

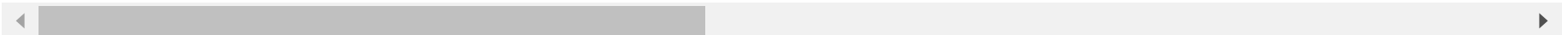
```
In [1]: import pandas as pd
from matplotlib import pyplot as plt
%matplotlib inline
```

```
In [2]: df=pd.read_csv(r"C:\Users\venky\Downloads\BreastCancerPrediction (1).csv")
df
```

Out[2]:

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

569 rows × 33 columns

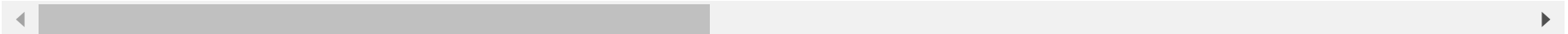


In [3]: `df.head()`

Out[3]:

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

5 rows × 33 columns

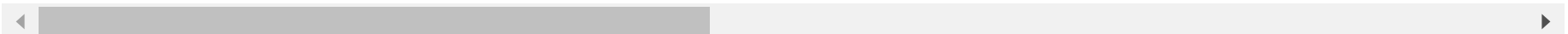


In [4]: `df.tail()`

Out[4]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	co points_
564	926424	M	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0
565	926682	M	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0
566	926954	M	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0
567	927241	M	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0
568	92751	B	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0

5 rows × 33 columns



```
In [5]: df.drop(['Unnamed: 32'],axis=1)
```

```
Out[5]:
```

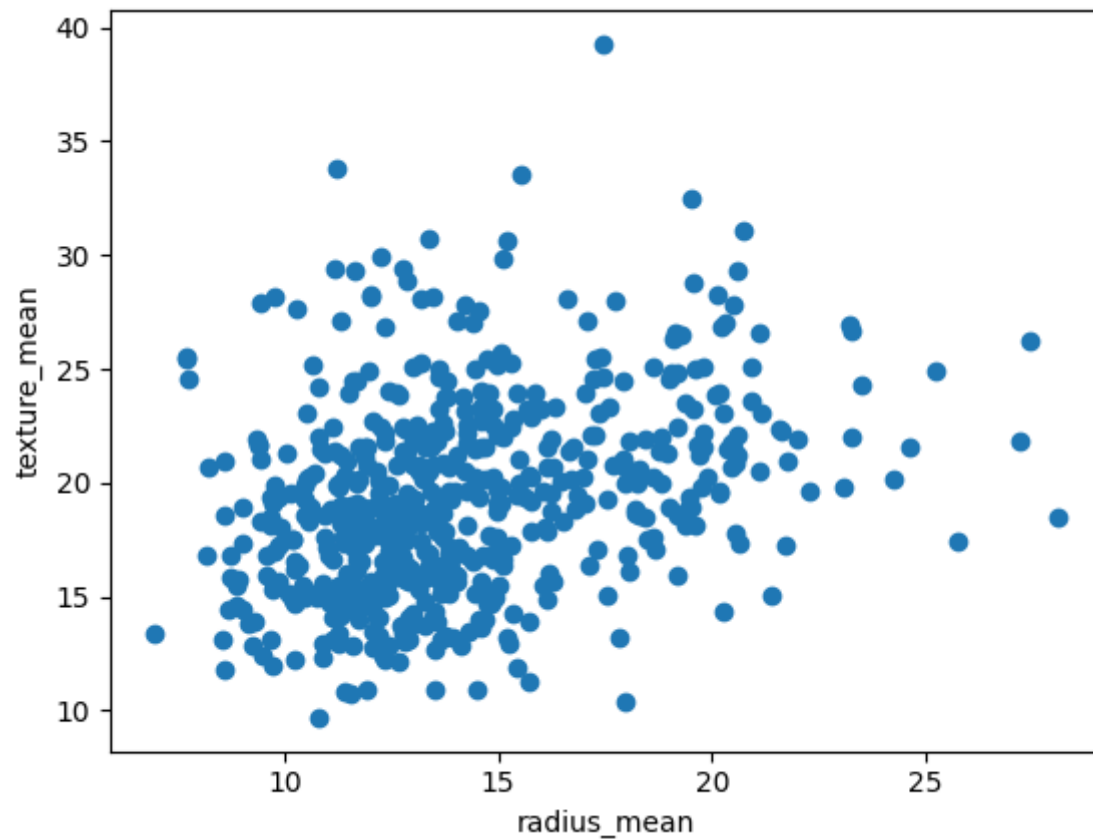
	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	poin
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740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

569 rows × 32 columns



```
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')



```
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\SASIDHAR ROYAL\AppData\Local\Programs\Python\Python311\Lib\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(

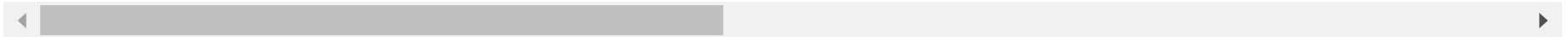
```
Out[8]: array([4, 1, 1, 3, 1, 4, 2, 5, 5, 5, 5, 2, 6, 5, 5, 7, 2, 2, 1, 4, 4, 0,
 4, 1, 2, 4, 5, 1, 5, 4, 6, 3, 6, 6, 2, 2, 5, 3, 5, 5, 5, 5, 6, 3,
 5, 2, 3, 3, 0, 5, 5, 4, 3, 2, 5, 3, 1, 5, 3, 0, 0, 3, 5, 0, 5, 5,
 3, 3, 3, 4, 1, 0, 6, 4, 3, 2, 0, 2, 6, 3, 3, 4, 6, 6, 0, 2, 5, 6,
 5, 4, 5, 5, 4, 3, 2, 1, 3, 3, 0, 2, 5, 0, 3, 3, 3, 4, 3, 3, 1, 5,
 3, 5, 2, 3, 0, 5, 0, 4, 5, 2, 0, 2, 1, 4, 4, 4, 5, 1, 4, 6, 0, 2,
 2, 4, 1, 5, 3, 0, 4, 0, 0, 2, 3, 4, 0, 0, 3, 2, 4, 3, 5, 3, 0, 0,
 4, 3, 2, 2, 0, 0, 3, 1, 1, 5, 1, 2, 0, 2, 6, 4, 0, 3, 4, 0, 0, 0,
 3, 2, 5, 0, 1, 6, 2, 0, 5, 0, 2, 3, 3, 4, 5, 5, 3, 7, 5, 4, 5, 2,
 1, 2, 3, 2, 6, 5, 3, 4, 3, 2, 5, 4, 1, 3, 1, 6, 5, 4, 3, 3, 1, 6,
 4, 4, 3, 2, 4, 4, 0, 4, 5, 5, 2, 7, 7, 6, 0, 5, 6, 1, 7, 7, 4, 0,
 3, 5, 6, 3, 3, 4, 5, 0, 6, 3, 1, 2, 1, 4, 6, 4, 5, 7, 6, 2, 2, 2,
 2, 6, 3, 5, 4, 3, 4, 0, 1, 0, 6, 3, 0, 1, 3, 4, 6, 0, 1, 2, 4, 3,
 3, 0, 3, 3, 2, 2, 4, 3, 0, 4, 0, 3, 2, 5, 1, 3, 6, 3, 3, 5, 4, 0,
 4, 4, 3, 4, 0, 0, 3, 3, 0, 2, 3, 3, 0, 1, 0, 1, 0, 3, 4, 3, 2, 2,
 4, 3, 3, 0, 3, 2, 4, 1, 3, 6, 4, 3, 0, 1, 0, 0, 3, 4, 0, 0, 3, 2,
 1, 5, 0, 3, 3, 4, 0, 3, 3, 5, 3, 2, 4, 1, 6, 3, 1, 1, 5, 4, 1, 1,
 4, 4, 3, 7, 4, 3, 0, 0, 5, 3, 4, 5, 0, 4, 0, 6, 0, 3, 2, 1, 3, 4,
 3, 3, 0, 3, 2, 0, 3, 4, 0, 3, 4, 5, 2, 3, 3, 3, 3, 5, 7, 5, 3, 2,
 0, 5, 3, 4, 0, 3, 3, 3, 0, 5, 3, 3, 5, 3, 1, 1, 4, 2, 3, 4, 3, 4,
 3, 6, 4, 3, 2, 5, 6, 4, 2, 1, 5, 6, 7, 4, 3, 7, 7, 5, 5, 7, 6, 6,
 7, 3, 3, 3, 5, 3, 6, 3, 3, 7, 4, 7, 0, 4, 2, 4, 0, 2, 3, 3, 4, 3,
 4, 4, 4, 1, 0, 2, 5, 4, 2, 0, 5, 2, 3, 3, 2, 1, 4, 5, 4, 1, 0, 0,
 3, 3, 4, 5, 0, 4, 5, 4, 2, 3, 2, 1, 3, 4, 0, 1, 3, 3, 0, 0, 3, 0,
 4, 0, 3, 3, 4, 1, 3, 1, 5, 5, 5, 5, 0, 5, 5, 7, 5, 5, 0, 3, 3, 5,
 5, 5, 7, 5, 7, 7, 3, 7, 5, 5, 7, 7, 7, 6, 1, 6, 6, 6, 5])
```

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

Out[9]:

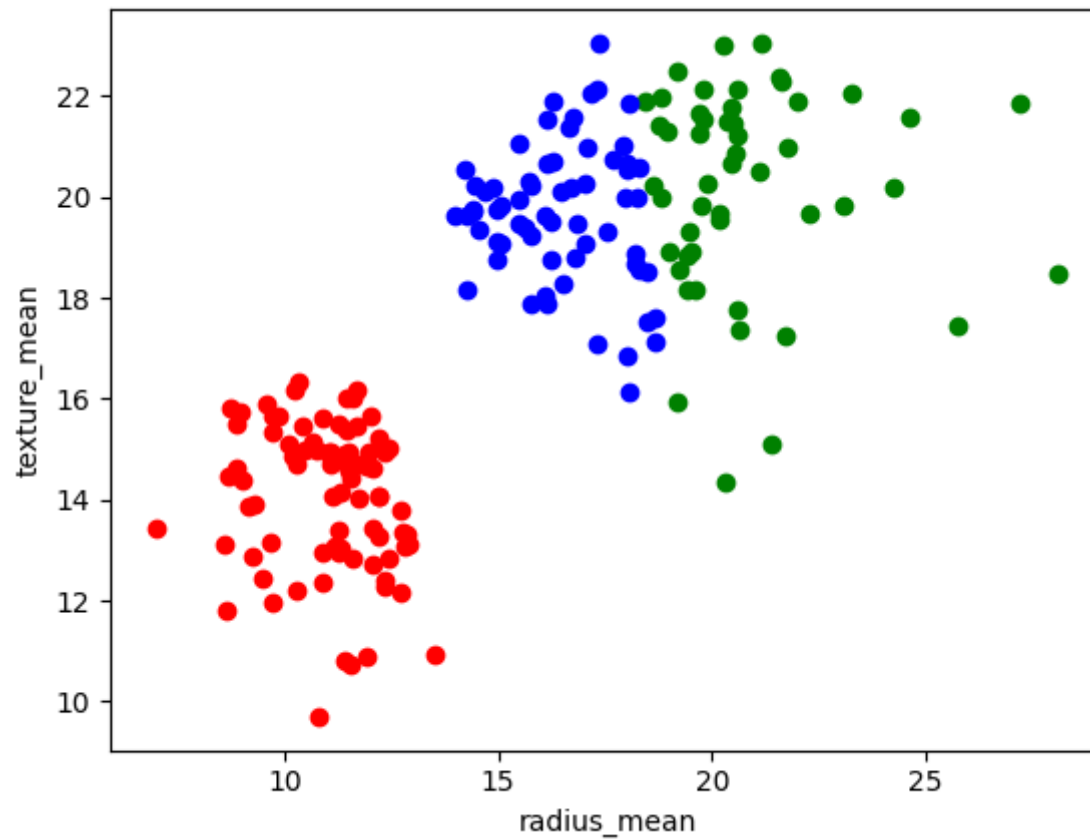
	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	co points_
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990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

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')
```

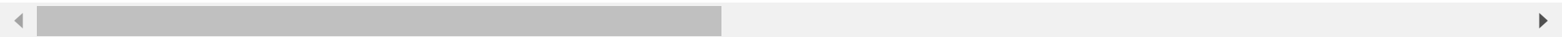


```
In [11]: from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
scaler.fit(df[["texture_mean"]])
df["texture_mean"]=scaler.transform(df[["texture_mean"]])
df.head()
```

Out[11]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	co points_
0	842302	M	17.99	0.022658	122.80	1001.0	0.11840	0.27760	0.3001	0
1	842517	M	20.57	0.272574	132.90	1326.0	0.08474	0.07864	0.0869	0
2	84300903	M	19.69	0.390260	130.00	1203.0	0.10960	0.15990	0.1974	0
3	84348301	M	11.42	0.360839	77.58	386.1	0.14250	0.28390	0.2414	0
4	84358402	M	20.29	0.156578	135.10	1297.0	0.10030	0.13280	0.1980	0

5 rows × 34 columns

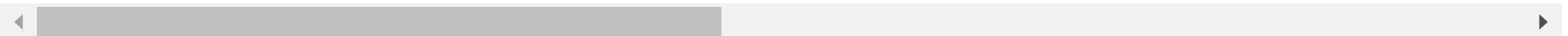


```
In [12]: scaler.fit(df[["radius_mean"]])
df["radius_mean"]=scaler.transform(df[["radius_mean"]])
df.head()
```

Out[12]:

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

5 rows × 34 columns




```
In [13]: y_predicted=km.fit_predict(df[["radius_mean","texture_mean"]])
y_predicted
```

C:\Users\SASIDHAR ROYAL\AppData\Local\Programs\Python\Python311\Lib\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[13]: array([2, 3, 3, 0, 3, 2, 3, 7, 7, 4, 7, 2, 6, 7, 7, 4, 7, 7, 3, 2, 2, 5,
 2, 1, 7, 3, 7, 3, 7, 3, 6, 0, 6, 6, 2, 7, 7, 0, 4, 7, 7, 0, 6, 7,
 7, 3, 5, 0, 5, 7, 0, 2, 0, 3, 7, 0, 3, 7, 0, 5, 5, 0, 7, 5, 4, 7,
 0, 0, 0, 2, 3, 5, 6, 2, 2, 7, 2, 3, 6, 0, 0, 2, 1, 6, 5, 3, 7, 6,
 7, 2, 7, 7, 2, 0, 7, 6, 0, 0, 5, 7, 4, 5, 0, 0, 0, 2, 0, 0, 1, 0,
 0, 0, 7, 0, 5, 0, 5, 2, 7, 3, 5, 3, 1, 2, 2, 2, 4, 3, 2, 6, 5, 7,
 7, 2, 3, 7, 0, 5, 2, 5, 5, 7, 0, 2, 5, 5, 0, 7, 2, 2, 7, 0, 5, 5,
 2, 0, 3, 3, 5, 5, 0, 3, 3, 7, 1, 7, 5, 3, 6, 2, 5, 7, 2, 5, 5, 5,
 0, 7, 7, 2, 1, 6, 7, 5, 7, 5, 3, 0, 0, 2, 7, 7, 0, 4, 7, 2, 7, 3,
 3, 7, 0, 3, 1, 7, 0, 2, 0, 3, 7, 2, 3, 0, 1, 6, 7, 2, 0, 0, 3, 6,
 2, 2, 0, 7, 2, 2, 5, 2, 4, 7, 3, 4, 4, 6, 5, 7, 1, 3, 4, 6, 2, 2,
 0, 7, 6, 0, 2, 2, 4, 5, 6, 0, 3, 3, 3, 2, 6, 2, 7, 4, 6, 6, 3, 7,
 3, 6, 0, 7, 2, 0, 2, 5, 1, 5, 6, 0, 5, 3, 2, 2, 6, 5, 3, 7, 2, 0,
 0, 2, 0, 0, 7, 7, 2, 0, 2, 2, 5, 0, 2, 0, 3, 0, 6, 0, 0, 4, 2, 5,
 2, 2, 0, 2, 2, 5, 0, 0, 5, 3, 0, 0, 5, 3, 2, 3, 5, 0, 2, 0, 7, 7,
 2, 0, 0, 5, 0, 3, 2, 3, 0, 1, 2, 5, 5, 3, 5, 5, 0, 2, 5, 5, 0, 7,
 1, 4, 5, 0, 0, 2, 5, 0, 0, 7, 0, 3, 2, 3, 6, 0, 3, 1, 7, 2, 3, 3,
 2, 2, 0, 4, 2, 0, 5, 5, 7, 0, 2, 7, 5, 2, 5, 6, 5, 5, 7, 1, 0, 2,
 7, 0, 5, 0, 3, 5, 0, 2, 2, 0, 2, 7, 3, 0, 0, 0, 0, 7, 4, 0, 0, 7,
 5, 0, 0, 2, 5, 7, 0, 0, 5, 0, 0, 0, 7, 0, 3, 3, 2, 7, 0, 2, 7, 2,
 0, 6, 2, 0, 3, 4, 6, 2, 7, 3, 0, 6, 4, 2, 0, 4, 4, 4, 4, 6, 1,
 4, 0, 0, 7, 7, 0, 6, 0, 0, 4, 2, 4, 5, 2, 7, 2, 5, 7, 0, 7, 2, 2,
 2, 2, 2, 3, 5, 3, 7, 2, 3, 5, 7, 7, 0, 0, 3, 3, 2, 4, 2, 1, 5, 5,
 0, 0, 2, 7, 5, 2, 7, 2, 7, 0, 3, 3, 0, 2, 5, 1, 0, 7, 5, 5, 7, 5,
 2, 5, 0, 0, 2, 3, 0, 3, 7, 4, 4, 4, 5, 4, 4, 4, 7, 7, 5, 5, 0, 4,
 0, 0, 4, 0, 4, 4, 0, 4, 7, 4, 4, 4, 4, 6, 1, 6, 6, 6, 4])
```

```
In [14]: df["New Cluster"]=y_predicted  
df.head()
```

Out[14]:

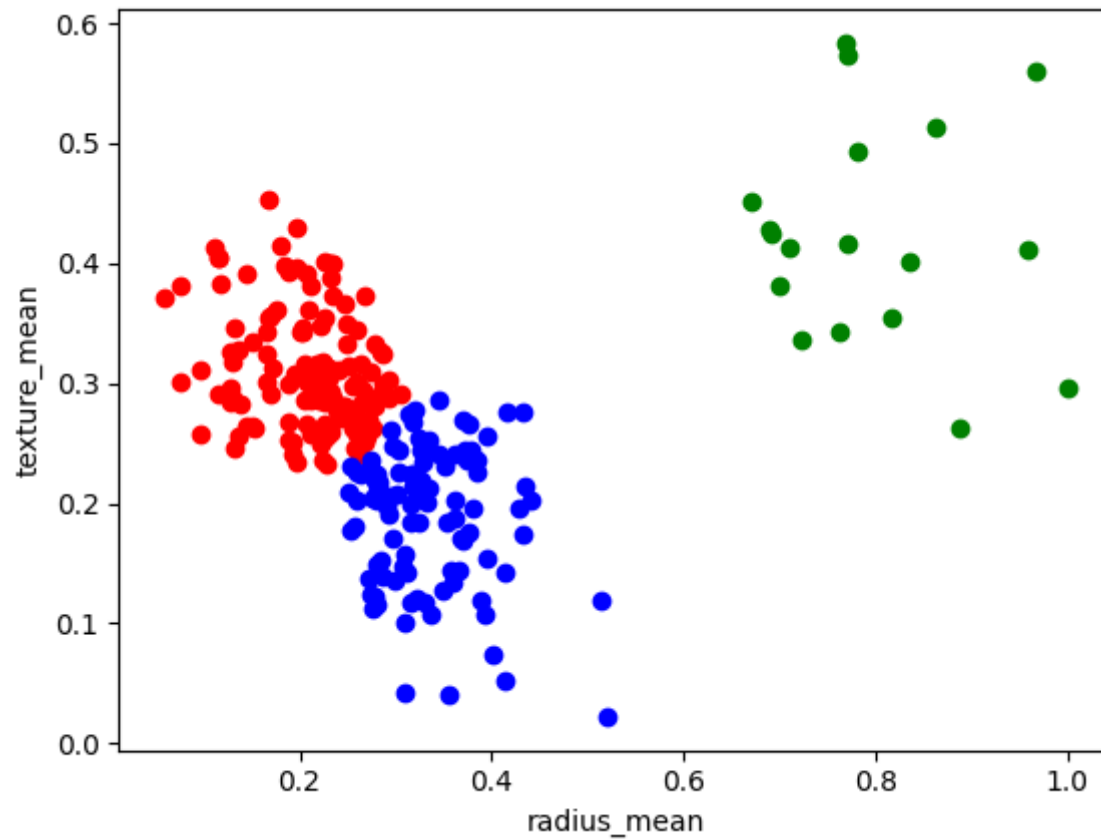
	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	co points_
0	842302	M	0.521037	0.022658	122.80	1001.0	0.11840	0.27760	0.3001	0
1	842517	M	0.643144	0.272574	132.90	1326.0	0.08474	0.07864	0.0869	0
2	84300903	M	0.601496	0.390260	130.00	1203.0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

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')
```

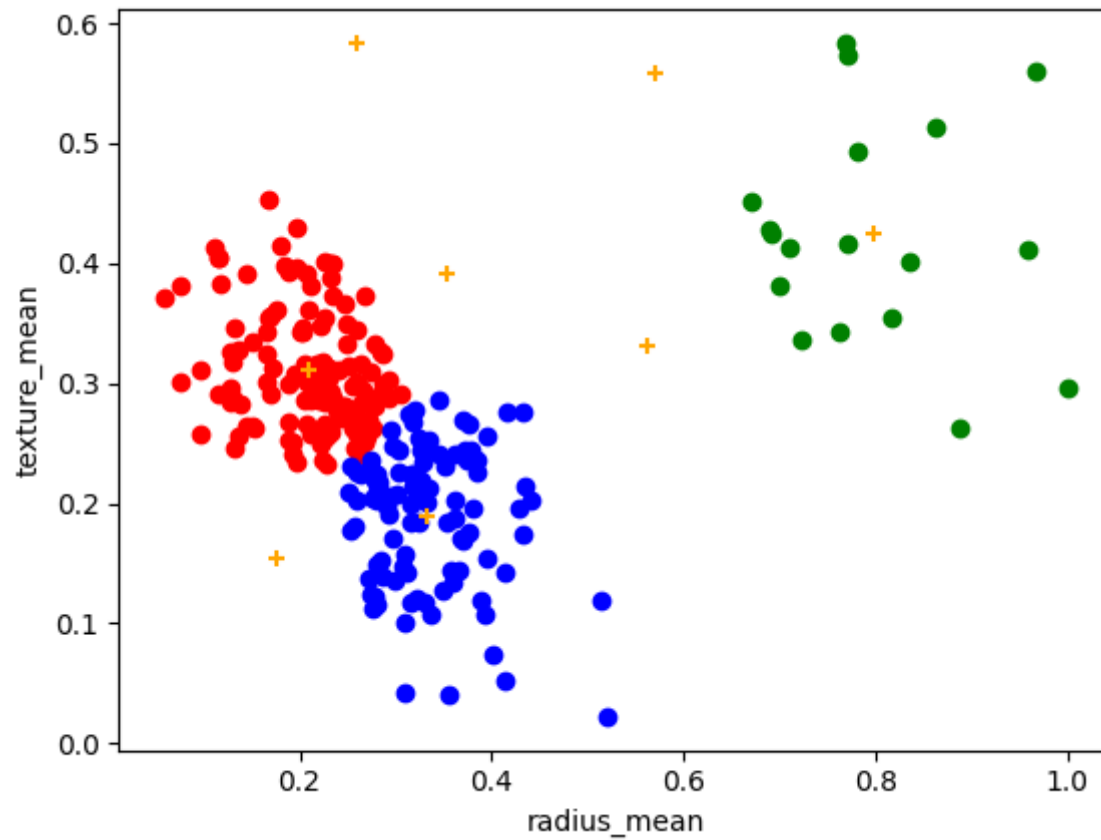


```
In [16]: km.cluster_centers_
```

```
Out[16]: array([[0.20878924, 0.31058452],  
                [0.79840767, 0.42469846],  
                [0.3331624 , 0.18999839],  
                [0.56287997, 0.33184226],  
                [0.2590623 , 0.58293879],  
                [0.17652977, 0.15382448],  
                [0.57132058, 0.55893025],  
                [0.3534653 , 0.39091896]])
```

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

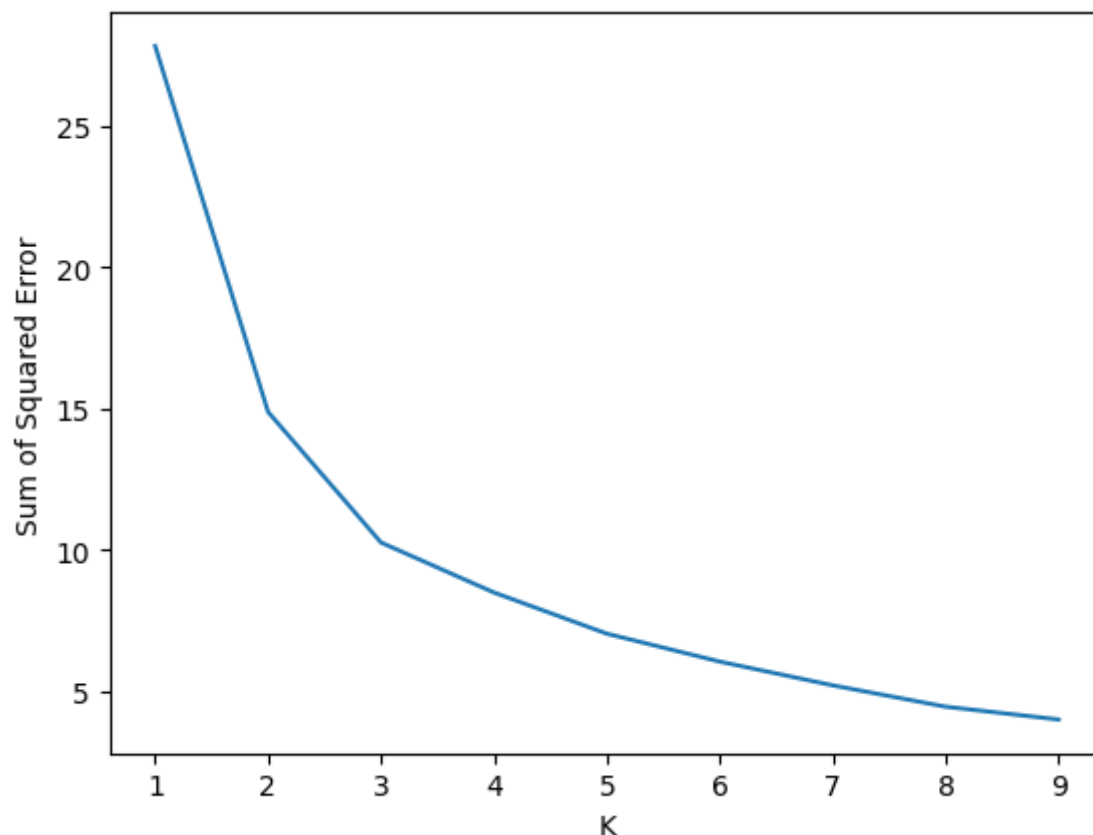
```
In [19]: for k in k_rng:
          km=KMeans(n_clusters=k)
          km.fit(df[["radius_mean","texture_mean"]])
          sse.append(km.inertia_)
```



```
In [20]: print(sse)
plt.plot(k_rng,sse)
plt.xlabel("K")
plt.ylabel("Sum of Squared Error")
```

```
[27.817507595043075, 14.87203295827117, 10.252751496105198, 8.484725277027607, 7.027303957640527, 6.039305768835715, 5.199953930194845, 4.44439527370828, 3.9915411403216825]
```

```
Out[20]: Text(0, 0.5, 'Sum of Squared Error')
```



Conclusion:- In Above DataSet we can use any models to get different accuracies. But by using clustering technique we can get best accuracy

for the Dataset. Therefore we can conclude that breast Cancer prediction DataSet is best fit for "k-Means clustering Model"

In []: