4, 4, 6, 0, 4, 4, 6,
       1, 4, 4, 3, 1, 6, 4, 4, 1, 4, 4, 4, 6, 4, 7, 7, 3, 6, 4, 3, 6, 3,
       4, 2, 3, 4, 7, 0, 2, 3, 6, 7, 4, 2, 0, 3, 4, 0, 0, 0, 0, 0, 2, 5,
       0, 4, 4, 6, 6, 4, 2, 4, 4, 0, 3, 0, 1, 3, 6, 3, 1, 6, 4, 6, 3, 3,
       3, 3, 3, 7, 1, 7, 6, 3, 7, 1, 6, 6, 4, 4, 7, 7, 3, 0, 3, 5, 1, 1,
       4, 4, 3, 6, 1, 3, 6, 3, 6, 4, 7, 7, 4, 3, 1, 5, 4, 6, 1, 1, 4, 1,
       3, 1, 4, 4, 3, 7, 4, 7, 6, 0, 0, 0, 1, 0, 0, 0, 6, 6, 1, 1, 4, 0,
       4, 4, 0, 4, 0, 0, 4, 0, 6, 0, 0, 0, 0, 2, 5, 2, 2, 2, 0])
```

In [25]:
```
df["New Cluster"]=y_predicted
df.head()
```

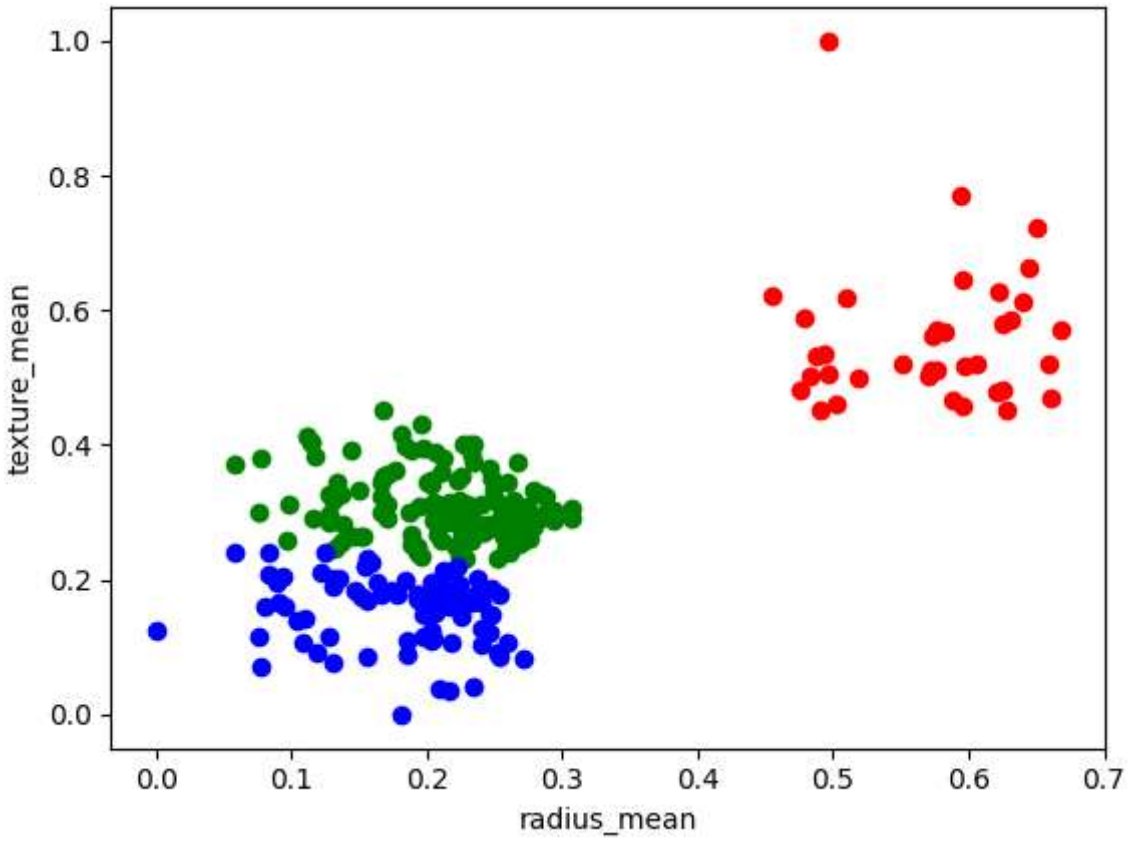
Out[25]:

	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 [26]:
```
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[26]:
```
Text(0, 0.5, 'texture_mean')
```



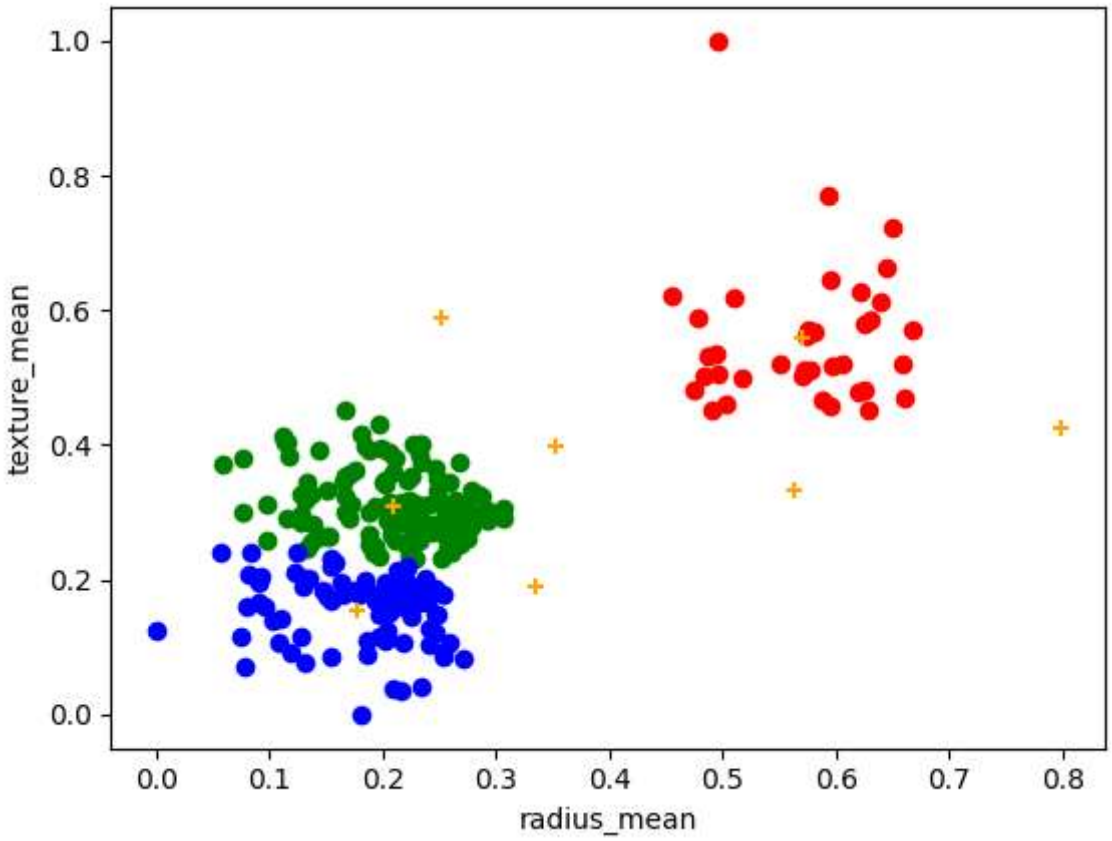
In [27]:
```
km.cluster_centers_
```

Out[27]:
```
array([[0.57132058, 0.55893025],
       [0.20987596, 0.3099295 ],
       [0.17750575, 0.15412045],
       [0.25223338, 0.58802181],
       [0.56287997, 0.33184226],
       [0.35310079, 0.39677038],
       [0.79840767, 0.42469846],
       [0.33570532, 0.19063107]])
```

In [28]:
```
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="oran
plt.xlabel("radius_mean")
plt.ylabel("texture_mean")

```

Out[28]:
```
Text(0, 0.5, 'texture_mean')
```



In [29]:
```
k_rng=range(1,10)
sse=[]
```

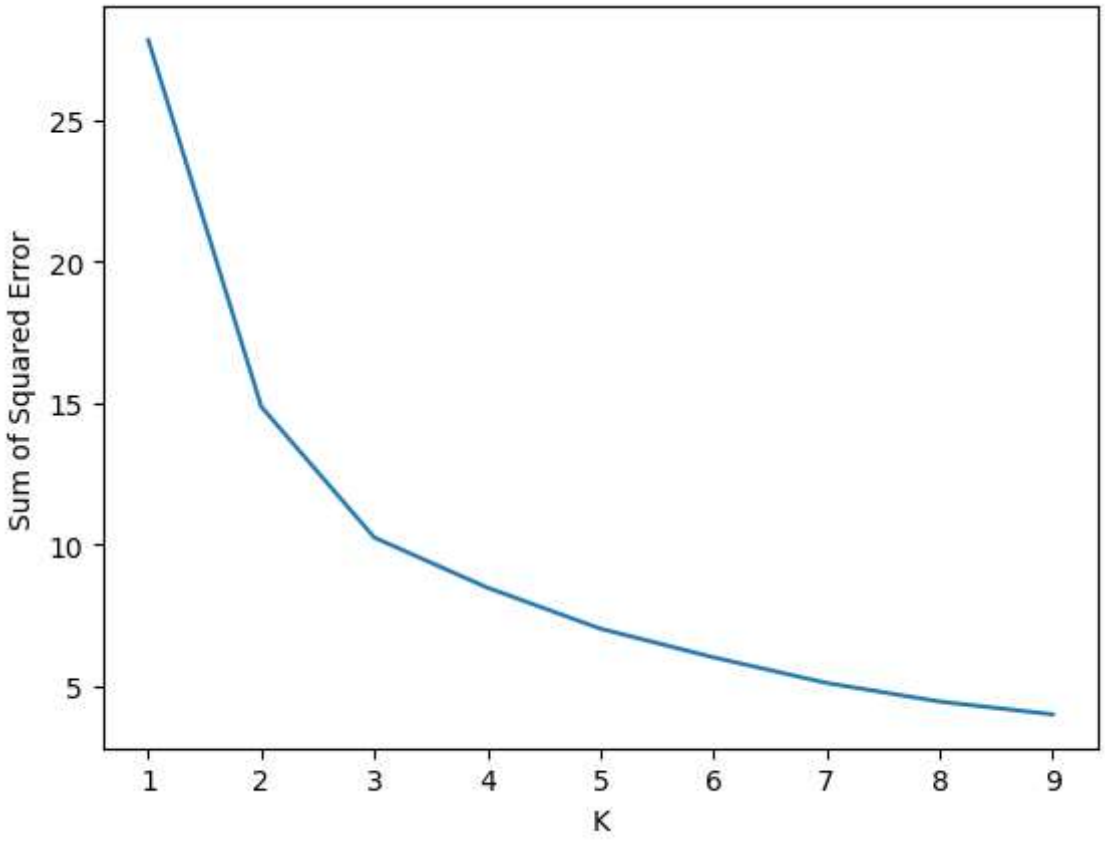
In [30]:

```
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\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will c
hange from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppr
ess the warning
  warnings.warn(
```

```
[27.817507595043075, 14.87203295827117, 10.252751496105198, 8.48472527702760
9, 7.035500433198194, 6.025319673756582, 5.120230748703645, 4.46419846608, 4.
011693711393216]
```

Out[30]: Text(0, 0.5, 'Sum of Squared Error')



CONCLUSION: The KMeans algorithm model is best suited for given dataset

In [ ]:
```
1
```