Problem Statement

In [ ]:
```
Brest cancer prediction based on respective features.
```

In [2]:
```python
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import LogisticRegression
```

In [3]:
```python
df=pd.read_csv(r"C:\Users\pappu\Downloads\BreastCancerPrediction.csv")
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 cleaning

In [4]:

```
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	0.10
3	84348301	M	11.42	20.38	77.58	386.1	0.14
4	84358402	M	20.29	14.34	135.10	1297.0	0.10

5 rows × 33 columns

In [5]:

```
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]:

```
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 [7]:
```
df.tail()
```

Out[7]:

	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	0.0
566	926954	M	16.60	28.08	108.30	858.1	0.0
567	927241	M	20.60	29.33	140.10	1265.0	0.1
568	92751	B	7.76	24.54	47.92	181.0	0.0

5 rows × 33 columns

In [8]:
```
df.drop(['Unnamed: 32'],axis=1)
```

Out[8]:

	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

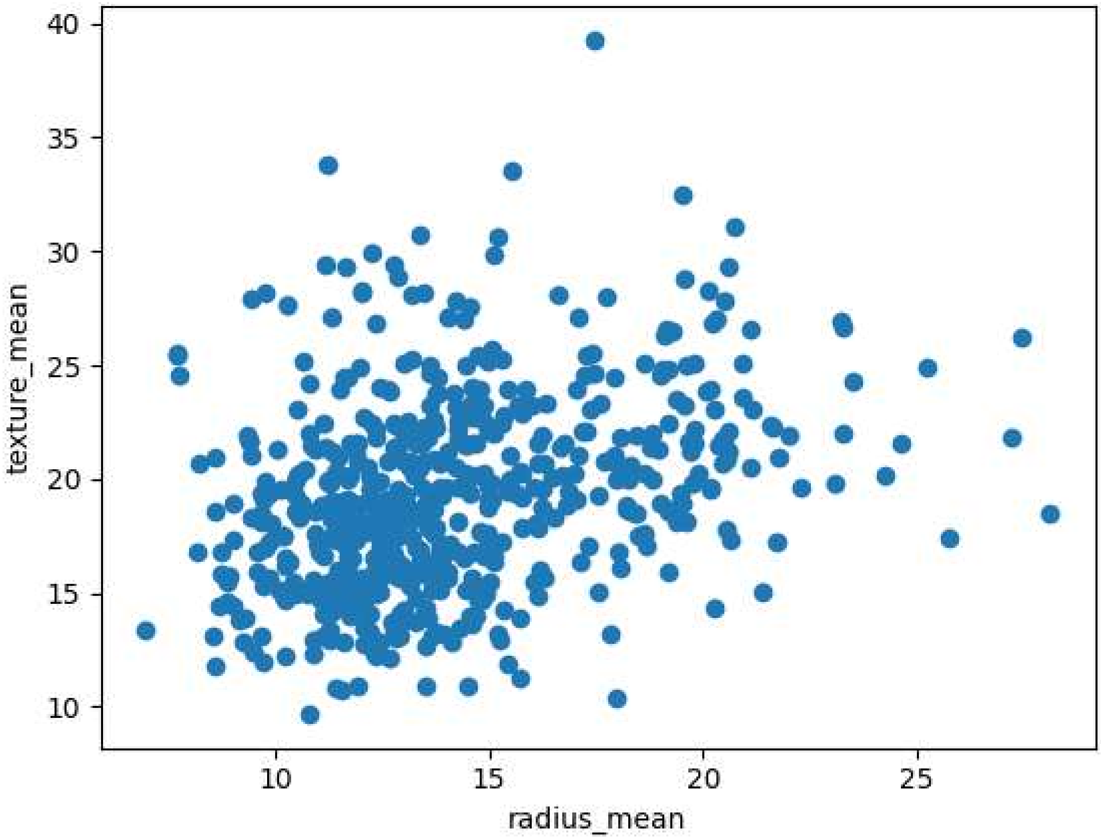
569 rows × 32 columns

In [9]:

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

Out[9]:

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



In [10]:

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

Out[10]:

```
KMeans()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [11]:

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

```
C:\Users\teppa\AppData\Local\Programs\Python\Python310\lib\site-packages\skle
arn\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 supp
ress the warning
  warnings.warn(
```

Out[11]:

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

In [12]:

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

Out[12]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_m
0	842302	M	17.99	10.38	122.80	1001.0	0.11
1	842517	M	20.57	17.77	132.90	1326.0	0.08
2	84300903	M	19.69	21.25	130.00	1203.0	0.10
3	84348301	M	11.42	20.38	77.58	386.1	0.14
4	84358402	M	20.29	14.34	135.10	1297.0	0.10

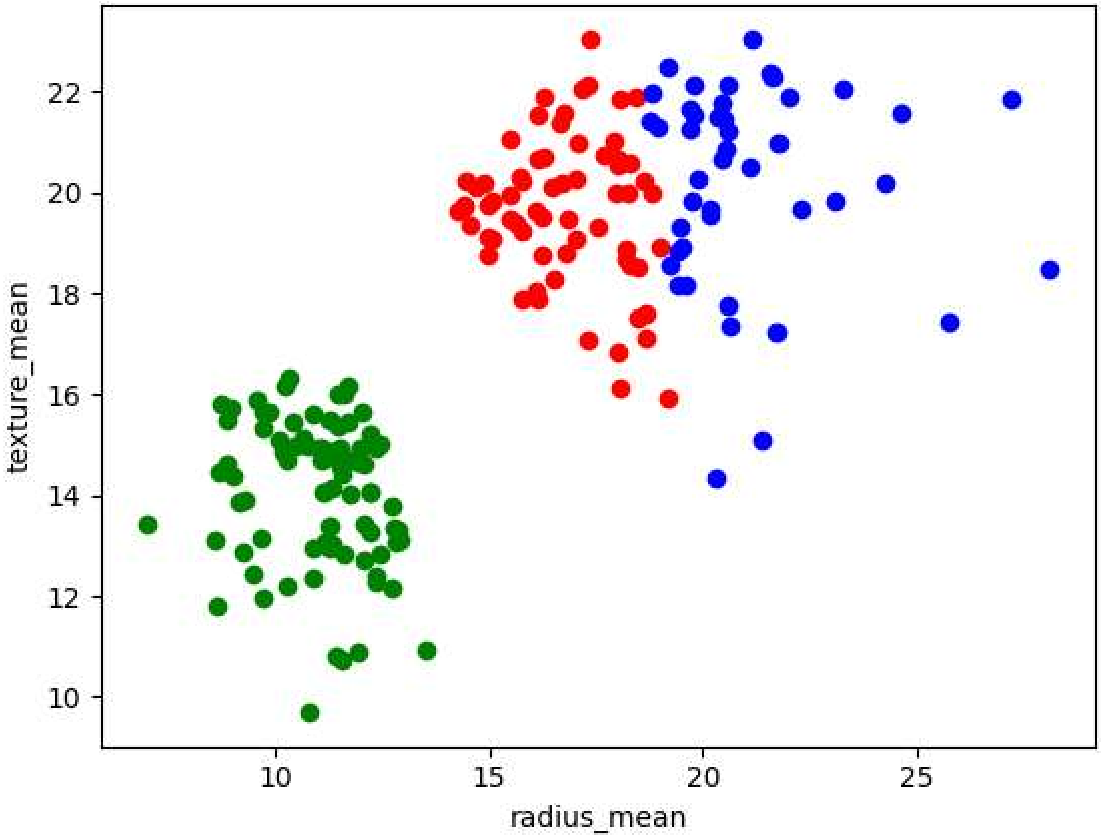
5 rows × 34 columns

In [13]:

```
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[13]:

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



In [14]:

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

Out[14]:

	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	0.08
2	84300903	M	19.69	0.390260	130.00	1203.0	0.10
3	84348301	M	11.42	0.360839	77.58	386.1	0.14
4	84358402	M	20.29	0.156578	135.10	1297.0	0.10

5 rows × 34 columns

In [15]:

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

Out[15]:

	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	0.08
2	84300903	M	0.601496	0.390260	130.00	1203.0	0.10
3	84348301	M	0.210090	0.360839	77.58	386.1	0.14
4	84358402	M	0.629893	0.156578	135.10	1297.0	0.10

5 rows × 34 columns

In [16]:

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

```
C:\Users\teppa\AppData\Local\Programs\Python\Python310\lib\site-packages\skle
arn\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 supp
ress the warning
  warnings.warn(
```

Out[16]:

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

In [17]:

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

Out[17]:

	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

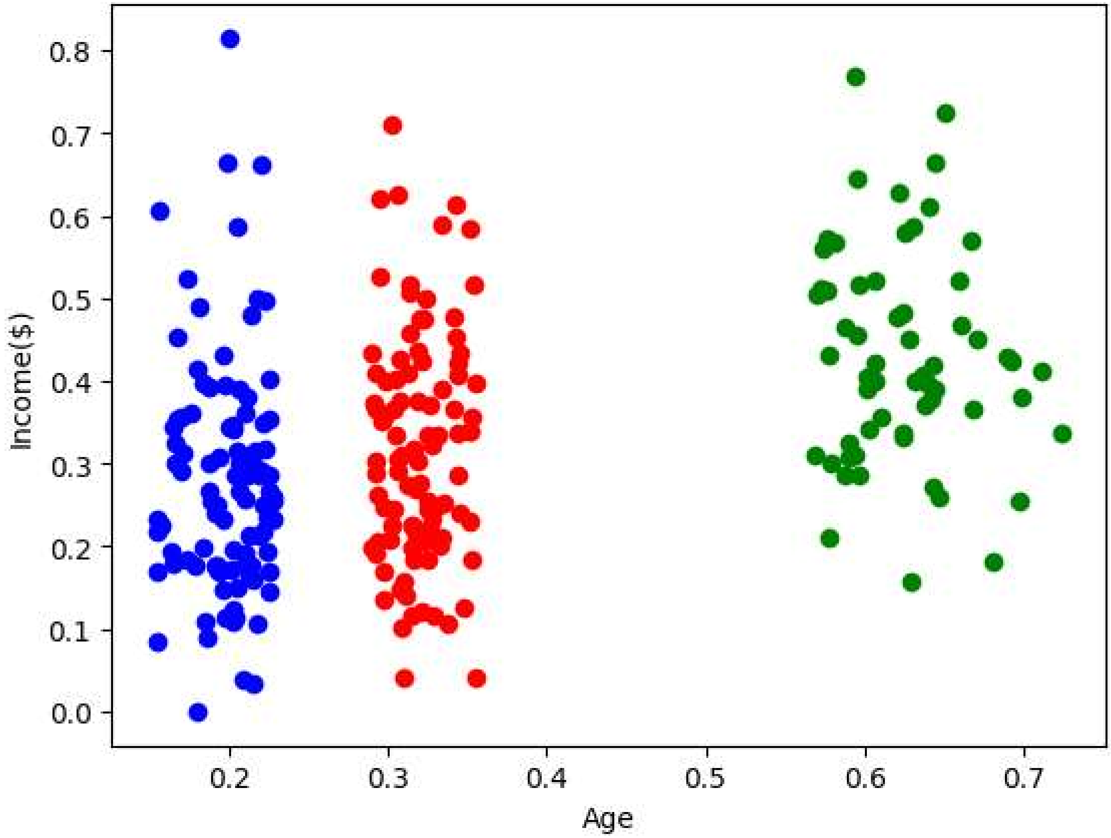
5 rows × 35 columns

In [18]:

```
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("Age")
plt.ylabel("Income($)")
```

Out[18]:

```
Text(0, 0.5, 'Income($)')
```



In [19]:

```
KM.cluster_centers_
```

Out[19]:

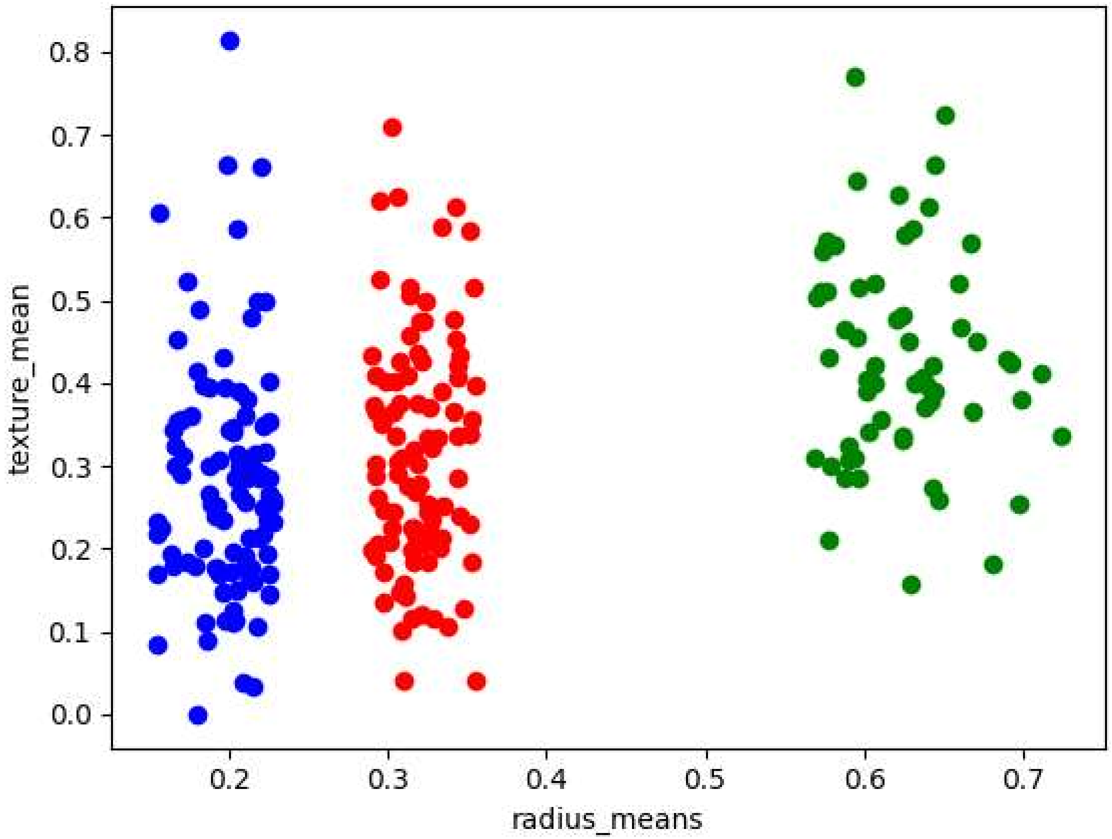
```
array([[0.31980516],
       [0.62451291],
       [0.19885465],
       [0.50980606],
       [0.84866771],
       [0.25974676],
       [0.10853522],
       [0.39468815]])
```

In [21]:

```
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_means")
plt.ylabel("texture_mean")
```

Out[21]:

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



In [23]:

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

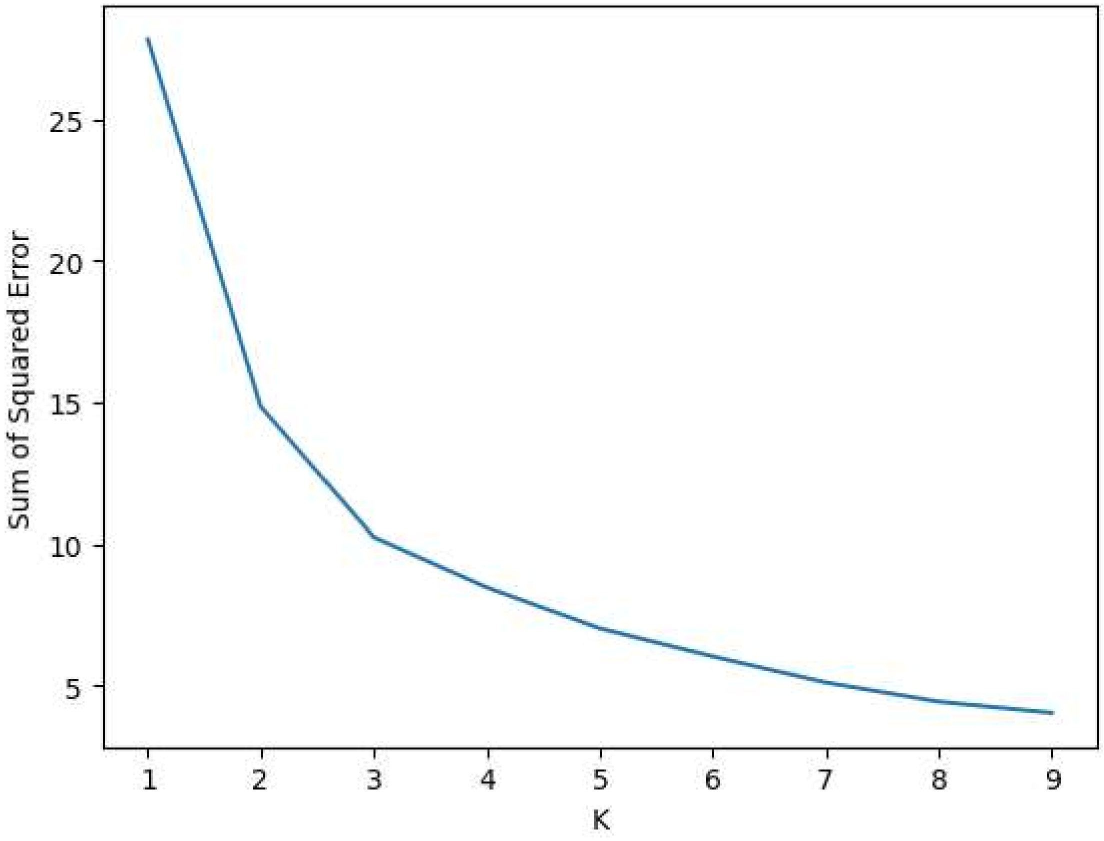
In [31]:

```
for K in k_rang:
    KM=KMeans(n_clusters=K)
    KM.fit(df[["radius_mean","texture_mean"]])
    sse.append(KM.inertia_)
print(sse)
plt.plot(k_rang,sse)
plt.xlabel("K")
plt.ylabel("Sum of Squared Error")

```

```
C:\Users\teppa\AppData\Local\Programs\Python\Python310\lib\site-packages\skle
arn\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 supp
ress the warning
  warnings.warn(
C:\Users\teppa\AppData\Local\Programs\Python\Python310\lib\site-packages\skle
arn\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 supp
ress the warning
  warnings.warn(
C:\Users\teppa\AppData\Local\Programs\Python\Python310\lib\site-packages\skle
arn\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 supp
ress the warning
  warnings.warn(
C:\Users\teppa\AppData\Local\Programs\Python\Python310\lib\site-packages\skle
arn\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 supp
ress the warning
  warnings.warn(
C:\Users\teppa\AppData\Local\Programs\Python\Python310\lib\site-packages\skle
arn\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 supp
ress the warning
  warnings.warn(
C:\Users\teppa\AppData\Local\Programs\Python\Python310\lib\site-packages\skle
arn\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 supp
ress the warning
  warnings.warn(
C:\Users\teppa\AppData\Local\Programs\Python\Python310\lib\site-packages\skle
arn\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 supp
ress the warning
  warnings.warn(
C:\Users\teppa\AppData\Local\Programs\Python\Python310\lib\site-packages\skle
arn\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 supp
ress the warning
  warnings.warn(

[27.81750759504307, 14.87203295827117, 10.252751496105196, 8.487131283091337,
7.0342608118317775, 6.04576993365012, 5.120183612235493, 4.444287355881231,
4.046574367624161]
```

```
C:\Users\teppa\AppData\Local\Programs\Python\Python310\lib\site-packages\skle
arn\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 supp
ress the warning
  warnings.warn(
```

```
Out[31]: Text(0, 0.5, 'Sum of Squared Error')
```



conclusion

for the given data all model are not predicted which is best fit but by uaing clustering k_Means Clustering is fitted

```
In [ ]:
```