Choosing Number of Cluster in GaussianMixture and BayesianGaussianMixture

You could use the inertia or the silhouette score to select the appropriate number of clusters, but with Gaussian mixtures, it is not possible to use these metrics because they are not reliable when the clusters are not spherical or have different sizes. Instead, you can try to find the model that minimizes a theoretical information criterion such as the Bayesian information criterion (BIC) or the Akaike information criterion (AIC).

Implementation:

Import dataset(mall customer -available on Kaggle) and mixture library as follows,

```
import pandas as pd
from sklearn.mixture import GaussianMixture
df=pd.read_csv("Mall_Customers.csv")# available on kaggle
x=df.iloc[:,3:]
df.head()
```

Output:

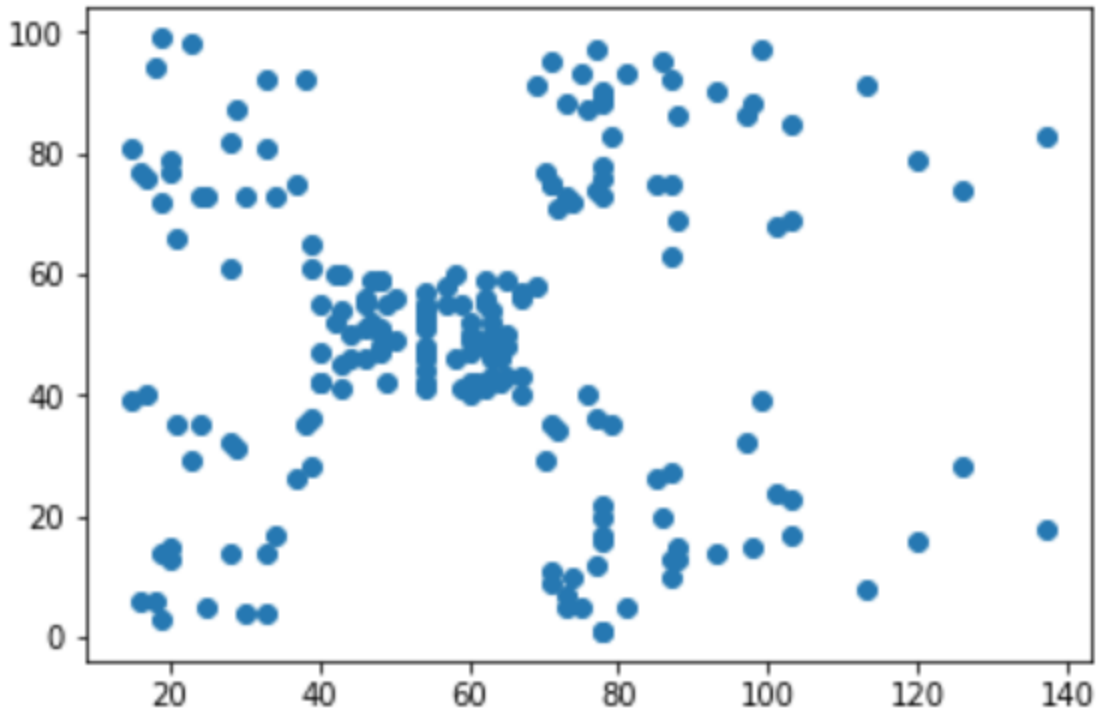
	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

Plot the dataset :

Using scatter plot we plot annual income and spending score of customer as follows,

```
import matplotlib.pyplot as plt
plt.scatter(x['Annual Income (k$)'],x['Spending Score (1-100)'])
```

Output:



Calculate AIC and BIC:

In this section we calculate two theoretical criterion Akaike Information Criteria (AIC) and Bayesian Information Criteria (BIC) for our model for different cluster values,

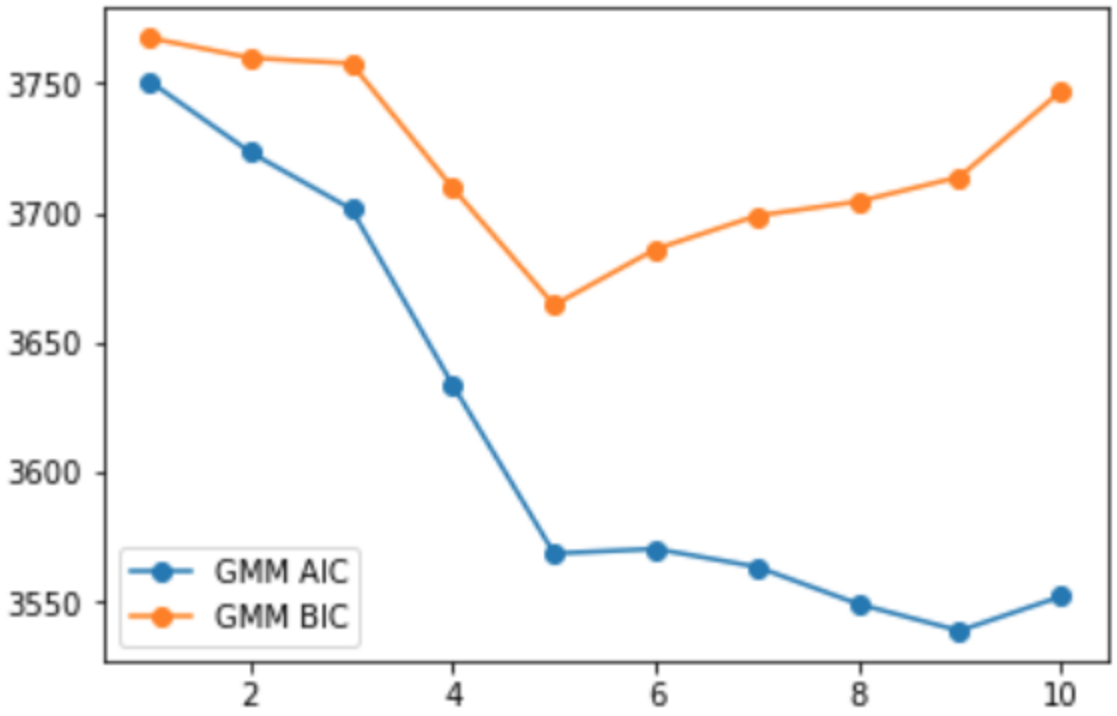
```
gm_aic=[]
gm_bic=[]
for i in range(1,11):
    gmm1=GaussianMixture(n_components=i)
    gmm1.fit(x)
    gm_aic.append(gmm1.aic(x))
    gm_bic.append(gmm1.bic(x))
```

Plot AIC and BIC:

In this section we plot the AIC and BIC vs number of clusters and observe the graph,

```
plt.plot(n_clusters,gm_aic,marker='o',label="GMM AIC")
plt.plot(n_clusters,gm_bic,marker='o',label="GMM BIC")
plt.legend()
```

Output:



From the above graph it is confirmed that at cluster number value =5 both curves get deflected hence the optimal number of clusters should be 5.

Rather than manually searching for the optimal number of clusters, it is possible to use instead the BayesianGaussianMixture class which is capable of giving weights equal (or close) to zero to unnecessary clusters.

```
from sklearn.mixture import BayesianGaussianMixture

bgm_weights=[]
for i in range(1,11):
    bgm=BayesianGaussianMixture(n_components=i)
    bgm.fit(x)
    bgm_weights=bgm.weights_
import numpy as np
np.round(bgm.weights_, 2)
```

Output:

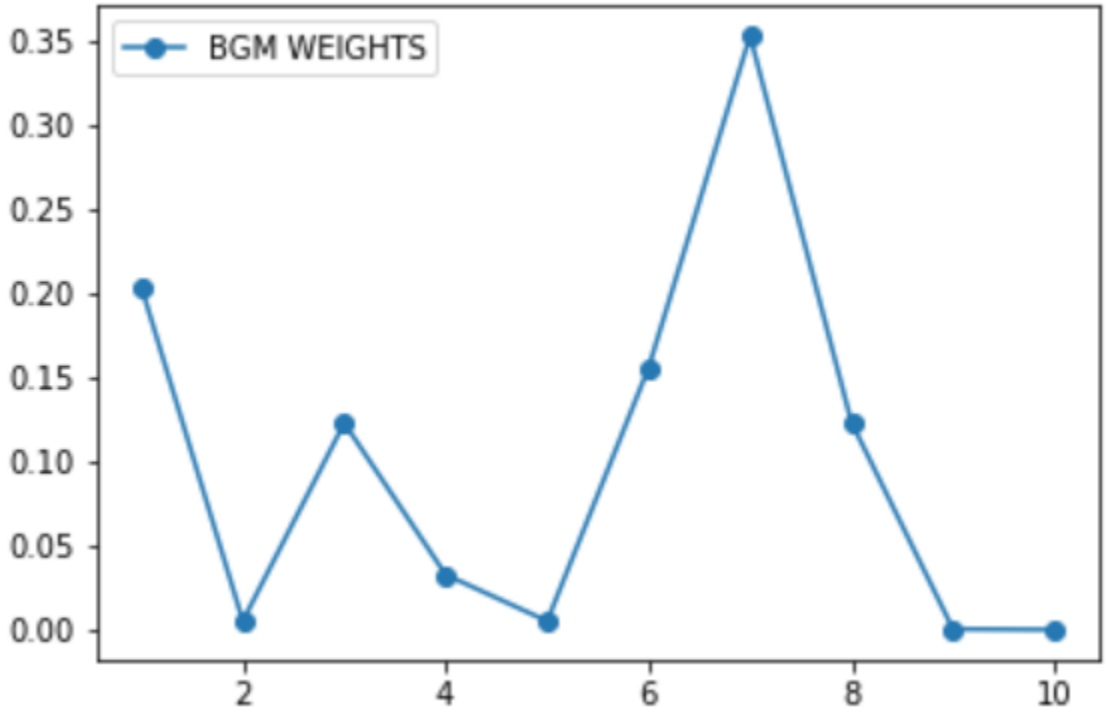
```
array([0.2 , 0.01, 0.12, 0.03, 0.01, 0.16, 0.35, 0.12, 0.  , 0.  ])
```

Perfect. The algorithm automatically detected that only 5 clusters are needed, and the resulting clusters are almost identical to the ones in aic and bic plot for GaussianMixture.

Plot weights_:

```
plt.plot(n_clusters,bgm_weights,marker='o',label="BGM WEIGHTS")
plt.legend()
```

Output:



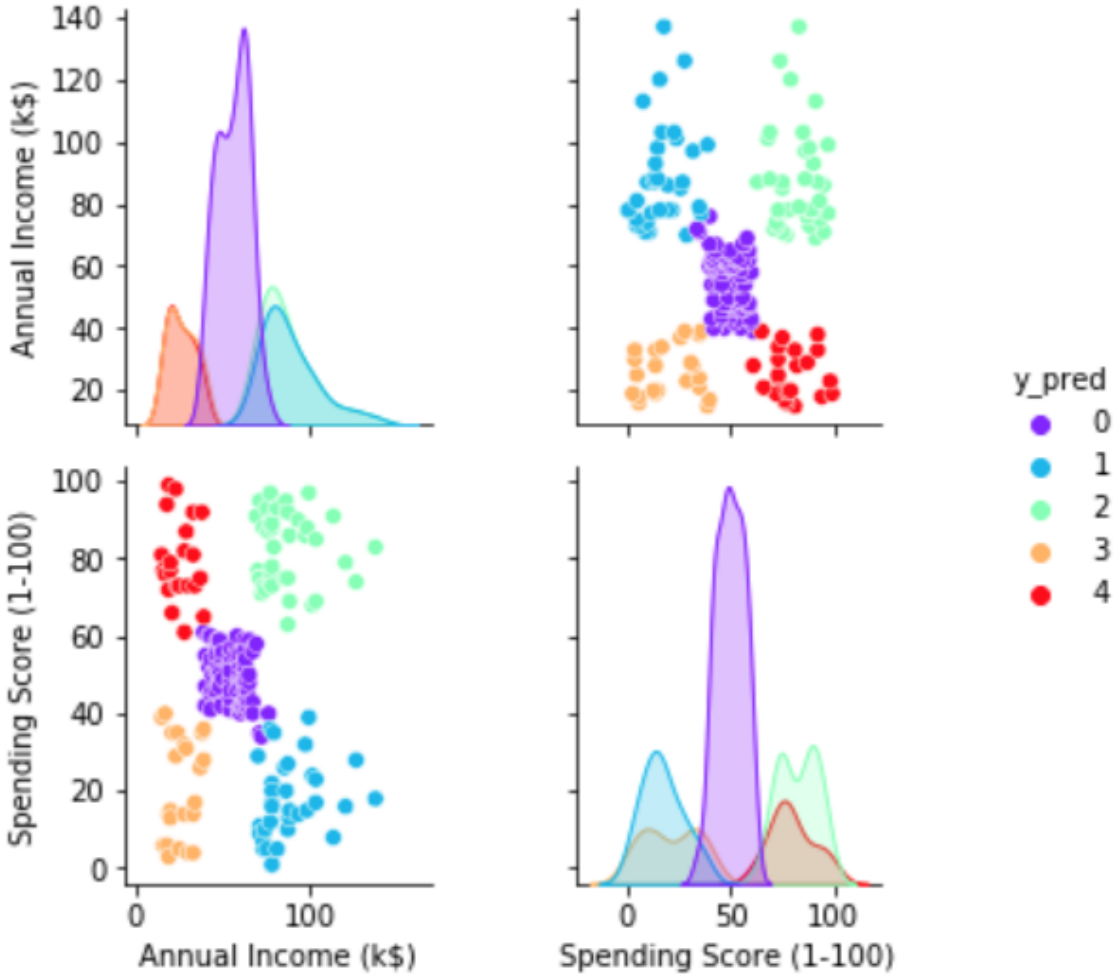
Apply BGM on dataset :
In this section we directly apply BGM instead of GMM as it is more promising in clusters prediction,

```
#now one of method to predict
bgm1=BayesianGaussianMixture(n_components=5)
bgm1.fit(x)
y_pred=bgm1.predict(x)
x['y_pred']=pd.DataFrame(y_pred)
```

Plot Result:
In this section we use pairplot method of seaborn package for better visualization.

```
import seaborn as sns
sns.pairplot(x,hue='y_pred',palette="rainbow")
```

Output:



Hence we found all desired numbers of clusters using the BGM method.