Project - 4 (DATASET: Breast Cancer Prediction)

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

In [3]:
```
df=pd.read_csv(r"C:\Users\DELL E5490\Downloads\BreastCancerPrediction.csv")
df
```

Out[3]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	poi
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 [4]:
```
1 df.head()
2
```

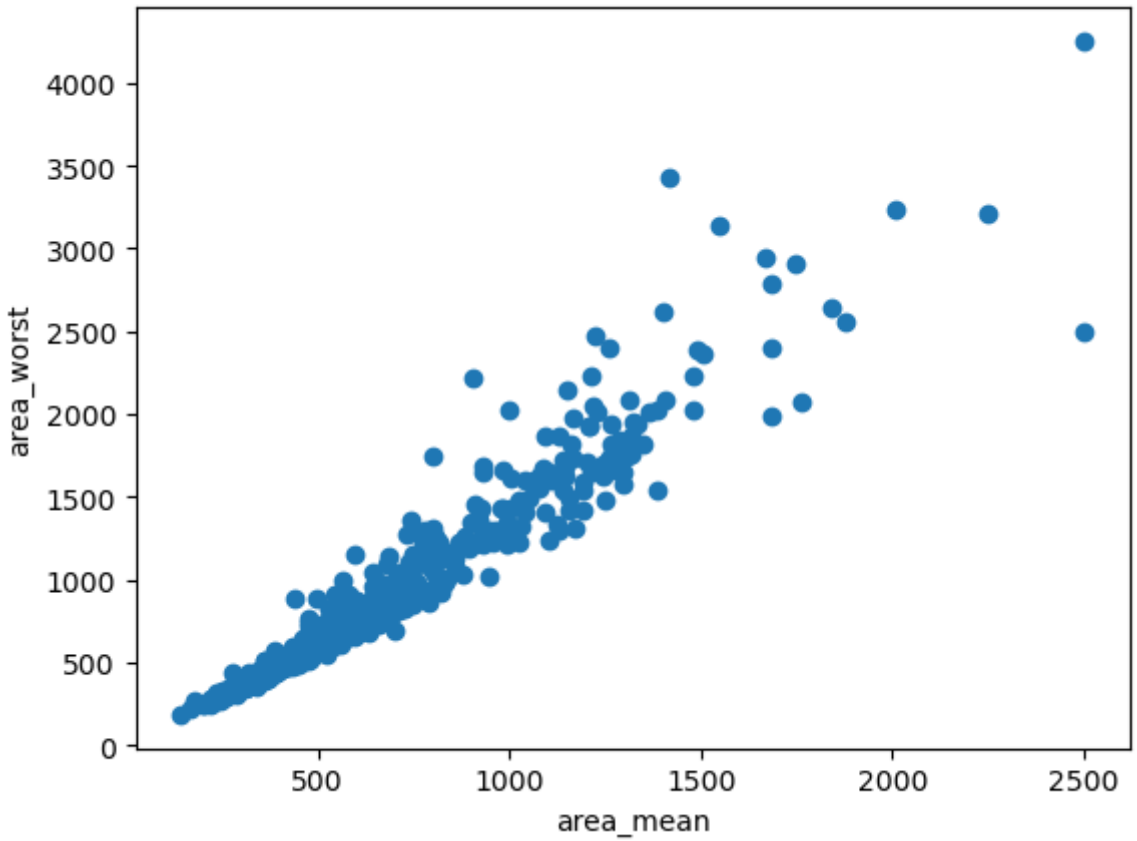
Out[4]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	c 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 [5]:
```
plt.scatter(df["area_mean"],df["area_worst"])
plt.xlabel("area_mean")
plt.ylabel("area_worst")
```

Out[5]: Text(0, 0.5, 'area_worst')



In [6]:
```
from sklearn.cluster import KMeans
km=KMeans()
km
```

Out[6]:
```
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 [8]:
```
y_predicted=km.fit_predict(df[["area_mean","area_worst"]])
y_predicted
```

```
C:\Users\DELL E5490\anaconda3\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(
C:\Users\DELL E5490\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1382: UserWarning: KMeans is known to hav
e a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
```

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

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

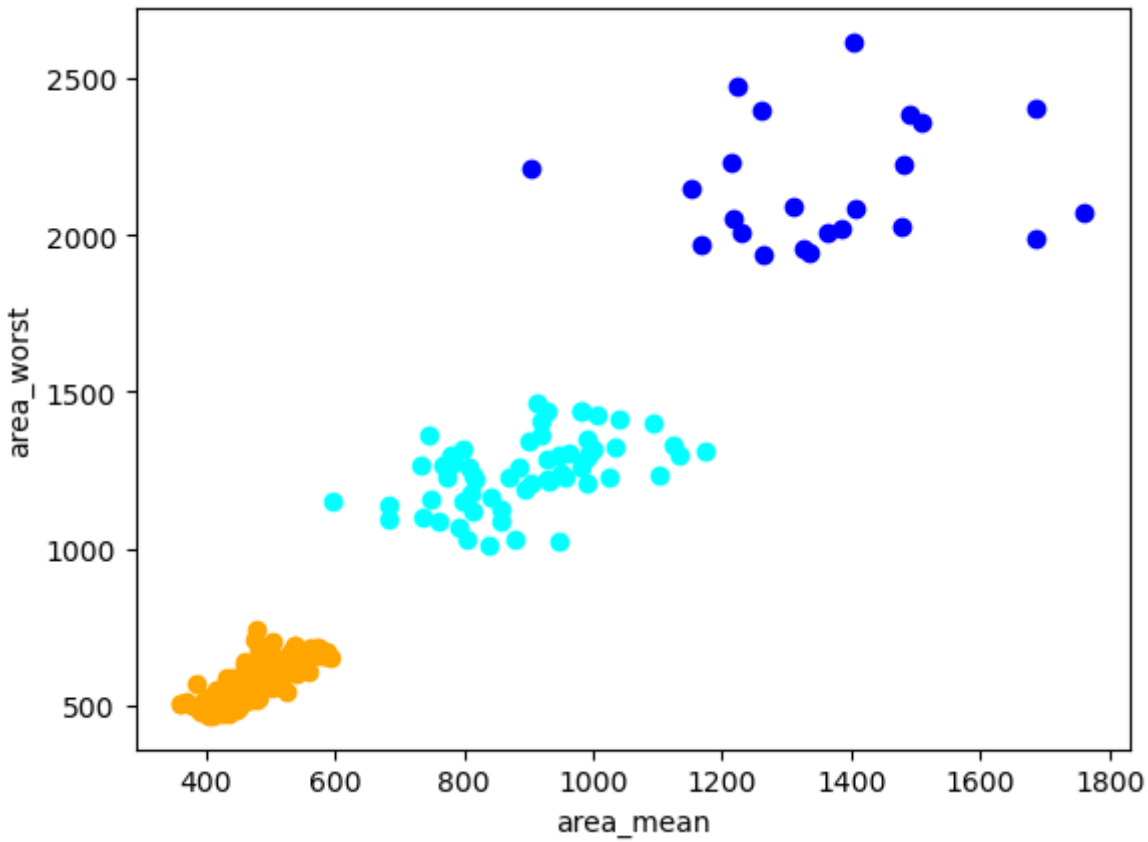
Out[9]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	c 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["area_mean"],df1["area_worst"],color="aqua")
plt.scatter(df2["area_mean"],df2["area_worst"],color="orange")
plt.scatter(df3["area_mean"],df3["area_worst"],color="blue")
plt.xlabel("area_mean")
plt.ylabel("area_worst")
```

Out[10]: Text(0, 0.5, 'area_worst')



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

Out[11]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	c 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 [12]:
```
scaler.fit(df[["area_mean"]])
df["area_mean"]=scaler.transform(df[["area_mean"]])
df.head()
```

Out[12]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	c points
0	842302	M	17.99	10.38	122.80	0.363733	0.11840	0.27760	0.3001	0
1	842517	M	20.57	17.77	132.90	0.501591	0.08474	0.07864	0.0869	0
2	84300903	M	19.69	21.25	130.00	0.449417	0.10960	0.15990	0.1974	0
3	84348301	M	11.42	20.38	77.58	0.102906	0.14250	0.28390	0.2414	0
4	84358402	M	20.29	14.34	135.10	0.489290	0.10030	0.13280	0.1980	0

5 rows × 34 columns

In [13]:
```
km=KMeans()
```

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

```
C:\Users\DELL E5490\anaconda3\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(
C:\Users\DELL E5490\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1382: UserWarning: KMeans is known to hav
e a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
```

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

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

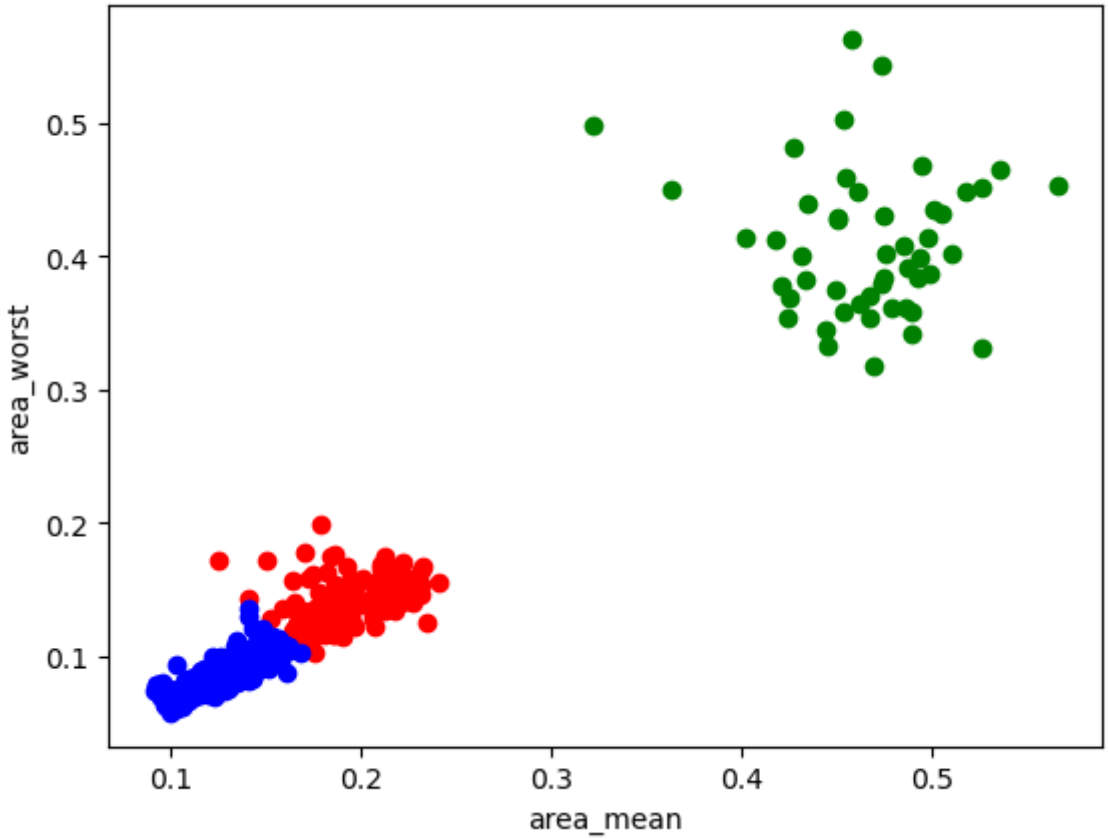
Out[16]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	c points
0	842302	M	17.99	10.38	122.80	0.363733	0.11840	0.27760	0.3001	0
1	842517	M	20.57	17.77	132.90	0.501591	0.08474	0.07864	0.0869	0
2	84300903	M	19.69	21.25	130.00	0.449417	0.10960	0.15990	0.1974	0
3	84348301	M	11.42	20.38	77.58	0.102906	0.14250	0.28390	0.2414	0
4	84358402	M	20.29	14.34	135.10	0.489290	0.10030	0.13280	0.1980	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["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[17]: Text(0, 0.5, 'area_worst')

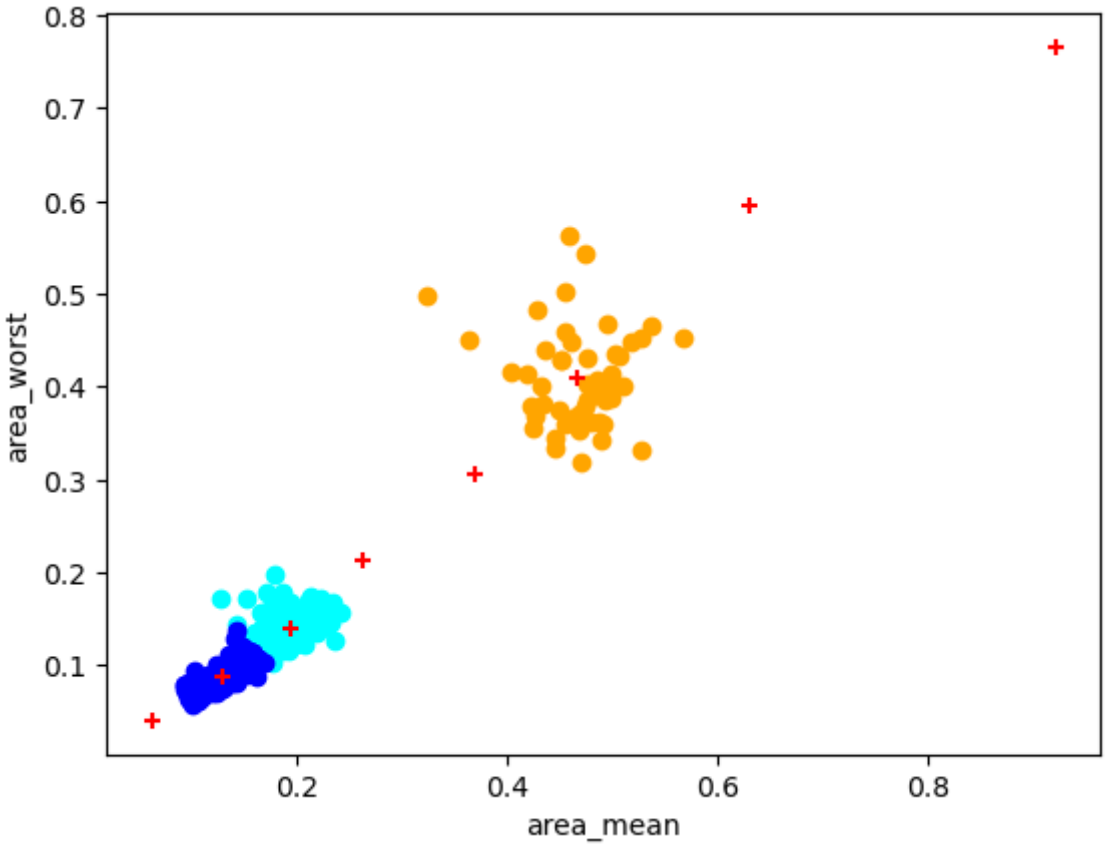


In [18]:
```
1 km.cluster_centers_
```

Out[18]:
```
array([[0.19279325, 0.13977361],
       [0.46659684, 0.40959878],
       [0.12835749, 0.08729682],
       [0.92110286, 0.76571716],
       [0.62996516, 0.5945277 ],
       [0.36968921, 0.30472318],
       [0.06191901, 0.0402961 ],
       [0.26188133, 0.21348382]])
```

In [19]:
```
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="aqua")
plt.scatter(df2["area_mean"],df2["area_worst"],color="orange")
plt.scatter(df3["area_mean"],df3["area_worst"],color="blue")
plt.scatter(km.cluster_centers_[:,0],km.cluster_centers_[:,1],color="red",marker="+")
plt.xlabel("area_mean")
plt.ylabel("area_worst")
```

Out[19]: Text(0, 0.5, 'area_worst')



In [20]:

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

In [21]:

```
for k in k_rng:
    km=KMeans(n_clusters=k)
    km.fit(df[["area_mean","area_worst"]])
    sse.append(km.inertia_)
print(sse)
plt.plot(k_rng,sse)
plt.xlabel("K")
plt.ylabel("Sum of Squared Error")

```

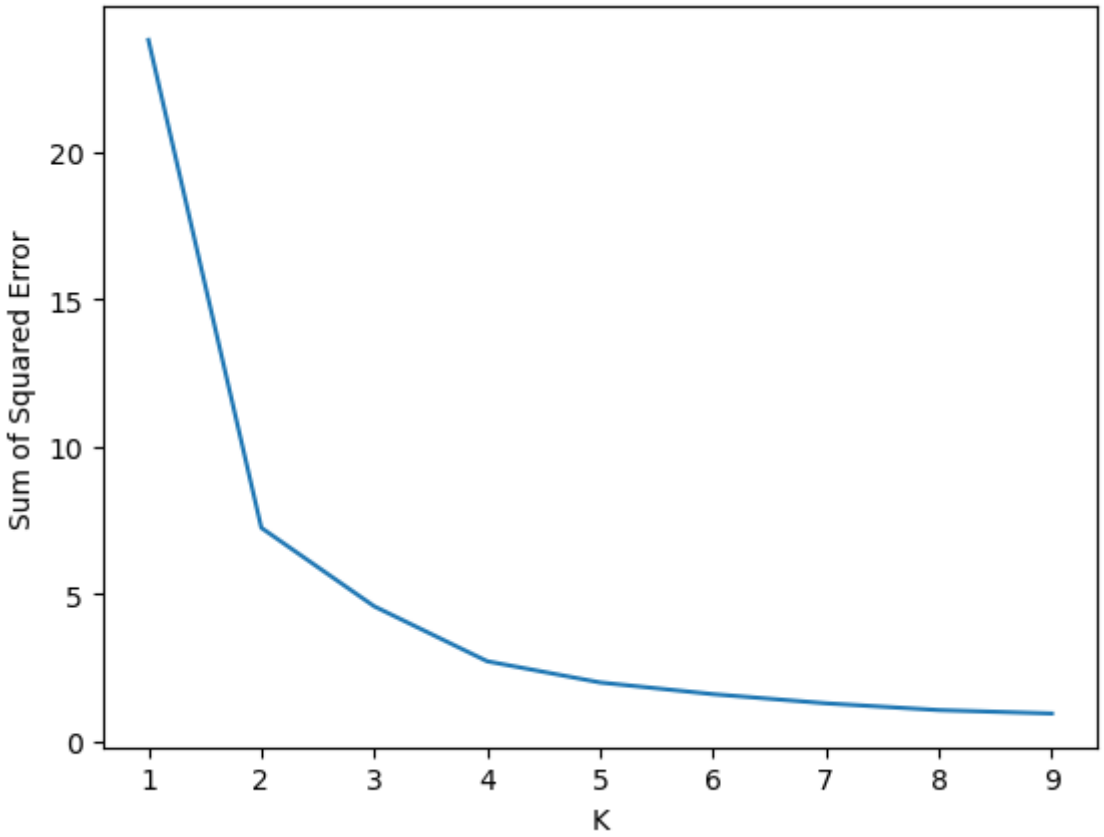
```
C:\Users\DELL E5490\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
C:\Users\DELL E5490\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1382: UserWarning: KMeans is known to hav
e a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
C:\Users\DELL E5490\anaconda3\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(
C:\Users\DELL E5490\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1382: UserWarning: KMeans is known to hav
e a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
C:\Users\DELL E5490\anaconda3\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(
C:\Users\DELL E5490\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1382: UserWarning: KMeans is known to hav
e a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
C:\Users\DELL E5490\anaconda3\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(
C:\Users\DELL E5490\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1382: UserWarning: KMeans is known to hav
e a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
C:\Users\DELL E5490\anaconda3\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(
C:\Users\DELL E5490\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1382: UserWarning: KMeans is known to hav
e a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
C:\Users\DELL E5490\anaconda3\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(
C:\Users\DELL E5490\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1382: UserWarning: KMeans is known to hav
e a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
```

```
  warnings.warn(
C:\Users\DELL E5490\anaconda3\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(
C:\Users\DELL E5490\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1382: UserWarning: KMeans is known to hav
e a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
C:\Users\DELL E5490\anaconda3\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(
C:\Users\DELL E5490\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1382: UserWarning: KMeans is known to hav
e a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(
C:\Users\DELL E5490\anaconda3\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(
C:\Users\DELL E5490\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1382: UserWarning: KMeans is known to hav
e a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting
the environment variable OMP_NUM_THREADS=3.
  warnings.warn(

[23.778690666252167, 7.245561269117197, 4.585106496063475, 2.7231942409326817, 2.004438108506093, 1.608390929665570
7, 1.2985554076486587, 1.0684640084665036, 0.9530965081394718]
```

Out[21]:

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



CONCLUSION

for the given dataset we can use multiple models,for that models we get different types of accuracies but that accuracies is not good so,that's why we will take it as a clustering and done with K-Means Clustering

In [ ]:
```
1
```