PROBLEM STATEMENT : Predictive study using the breast cancer diagnostic dataset

Importing Libraries

In [2]:

```
import numpy as np
import pandas as pd
from sklearn import preprocessing
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(style="white")#white background for seaborn plots
sns.set(style="whitegrid",color_codes=True)
import warnings
warnings.simplefilter(action="ignore")
```

Reading the data

In [3]:

```
df=pd.read_csv(r"C:\Users\G S R  KARTHIK\Documents\BreastCancerPrediction.csv")
df
```

Out[3]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothn
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

Data Cleaning and Preprocessing

In [4]:

```
df.head(10)
```

Out[4]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothnes
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	843786	M	12.45	15.70	82.57	477.1	
6	844359	M	18.25	19.98	119.60	1040.0	
7	84458202	M	13.71	20.83	90.20	577.9	
8	844981	M	13.00	21.82	87.50	519.8	
9	84501001	M	12.46	24.04	83.97	475.9	

10 rows × 33 columns

In [5]:

```
df.tail()
```

Out[5]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothnes
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 [6]:

```
df.shape
```

Out[6]:

```
(569, 33)
```

In [7]:

```
df.describe()
```

Out[7]:

	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 [8]:

```
df.columns
```

Out[8]:

```
Index(['id', 'diagnosis', 'radius_mean', 'texture_mean', 'perimeter_mean',
       'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mea
n',
       'concave points_mean', 'symmetry_mean', 'fractal_dimension_mean',
       'radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_s
e',
       'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_s
e',
       '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 [9]:

```
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 [10]:

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

Out[10]:

```
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 [11]:

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

In [12]:

```
df
```

Out[12]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothn
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 [13]:

```
df["diagnosis"].value_counts()
```

Out[13]:

```
diagnosis
B    357
M    212
Name: count, dtype: int64
```

In [14]:

```
diagnosis={"diagnosis":{"B":0,"M":1}}
df=df.replace(diagnosis)
df
```

Out[14]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothn
0	842302	1	17.99	10.38	122.80	1001.0	
1	842517	1	20.57	17.77	132.90	1326.0	
2	84300903	1	19.69	21.25	130.00	1203.0	
3	84348301	1	11.42	20.38	77.58	386.1	
4	84358402	1	20.29	14.34	135.10	1297.0	
...	...	...	...	...	...	...	
564	926424	1	21.56	22.39	142.00	1479.0	
565	926682	1	20.13	28.25	131.20	1261.0	
566	926954	1	16.60	28.08	108.30	858.1	
567	927241	1	20.60	29.33	140.10	1265.0	
568	92751	0	7.76	24.54	47.92	181.0	

569 rows × 32 columns

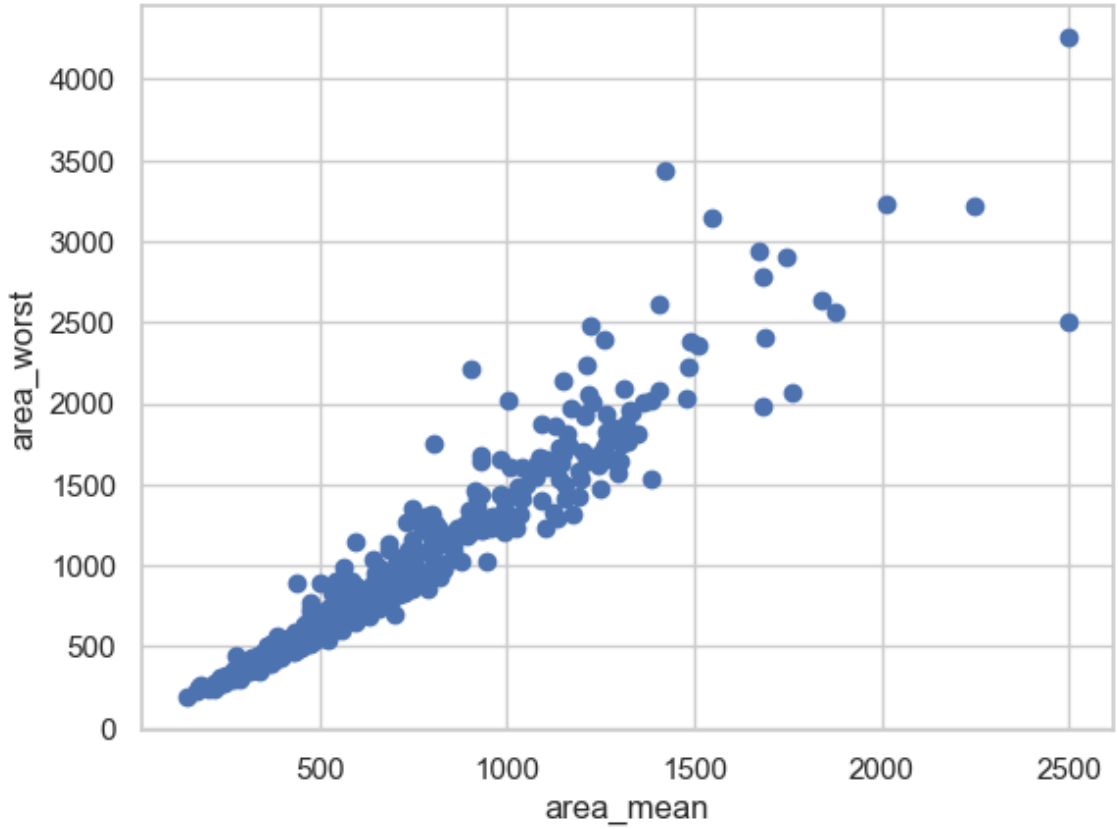
Data Visualization

In [16]:

```
plt.scatter(df["area_mean"],df["area_worst"])
plt.xlabel("area_mean")
plt.ylabel("area_worst")
```

Out[16]:

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



Importing KMeans

In [17]:

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

Out[17]:

```
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 [18]:

```
y_predicted=km.fit_predict(df[["area_mean","area_worst"]])
y_predicted
```

Out[18]:

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

In [19]:

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

Out[19]:

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

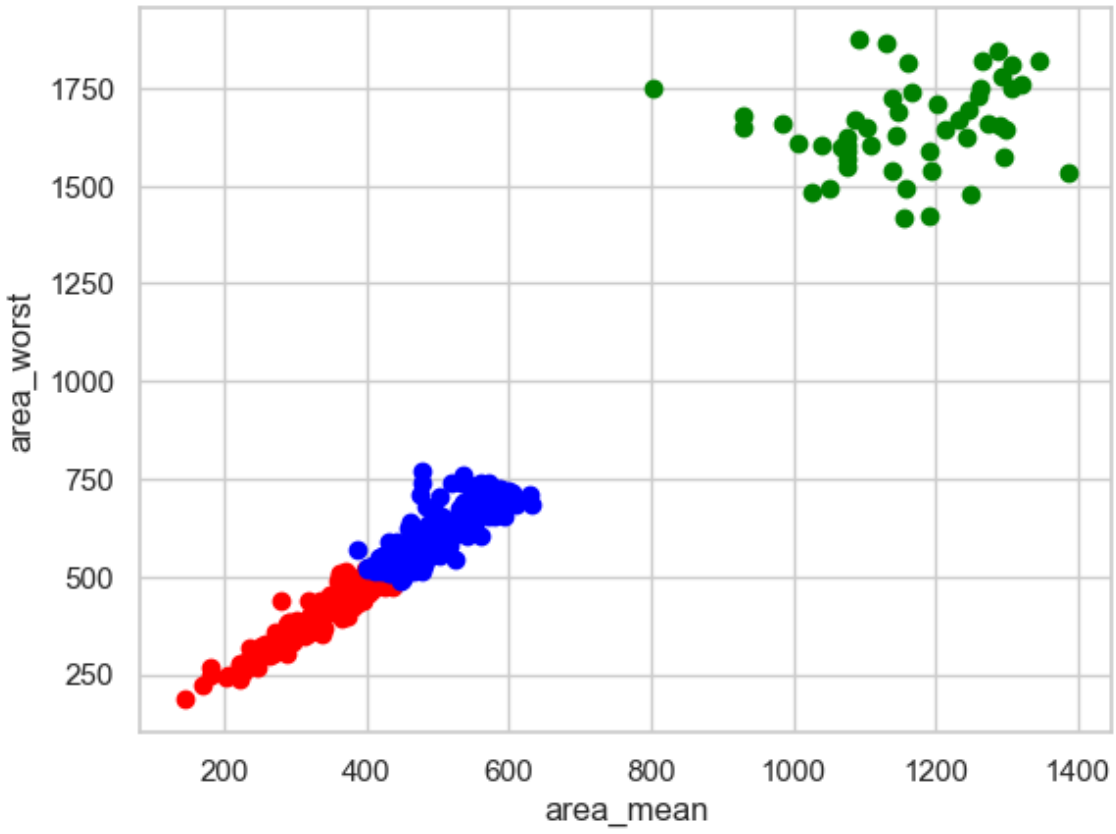
5 rows × 33 columns

In [20]:

```
df1=df[df.cluster==0]
df2=df[df.cluster==1]
df3=df[df.cluster==2]
plt.scatter(df1["area_mean"],df1["area_worst"],color="red")
plt.scatter(df2["area_mean"],df2["area_worst"],color="green")
plt.scatter(df3["area_mean"],df3["area_worst"],color="blue")
plt.xlabel("area_mean")
plt.ylabel("area_worst")
```

Out[20]:

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



In [21]:

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

Out[21]:

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

5 rows × 33 columns

In [22]:

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

Out[22]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothnes
0	842302	1	17.99	10.38	122.80	0.363733	
1	842517	1	20.57	17.77	132.90	0.501591	
2	84300903	1	19.69	21.25	130.00	0.449417	
3	84348301	1	11.42	20.38	77.58	0.102906	
4	84358402	1	20.29	14.34	135.10	0.489290	

5 rows × 33 columns

In [23]:

```
y_predicted=km.fit_predict(df[["area_mean","area_worst"]])
y_predicted
```

Out[23]:

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

In [24]:

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

Out[24]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothnes
0	842302	1	17.99	10.38	122.80	0.363733	
1	842517	1	20.57	17.77	132.90	0.501591	
2	84300903	1	19.69	21.25	130.00	0.449417	
3	84348301	1	11.42	20.38	77.58	0.102906	
4	84358402	1	20.29	14.34	135.10	0.489290	

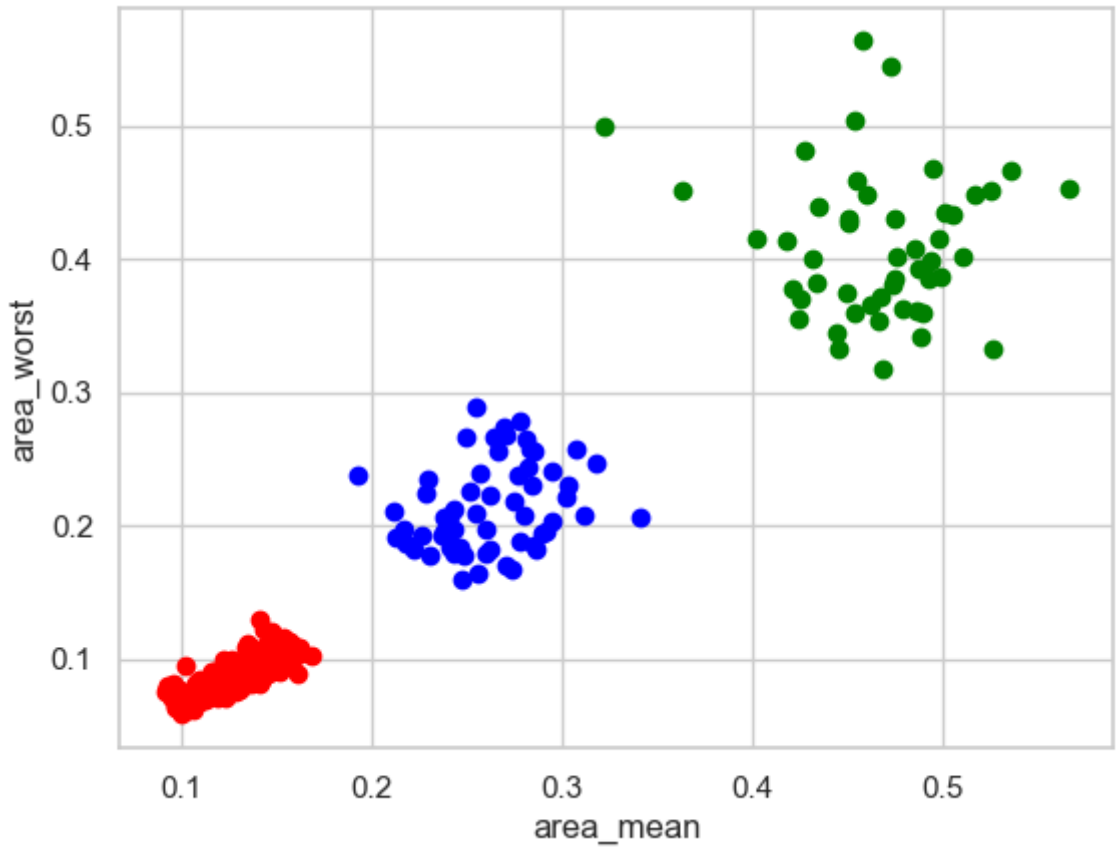
5 rows × 34 columns

In [25]:

```
df1=df[df["New Cluster"]==0]
df2=df[df["New Cluster"]==1]
df3=df[df["New Cluster"]==2]
plt.scatter(df1["area_mean"],df1["area_worst"],color="red")
plt.scatter(df2["area_mean"],df2["area_worst"],color="green")
plt.scatter(df3["area_mean"],df3["area_worst"],color="blue")
plt.xlabel("area_mean")
plt.ylabel("area_worst")
```

Out[25]:

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



In [26]:

```
km.cluster_centers_
```

Out[26]:

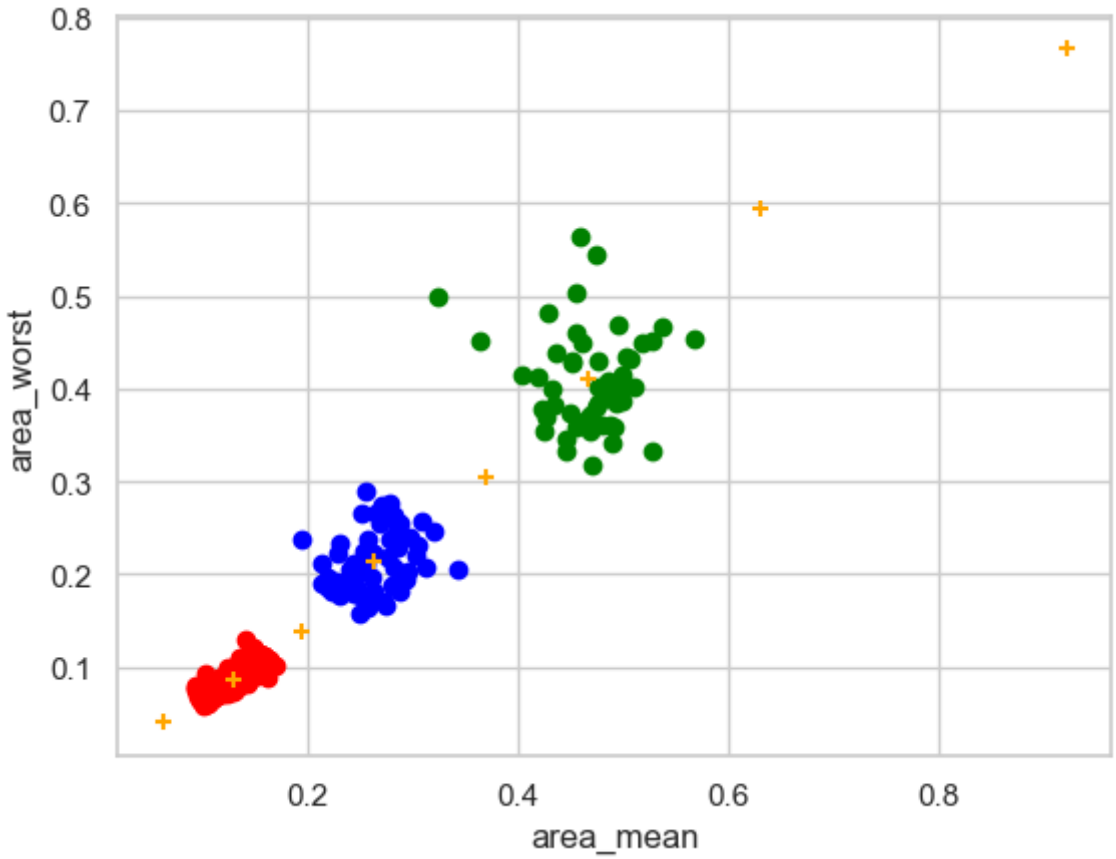
```
array([[0.12812992, 0.08721176],
       [0.46659684, 0.40959878],
       [0.26188133, 0.21348382],
       [0.92110286, 0.76571716],
       [0.36968921, 0.30472318],
       [0.62996516, 0.5945277 ],
       [0.06191901, 0.0402961 ],
       [0.19260919, 0.13948641]])
```

In [27]:

```
df1=df[df["New Cluster"]==0]
df2=df[df["New Cluster"]==1]
df3=df[df["New Cluster"]==2]
plt.scatter(df1["area_mean"],df1["area_worst"],color="red")
plt.scatter(df2["area_mean"],df2["area_worst"],color="green")
plt.scatter(df3["area_mean"],df3["area_worst"],color="blue")
plt.scatter(km.cluster_centers_[:,0],km.cluster_centers_[:,1],color="orange",marker="+")
plt.xlabel("area_mean")
plt.ylabel("area_worst")
```

Out[27]:

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



In [28]:

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

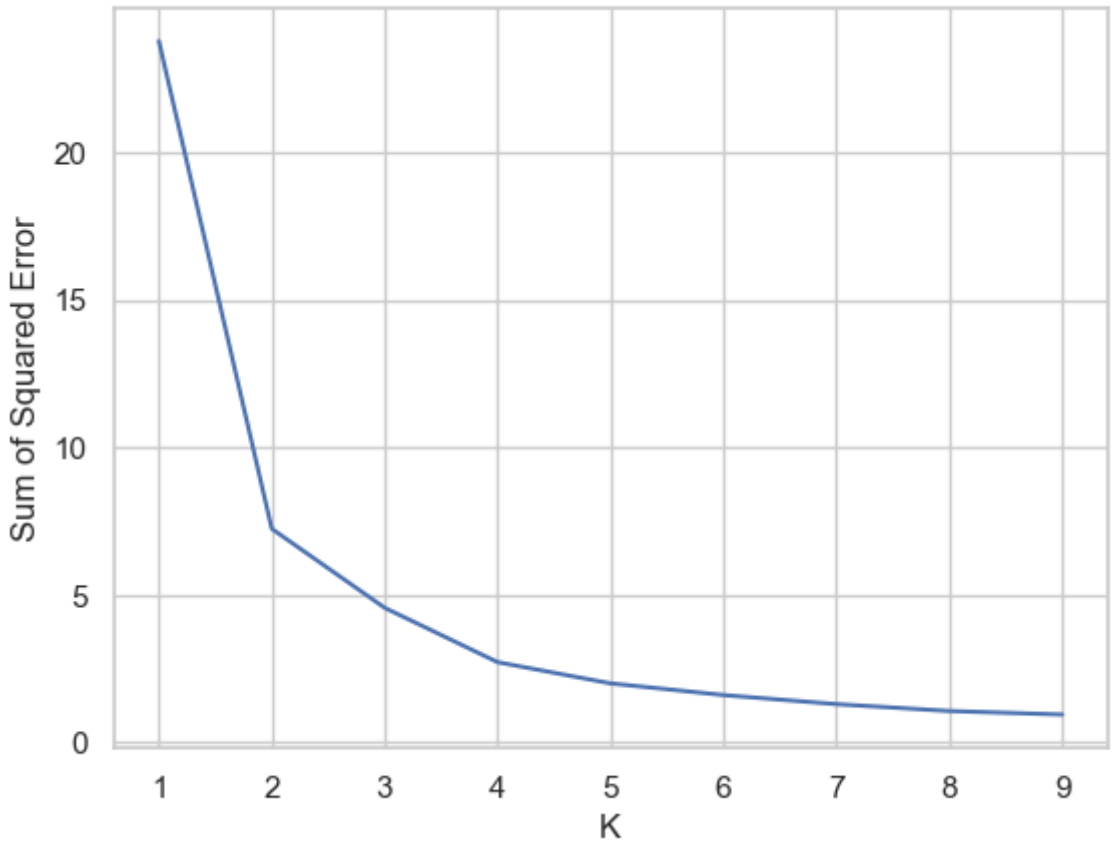
In [29]:

```
for k in k_rng:
    km=KMeans(n_clusters=k)
    km.fit(df[["area_mean","area_worst"]])
    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")
```

```
[23.778690666252167, 7.245561269117197, 4.565244820212003, 2.7231942409326
817, 2.004438108506093, 1.608592731209559, 1.3006491697131504, 1.068487900
5257049, 0.9481340800064465]
```

Out[29]:

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



CONCLUSION : The KMeans model is the best for the given dataset

In [ ]: