

IE406 Machine Learning

Lab Assignment - 9

Group 28

201801015: Shantanu Tyagi
201801076: Shivani Nandani
201801407: Pratvi Shah
201801408: Arkaprabha Baerjee

Question 1

Implement GMM algorithm. Use the two dataset files for the following:

1. Visualize the datasets.
2. Use random initial cluster centers and try the algorithm for different values for K (i.e. k=1,2,3...)
3. Visualize the cluster formation for each value of K for both the datasets.
4. Utilize the Elbow method to find out the optimal number of Clusters (i.e. K)

Compare with Q2 of lab8.

Answer

We have 2 datasets for this question. Dataset in q1a is 2D and is not labelled while q1b dataset is 2D but has labels.

We visualized the data and applied k-means clustering algorithm from sklearn library. We calculate the AIC and BIC values for finding the optimal clustering model and tried to implement SSE based curve for comparison between k-means and GMM.

Code

```
1 # Q1
2
3 import numpy as np
4 import matplotlib.pyplot as plt
5 import pandas as pd
6 import random
7 %matplotlib inline
8 from sklearn.preprocessing import StandardScaler
9 import seaborn as sns
10 from sklearn.metrics import mean_squared_error as mse
11 from sklearn.metrics import accuracy_score
12 from scipy.stats import mode
13 from sklearn.cluster import KMeans
14 from sklearn.mixture import GaussianMixture
15 import warnings
16 warnings.filterwarnings('ignore')
17 from scipy.stats import multivariate_normal as mvn
18
19
20 #Ref: https://scikit-learn.org/stable/auto_examples/mixture/plot_gmm_covariances.html#sphx-glr
    -auto-examples-mixture-plot-gmm-covariances-py
21 from matplotlib.patches import Ellipse
22 def plot_2D(data, centers, y_pred, gmm, axes_label):
23     plt.figure(figsize=[10,6])
24     ax = plt.gca()
25
```

```

26 # Plot points according to the value of their max probability
27 size = 100 * gmm.predict_proba(data).max(1) ** 2 # square emphasizes differences
28 ax.scatter(data[:, 0], data[:, 1], c=y_pred, cmap='viridis', s=size)
29
30 # Plot the cluster area depending on the mean and covariance
31 for pos, covar, w in zip(gmm.means_, gmm.covariances_, gmm.weights_):
32     # Convert covariance to principal axes
33     U, s, Vt = np.linalg.svd(covar)
34     angle = 180*(np.arctan2(U[1, 0], U[0, 0]))/np.pi
35     width, height = 2 * np.sqrt(s)
36
37     # Draw the Ellipse for n*sigma
38     for n in range(1, 4):
39         ax.add_patch(Ellipse(pos, n*width, n*height, angle, alpha=0.1*(w/gmm.weights_.max
40         ())))
41
42     ax.axis('equal')
43     plt.grid()
44     plt.title('K='+str(i), fontsize=15)
45     plt.xlabel(axes_label[0], fontsize=15)
46     plt.ylabel(axes_label[1], fontsize=15)
47     plt.show()
48
49 def find_inertia(k, centroids, data):
50     sse_error=0
51     for i in range(len(data)):
52         min_val= np.inf
53         for j in range(k):
54             temp = (data[i][0]-centroids[j][0])**2 + (data[i][1]-centroids[j][1])**2
55             min_val = min(min_val, temp)
56         sse_error = sse_error + min_val
57     return sse_error
58
59 # Q1 a)
60 data = pd.read_excel('Question2a.xlsx')
61 data.head()
62
63 plt.figure(figsize=(10,6))
64 sns.scatterplot(data['x'], data['y'], s=200)
65 plt.xlabel('X', fontsize=15)
66 plt.ylabel('Y', fontsize=15)
67 plt.grid()
68
69 # Standardize data
70 df=[]
71 df.append(data['x'])
72 df.append(data['y'])
73 df=np.array(df).transpose()
74 scaler = StandardScaler()
75 df_scaled = scaler.fit_transform(df)
76
77 #Q1a 3) Visualize the cluster formation for each k
78
79 SSE_q1a = []
80 SSE_q1a_kmeans=[]
81 q1a_aic = []
82 q1a_bic = []
83 axes_label=['X', 'Y']
84 for i in range(1,11):
85     # GMM
86     gmm = GaussianMixture(n_components=i, covariance_type='full', random_state=0).fit(df_scaled)
87     y_pred = gmm.predict(df_scaled)
88     probs = gmm.predict_proba(df_scaled)
89
90     # K-Means
91     kmeans = KMeans(n_clusters = i, init='k-means++')
92     kmeans.fit(df_scaled)
93
94     # Creating plot
95     centers = np.zeros((i,2))
96     for j in range(i):
97         density = mvn(cov=gmm.covariances_[j], mean=gmm.means_[j]).logpdf(df_scaled)
98         centers[j, :] = df_scaled[np.argmax(density)]
99     plot_2D(df_scaled, centers, y_pred, gmm, axes_label)

```

```

100
101 #Calculate SSE
102 SSE_q1a.append(find_inertia(i,centers,df_scaled))
103 SSE_q1a_kmeans.append(kmeans.inertia_)
104 q1a_aic.append(gmm.aic(df_scaled))
105 q1a_bic.append(gmm.bic(df_scaled))
106
107 #Q1a 4) Plot elbow curve
108
109 x_axis=np.arange(1,11)
110 plt.figure(figsize=[10,6])
111 plt.plot(x_axis,q1a_aic,'b-*')
112 plt.plot(x_axis,q1a_bic,'k-o')
113 plt.grid()
114 plt.legend(['AIC','BIC'],fontsize=15)
115 plt.xlabel('K(number of clusters)',fontsize=15)
116
117 x_axis=np.arange(1,11)
118 plt.figure(figsize=[10,6])
119 plt.plot(x_axis,SSE_q1a,'b-*')
120 plt.plot(x_axis,SSE_q1a_kmeans,'k-o')
121 plt.grid()
122 plt.ylabel('SSE',fontsize=15)
123 plt.xlabel('K(number of clusters)',fontsize=15)
124 plt.title('Elbow Curve',fontsize=15)
125 plt.legend(['GMM','K-Means'],fontsize=15)
126
127 # Q1b
128
129 data2 = pd.read_excel('Question2b.xls')
130 data2.head()
131
132 plt.figure(figsize=(10,6))
133 sns.scatterplot(data2['x1'],data2['x2'],s=200,c=data2['y'])
134 plt.xlabel('X1',fontsize=15)
135 plt.ylabel('X2',fontsize=15)
136 plt.grid()
137
138 # Standardize data
139 df2=[]
140 df2.append(data2['x1'])
141 df2.append(data2['x2'])
142 df2=np.array(df2).transpose()
143 scaler = StandardScaler()
144 df_scaled2 = scaler.fit_transform(df2)
145
146 #Q1b 3) Visualize the cluster formation for each k
147
148 SSE_q1b = []
149 SSE_q1b_kmeans=[]
150 q1b_aic=[]
151 q1b_bic=[]
152 n_iter = []
153 axes_label=['X1','X2']
154 for i in range(1,11):
155     # GMM
156     gmm = GaussianMixture(n_components=i, covariance_type='full', random_state=0).fit(df_scaled2)
157     y_pred = gmm.predict(df_scaled2)
158     probs = gmm.predict_proba(df_scaled2)
159     n_iter.append(gmm.n_iter_)
160
161     # K-Means
162     kmeans = KMeans(n_clusters = i, init='k-means++')
163     kmeans.fit(df_scaled2)
164
165     # Creating plot
166     centers = np.zeros((i,2))
167     for j in range(i):
168         density = mvn(cov=gmm.covariances_[j], mean=gmm.means_[j]).logpdf(df_scaled2)
169         centers[j, :] = df_scaled2[np.argmax(density)]
170     plot_2D(df_scaled2, centers, y_pred, gmm, axes_label)
171
172     if i==2:
173         print('Accuracy for k=2: ',accuracy_score(data2['y'],y_pred))

```

```

174
175 #Calculate SSE
176 SSE_q1b.append(find_inertia(i,centers,df_scaled2))
177 SSE_q1b_kmeans.append(kmeans.inertia_)
178 q1b_aic.append(gmm.aic(df_scaled2))
179 q1b_bic.append(gmm.bic(df_scaled2))
180
181 #Q1b 4) Plot elbow curve
182
183 x_axis=np.arange(1,11)
184 plt.figure(figsize=[10,6])
185 plt.plot(x_axis,q1b_aic,'b-*')
186 plt.plot(x_axis,q1b_bic,'k-o')
187 plt.grid()
188 plt.legend(['AIC','BIC'],fontsize=15)
189 plt.xlabel('K(number of clusters)',fontsize=15)
190
191 x_axis=np.arange(1,11)
192 plt.figure(figsize=[10,6])
193 plt.plot(x_axis,SSE_q1b,'b-*')
194 plt.plot(x_axis,SSE_q1b_kmeans,'k-o')
195 plt.grid()
196 plt.ylabel('SSE',fontsize=15)
197 plt.xlabel('K(number of clusters)',fontsize=15)
198 plt.title('Elbow Curve',fontsize=15)
199 plt.legend(['GMM','K-Means'],fontsize=15)

```

Listing 1: Question 1

Result

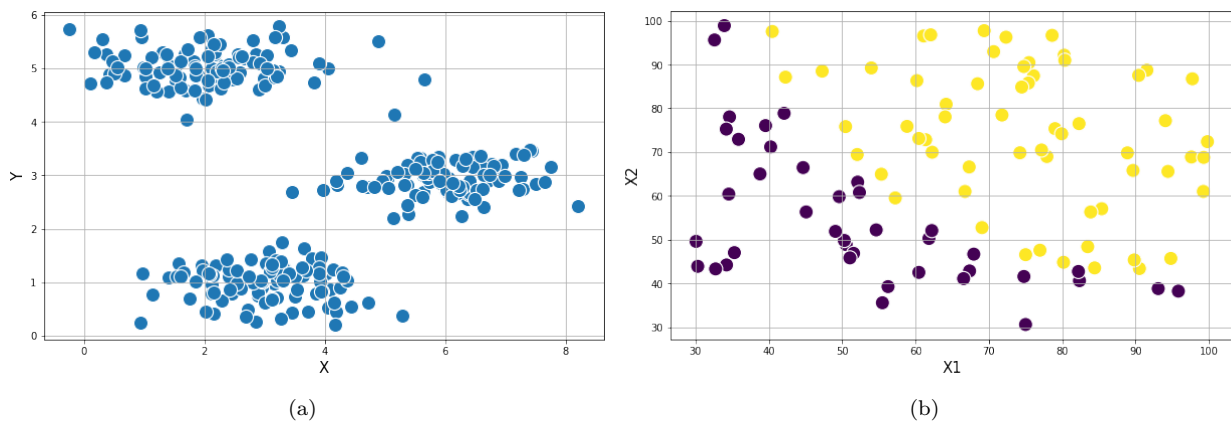


Figure 1: Initial data

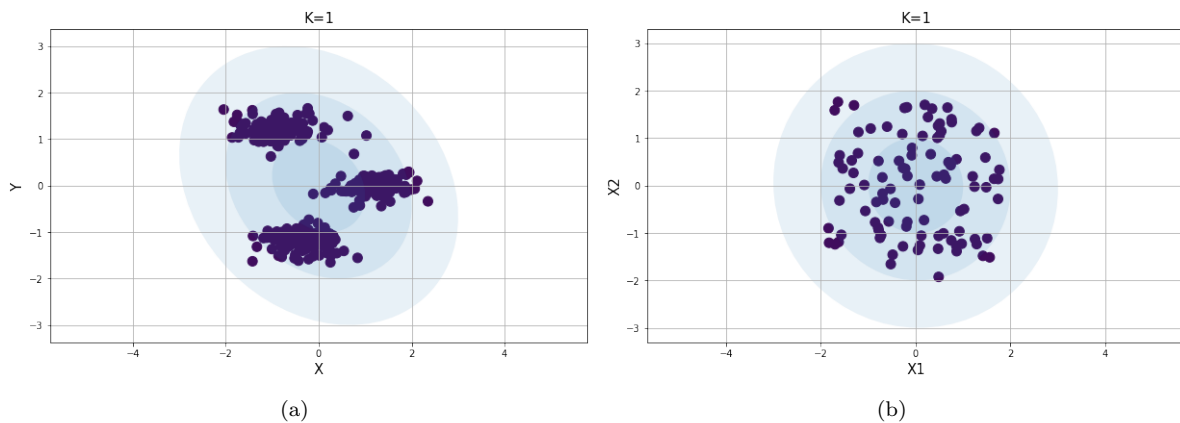
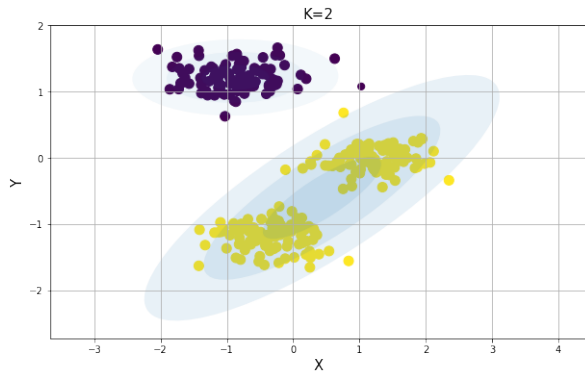
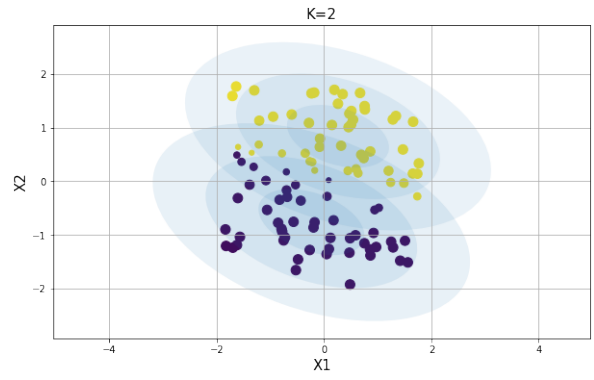


Figure 2: $K = 1$

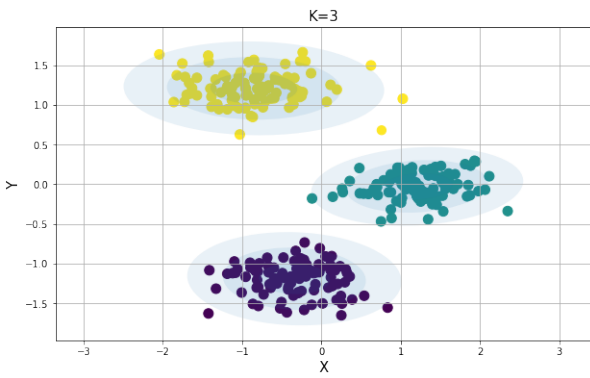


(a)

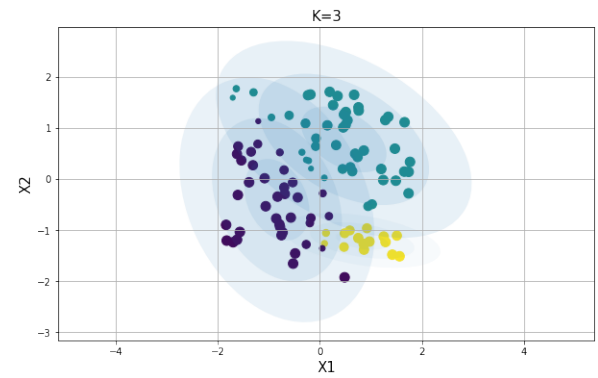


(b)

Figure 3: $K = 2$

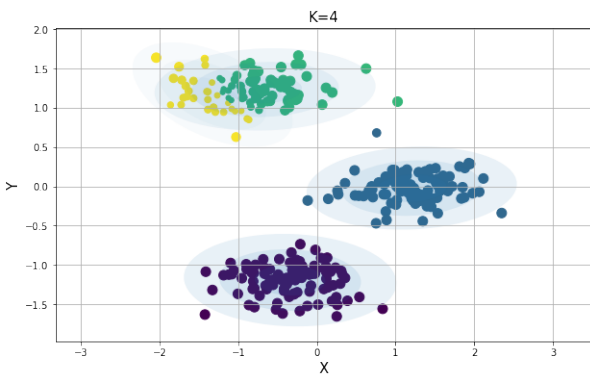


(a)

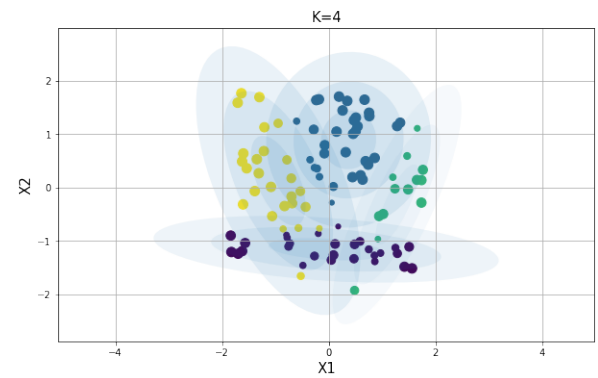


(b)

Figure 4: $K = 3$

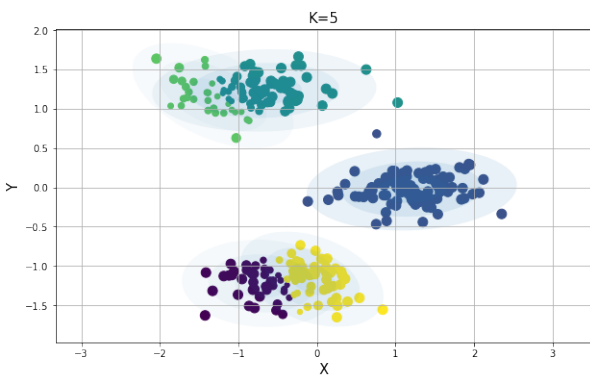


(a)

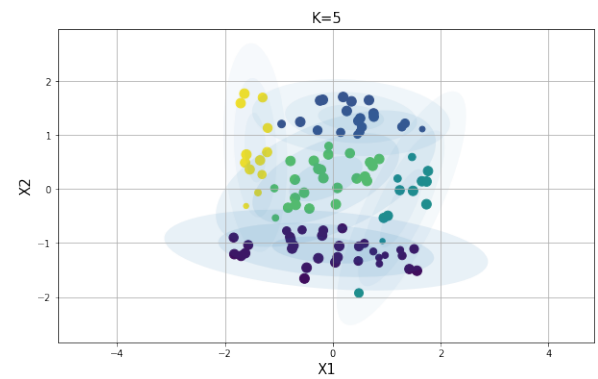


(b)

Figure 5: $K = 4$

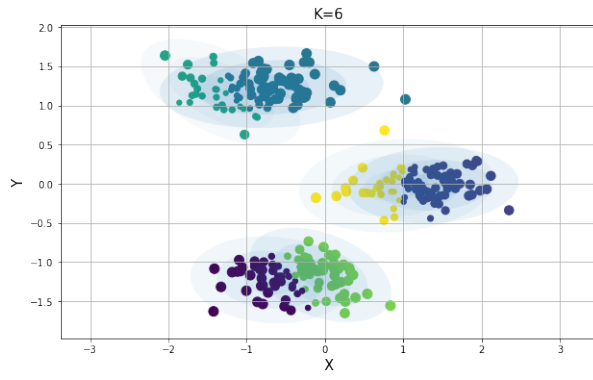


(a)

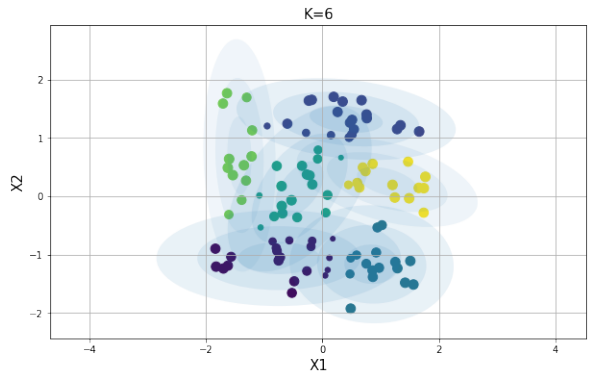


(b)

Figure 6: $K = 5$

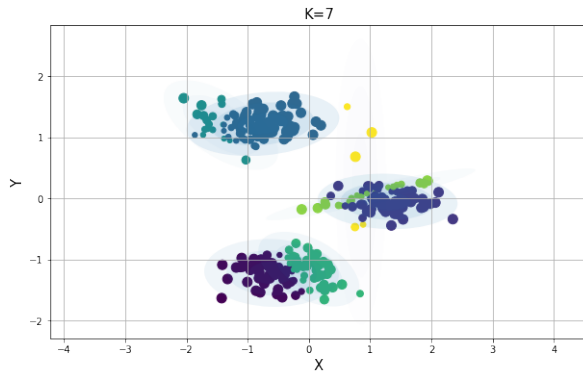


(a)

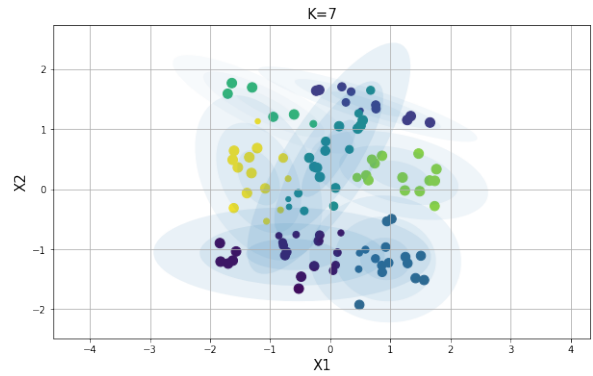


(b)

Figure 7: $K = 6$

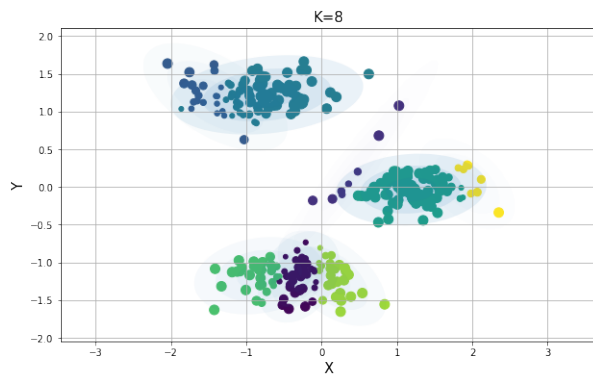


(a)

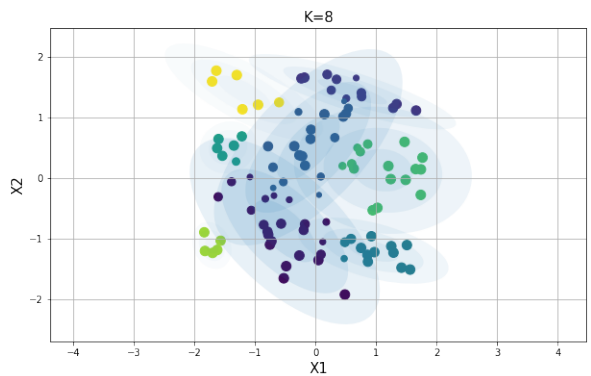


(b)

Figure 8: $K = 7$

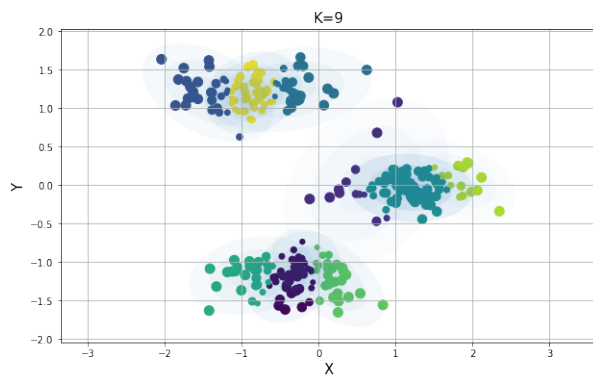


(a)

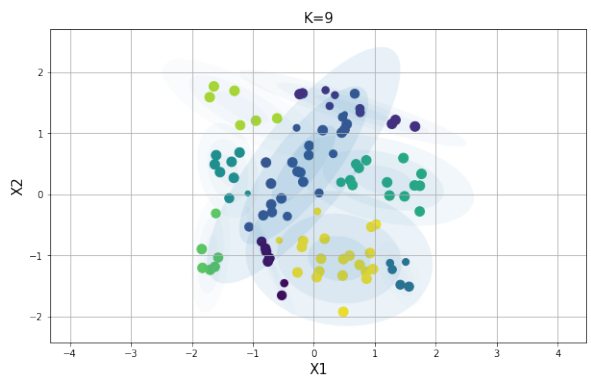


(b)

Figure 9: $K = 8$



(a)



(b)

Figure 10: $K = 9$

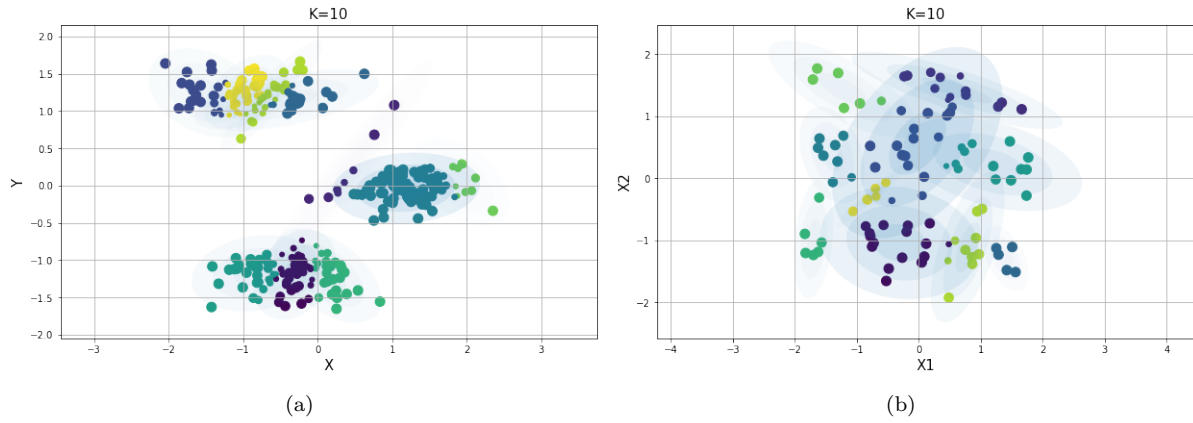


Figure 11: $K = 10$

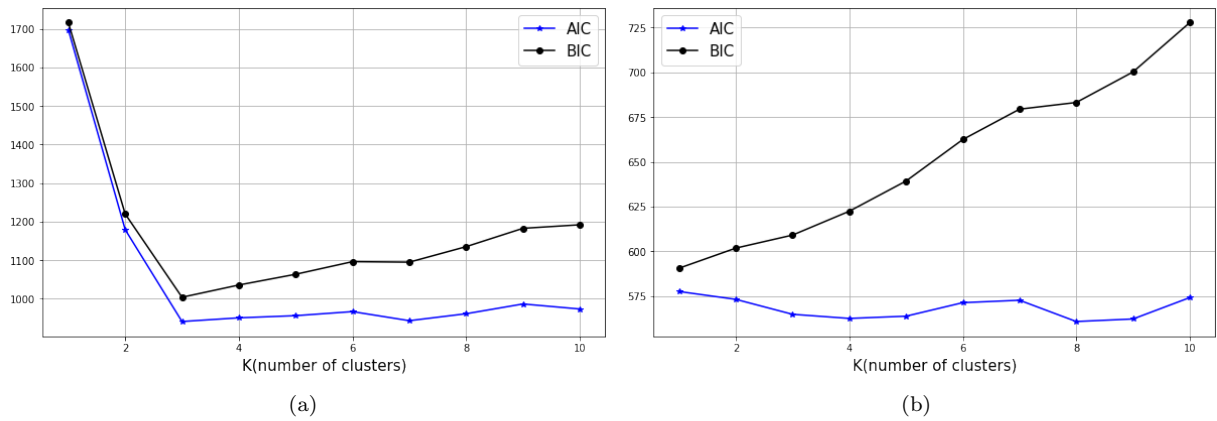


Figure 12: Elbow curve(AIC/BIC)

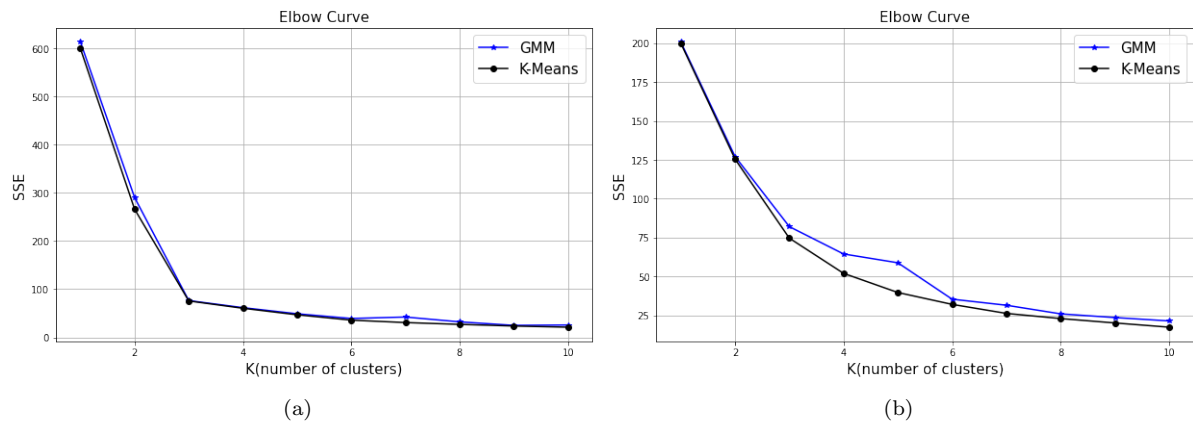


Figure 13: Comparison based on SSE(Only for exploration)

Observation/ Justification

We could draw the following observations:

- We can judge from Fig. 1 (a) that it has 3 clusters while 1b has scattered data and hence, we cannot directly comment about the number of clusters.
- From previous lab, kmeans clustering predicted 3 clusters for part (a) and for part (b), 5 was taken as a safe approximation for the number of clusters.
- From Fig. 12 a), we can verify that the initial assumption of 3 clusters and the result obtained by both k-means clustering and GMM are consistent.
- From Fig. 12 b), we can observe that AIC decreases initially till $k=3$ remains constant thereafter till $k=5$ and then increases. This does not show very promising result about the optimal number of clusters but if we only consider the trend from AIC then we can say that optimal number of clusters can be anything

between 3-5 since the curve shows that the clustering has maximum likelihood and only considerable penalty in terms of the number of cluster for k in range 3 to 5. But BIC values did not provide any significant observation. This result is similar to that obtained from k-means clustering in previous lab.

- We expected GMM to perform better than k-means clustering in part b as it is one of the limitations of k-means clustering that it cannot work well with data having complex distribution. We tried to implement this algorithm for multiple random initialization but could not get any better result.
- The accuracy in part (b), (third column in dataset was assumed to be the true labels) was found to be around 0.8.
- For this lab we have implemented SSE by approximating the centers of all the gaussian distributions for a given k . We implemented this by generating the log of the probability density function of multivariate normal distribution with mean and covariance as the cluster mean and cluster covariance. For each of the distribution we then choose the entry with maximum value of log probability as the center for that cluster. This was only done to provided a consistent measure of error with k-means clustering. It is an approximate method(not sure about the correctness) but provides consistent results with that of k-means clustering implemented in previous lab as shown in Fig 13. This plot also shows that $k=3$ is optimal for Q1(a) and $k=5$ is the point after which the trajectory changes in Q1b.

Question 2

In the given dataset (dataset3.csv from lab:8), you have Customer.Id, Gender, Age, Annual Income (\$), and Spending Score (which is the calculated value of how much a customer has spent in the mall, the more the value, the more he has spent). From this dataset, you need to calculate some patterns. Compare your result with k-means results which you performed in Lab-8.

Answer

We will consider pairs of features and perform k means clustering of each of these pairs of features and using the elbow curve, find out the ideal value for k. We will also look at patterns in the dataset.

Code

```
1 # import libraries
2 import numpy as np
3 import pandas as pd
4 import matplotlib.pyplot as plt
5 import seaborn as sns
6 from sklearn.mixture import GaussianMixture
7 from sklearn.preprocessing import StandardScaler
8
9
10 # function to find inertia for GMM
11 def find_inertia(k, centroids, data):
12     sse_error = 0
13     for i in range(len(data)):
14         min_val = np.inf
15         for j in range(k):
16             temp = (data[i][0]-centroids[j][0])**2 + \
17                 (data[i][1]-centroids[j][1])**2
18             min_val = min(min_val, temp)
19         sse_error = sse_error + min_val
20     return sse_error
21
22
23 # import datafile
24 data = pd.read_csv('dataset3.csv')
25 data.head()
26
27
28 # Annual Income and Spending Score
29
30 # scatter plot
31 plt.figure(figsize=(10, 6))
32 sns.scatterplot(x=data['Annual Income (k$)'],
33                y=data['Spending Score (1-100)'], color='b', s=50)
34 plt.xlabel('Annual Income (k$)', fontsize=15)
35 plt.ylabel('Spending Score (1-100)', fontsize=15)
36 plt.grid()
37
38
39 # centroids for varying k
40 df = []
41 df.append(data['Annual Income (k$)'])
42 df.append(data['Spending Score (1-100)'])
43 df = np.array(df).transpose()
44 scaler = StandardScaler()
45 df_scaled = scaler.fit_transform(df)
46
47 k_set = [int(i) for i in range(1, 11)]
48
49 for k in k_set:
50     gm = GaussianMixture(n_components=k).fit(df_scaled)
51     y_pred = gm.predict(df_scaled)
52
53     plt.figure(figsize=[10, 8])
54     sns.scatterplot(x=df_scaled[:, 0], y=df_scaled[:, 1], c=y_pred, s=100)
55     centers = gm.means_
56     sns.scatterplot(x=centers[:, 0], y=centers[:, 1],
57                    color='.2', marker='*', s=500)
58     plt.grid()
```

```

59     plt.title('K='+str(k), fontsize=15)
60     plt.xlabel('Annual Income (k$)', fontsize=15)
61     plt.ylabel('Spending Score (1-100)', fontsize=15)
62     plt.show()
63
64
65 # plot aic and bic
66 sum_bic = []
67 sum_aic = []
68
69 k_set = range(1, 11)
70 for k in k_set:
71     gm = GaussianMixture(n_components=k).fit(df_scaled)
72     sum_bic.append(gm.bic(df_scaled))
73     sum_aic.append(gm.aic(df_scaled))
74
75 x_axis = np.arange(1, 11)
76 plt.figure(figsize=[10, 6])
77 plt.plot(x_axis, sum_aic, 'b-*', label='AIC')
78 plt.plot(x_axis, sum_bic, 'r-o', label='BIC')
79 plt.grid()
80 plt.title('AIC and BIC for different numbers of k', fontsize=15)
81 plt.xlabel('K(number of clusters)', fontsize=15)
82 plt.ylabel('Value', fontsize=15)
83 plt.legend(loc='upper right')
84 plt.show()
85
86
87 # plot inertia
88 sse = []
89
90 k_set = range(1, 11)
91 for k in k_set:
92     gm = GaussianMixture(n_components=k).fit(df_scaled)
93     centers = gm.means_
94     sse.append(find_inertia(k, centers, df_scaled))
95
96 x_axis = np.arange(1, 11)
97 plt.figure(figsize=[10, 6])
98 plt.plot(x_axis, sse, 'b-*')
99 plt.grid()
100 plt.title('Elbow Curve', fontsize=15)
101 plt.xlabel('K(number of clusters)', fontsize=15)
102 plt.ylabel('SSE', fontsize=15)
103 plt.show()
104
105
106 # Age and Spending
107
108 # scatter plot
109 plt.figure(figsize=(10, 6))
110 sns.scatterplot(
111     x=data['Age'], y=data['Spending Score (1-100)'], color='b', s=50)
112 plt.xlabel('Age', fontsize=15)
113 plt.ylabel('Spending Score (1-100)', fontsize=15)
114 plt.grid()
115
116
117 # centroids for varying k
118 df = []
119 df.append(data['Age'])
120 df.append(data['Spending Score (1-100)'])
121 df = np.array(df).transpose()
122 scaler = StandardScaler()
123 df_scaled = scaler.fit_transform(df)
124
125 k_set = [int(i) for i in range(1, 11)]
126
127 for k in k_set:
128     gm = GaussianMixture(n_components=k).fit(df_scaled)
129     y_pred = gm.predict(df_scaled)
130
131     plt.figure(figsize=[10, 8])
132     sns.scatterplot(x=df_scaled[:, 0], y=df_scaled[:, 1], c=y_pred, s=100)
133     centers = gm.means_

```

```

134     sns.scatterplot(x=centers[:, 0], y=centers[:, 1],
135                    color='.2', marker='*', s=500)
136     plt.grid()
137     plt.title('K='+str(k), fontsize=15)
138     plt.xlabel('Age', fontsize=15)
139     plt.ylabel('Spending Score (1-100)', fontsize=15)
140     plt.show()
141
142
143 # plot aic and bic
144 sum_bic = []
145 sum_aic = []
146
147 k_set = range(1, 11)
148 for k in k_set:
149     gm = GaussianMixture(n_components=k).fit(df_scaled)
150     sum_bic.append(gm.bic(df_scaled))
151     sum_aic.append(gm.aic(df_scaled))
152
153 x_axis = np.arange(1, 11)
154 plt.figure(figsize=[10, 6])
155 plt.plot(x_axis, sum_aic, 'b-*', label='AIC')
156 plt.plot(x_axis, sum_bic, 'r-o', label='BIC')
157 plt.grid()
158 plt.title('AIC and BIC for different numbers of k', fontsize=15)
159 plt.xlabel('K(number of clusters)', fontsize=15)
160 plt.ylabel('Value', fontsize=15)
161 plt.legend(loc='upper right')
162 plt.show()
163
164
165 # plot inertia
166 sse = []
167
168 k_set = range(1, 11)
169 for k in k_set:
170     gm = GaussianMixture(n_components=k).fit(df_scaled)
171     centers = gm.means_
172     sse.append(find_inertia(k, centers, df_scaled))
173
174 x_axis = np.arange(1, 11)
175 plt.figure(figsize=[10, 6])
176 plt.plot(x_axis, sse, 'b-*')
177 plt.grid()
178 plt.title('Elbow Curve', fontsize=15)
179 plt.xlabel('K(number of clusters)', fontsize=15)
180 plt.ylabel('SSE', fontsize=15)
181 plt.show()
182
183
184 # Age and Annual Income
185
186 # scatter plot
187 plt.figure(figsize=(10, 6))
188 sns.scatterplot(x=data['Age'], y=data['Annual Income (k$)'], color='b', s=50)
189 plt.xlabel('Age', fontsize=15)
190 plt.ylabel('Annual Income (k$)', fontsize=15)
191 plt.grid()
192
193
194 # centroids for varying k
195 df = []
196 df.append(data['Age'])
197 df.append(data['Annual Income (k$)'])
198 df = np.array(df).transpose()
199 scaler = StandardScaler()
200 df_scaled = scaler.fit_transform(df)
201
202 k_set = [int(i) for i in range(1, 11)]
203
204 for k in k_set:
205     gm = GaussianMixture(n_components=k).fit(df_scaled)
206     y_pred = gm.predict(df_scaled)
207
208     plt.figure(figsize=[10, 8])

```

```

209     sns.scatterplot(x=df_scaled[:, 0], y=df_scaled[:, 1], c=y_pred, s=100)
210     centers = gm.means_
211     sns.scatterplot(x=centers[:, 0], y=centers[:, 1],
212                    color='.2', marker='*', s=500)
213     plt.grid()
214     plt.title('K='+str(k), fontsize=15)
215     plt.xlabel('Age', fontsize=15)
216     plt.ylabel('Annual Income (k$)', fontsize=15)
217     plt.show()
218
219
220 # plot aic and bic
221 sum_bic = []
222 sum_aic = []
223
224 k_set = range(1, 11)
225 for k in k_set:
226     gm = GaussianMixture(n_components=k).fit(df_scaled)
227     sum_bic.append(gm.bic(df_scaled))
228     sum_aic.append(gm.aic(df_scaled))
229
230 x_axis = np.arange(1, 11)
231 plt.figure(figsize=[10, 6])
232 plt.plot(x_axis, sum_aic, 'b-*', label='AIC')
233 plt.plot(x_axis, sum_bic, 'r-o', label='BIC')
234 plt.grid()
235 plt.title('AIC and BIC for different numbers of k', fontsize=15)
236 plt.xlabel('K(number of clusters)', fontsize=15)
237 plt.ylabel('Value', fontsize=15)
238 plt.legend(loc='upper right')
239 plt.show()
240
241
242 # plot inertia
243 sse = []
244
245 k_set = range(1, 11)
246 for k in k_set:
247     gm = GaussianMixture(n_components=k).fit(df_scaled)
248     centers = gm.means_
249     sse.append(find_inertia(k, centers, df_scaled))
250
251 x_axis = np.arange(1, 11)
252 plt.figure(figsize=[10, 6])
253 plt.plot(x_axis, sse, 'b-*')
254 plt.grid()
255 plt.title('Elbow Curve', fontsize=15)
256 plt.xlabel('K(number of clusters)', fontsize=15)
257 plt.ylabel('SSE', fontsize=15)
258 plt.show()
259
260
261 # Gender and Annual Income
262
263 # scatter plot
264 plt.figure(figsize=(10, 6))
265 sns.scatterplot(x=data['Gender'],
266                y=data['Annual Income (k$)'], color='b', s=50)
267 plt.xlabel('Gender', fontsize=15)
268 plt.ylabel('Annual Income (k$)', fontsize=15)
269 plt.grid()
270
271
272 # mean and std for male
273 print(np.mean(data[data['Gender'] == 'Male']['Annual Income (k$)']))
274 print(np.std(data[data['Gender'] == 'Male']['Annual Income (k$)']))
275
276
277 # mean and std for female
278 print(np.mean(data[data['Gender'] == 'Female']['Annual Income (k$)']))
279 print(np.std(data[data['Gender'] == 'Female']['Annual Income (k$)']))
280
281
282 # Gender and Spending Score
283

```

```

284 plt.figure(figsize=(10, 6))
285 sns.scatterplot(x=data['Gender'],
286                y=data['Spending Score (1-100)'], color='b', s=50)
287 plt.xlabel('Gender', fontsize=15)
288 plt.ylabel('Spending Score (1-100)', fontsize=15)
289 plt.grid()
290
291 # mean and std for male
292 print(np.mean(data[data['Gender'] == 'Male']['Spending Score (1-100)']))
293 print(np.std(data[data['Gender'] == 'Male']['Spending Score (1-100)']))
294
295 # mean and std for female
296 print(np.mean(data[data['Gender'] == 'Female']['Spending Score (1-100)']))
297 print(np.std(data[data['Gender'] == 'Female']['Spending Score (1-100)']))

```

Listing 2: Question 2

Result

Annual Income (k\$) and Spending Score (1-100)

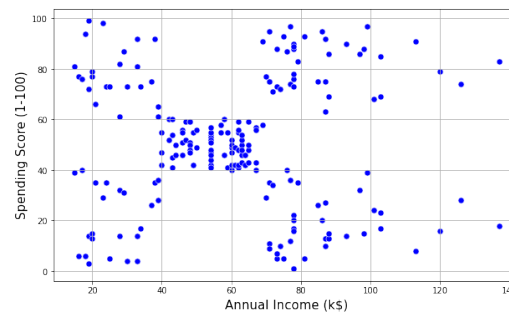


Figure 14: Scatter Plot



Figure 15: varying K

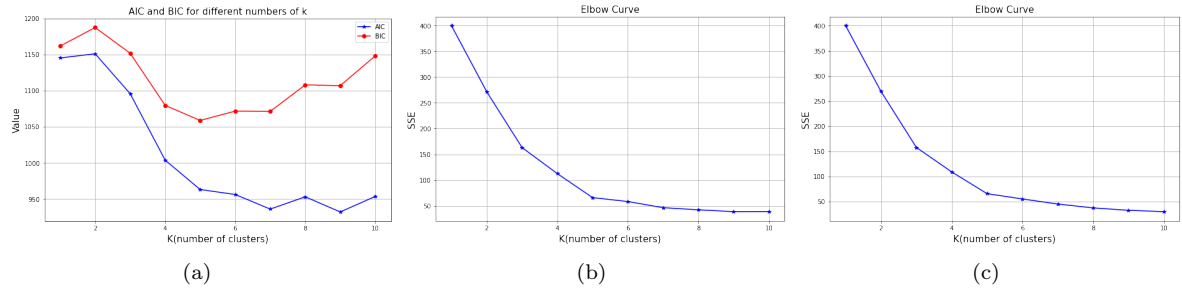


Figure 16: Error Plots (a) AIC and BIC for GMM (b) SSE for GMM (c) SSE for K-means

Age and Spending Score (1-100)

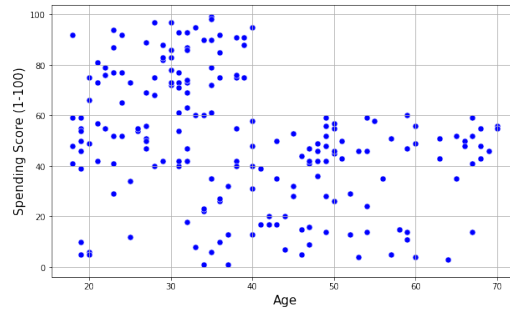


Figure 17: Scatter Plot

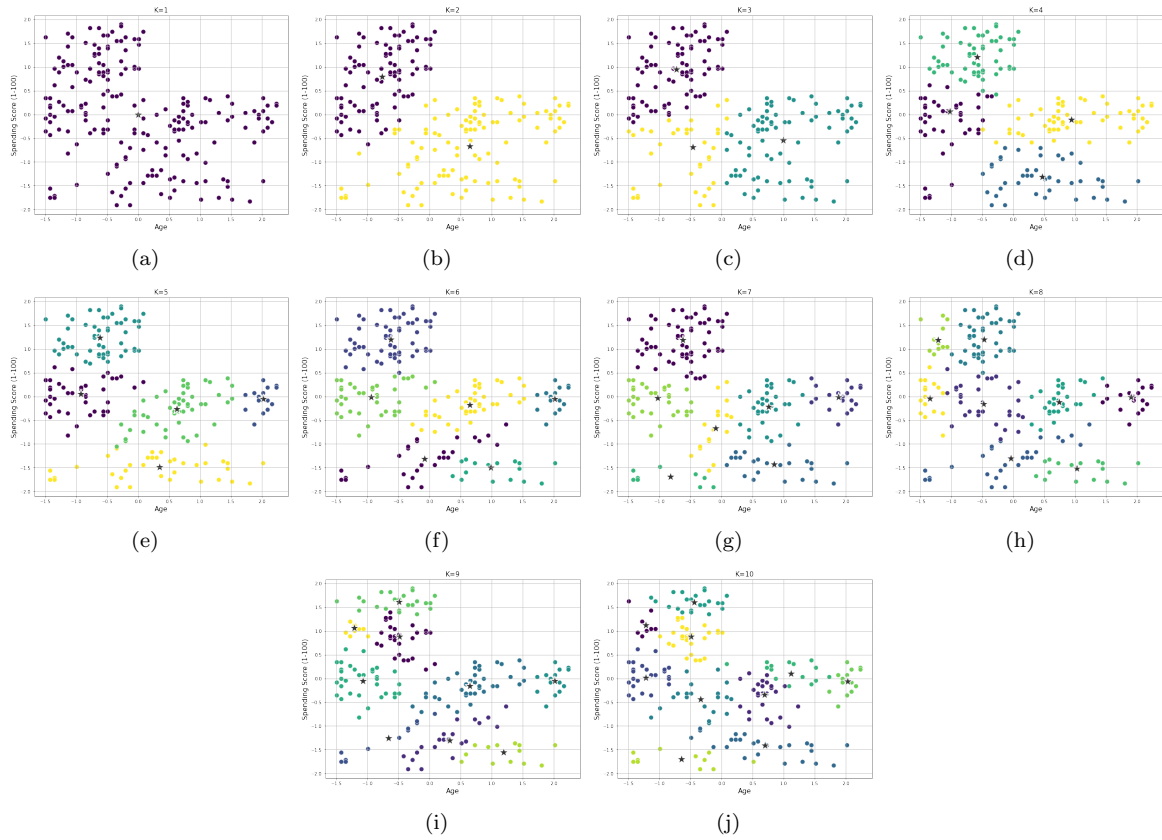


Figure 18: varying K

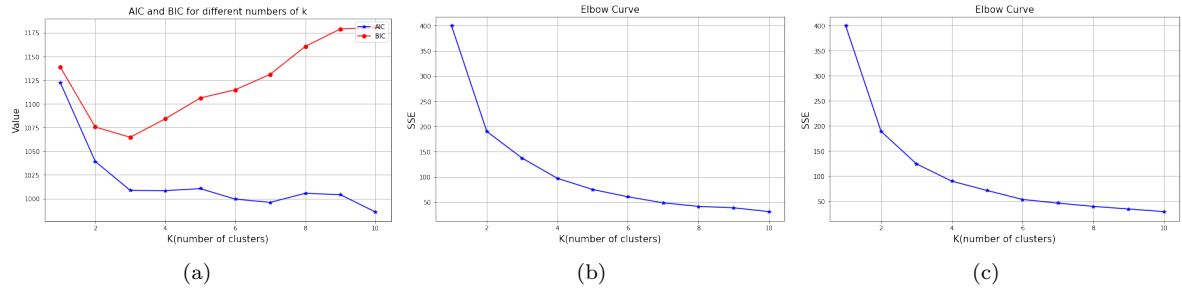


Figure 19: Error Plots (a) AIC and BIC for GMM (b) SSE for GMM (c) SSE for K-means

Age and Annual Income (k\$)

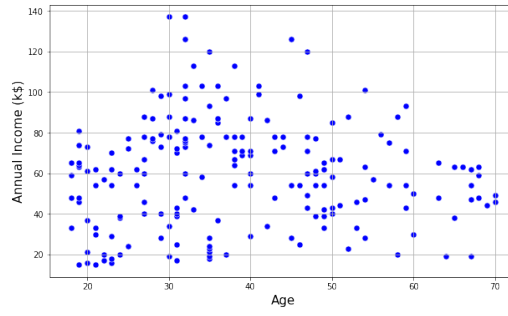


Figure 20: Scatter Plot

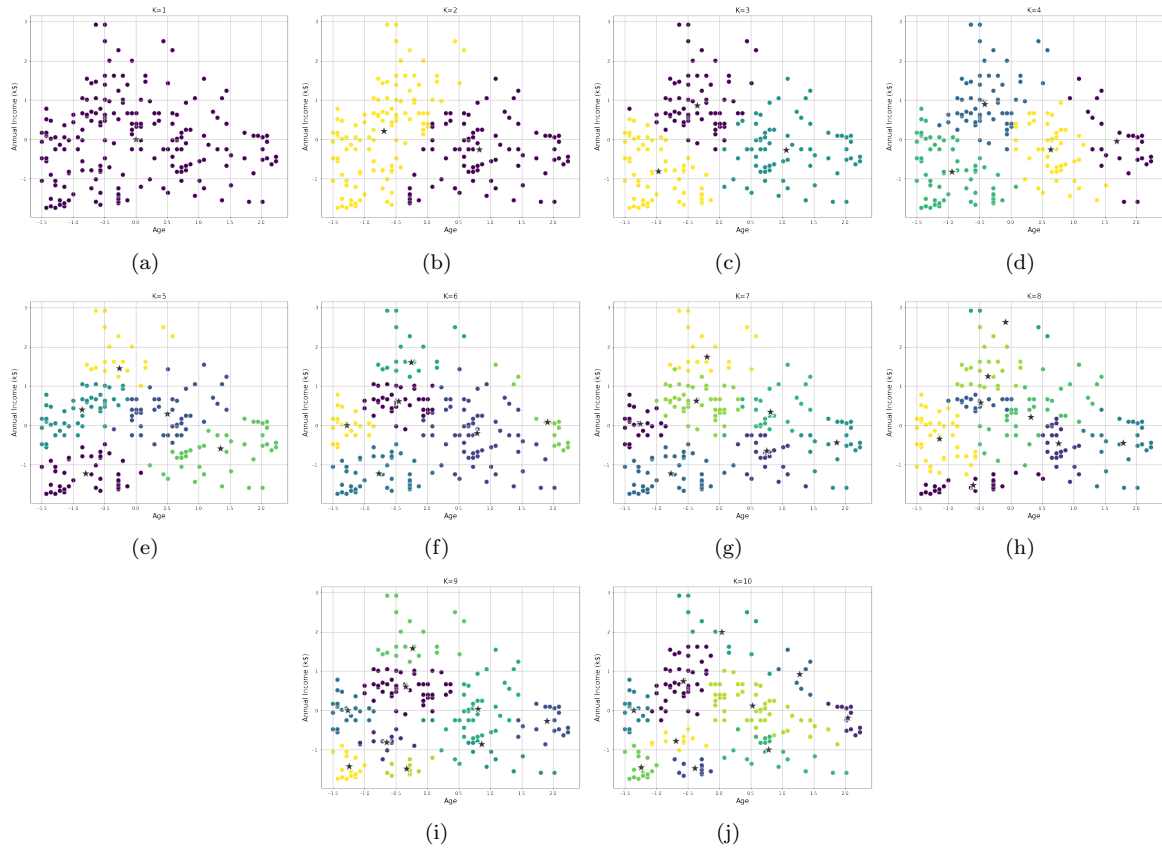


Figure 21: varying K

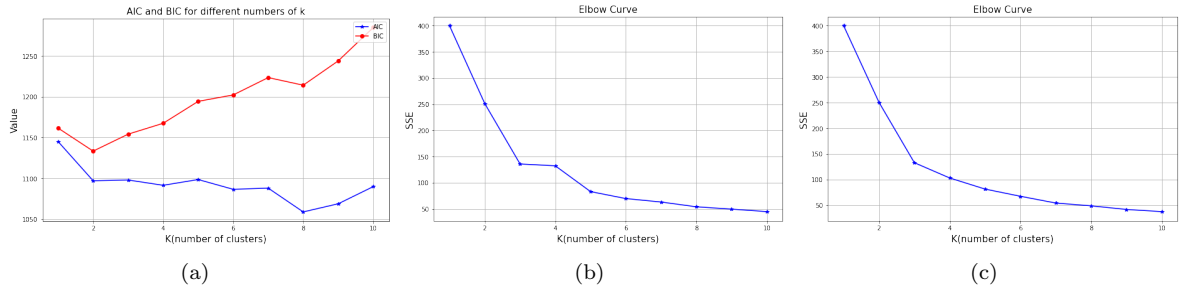


Figure 22: Error Plots (a) AIC and BIC for GMM (b) SSE for GMM (c) SSE for K-means

Gender and Annual Income (k\$)



Figure 23: Scatter Plot

	Male	Female
Mean	62.23	59.25
Std	26.49	25.90

Table 1: Mean and Std for Annual Income (k\$)

Gender and Spending Score (1-100)



Figure 24: Scatter Plot

	Male	Female
Mean	48.51	51.53
Std	27.74	24.01

Table 2: Mean and Std for Spending Score (1-100)

Observation/ Justification

- When we take Age and Spending Score, we see from the initial scatter plot that there are approximately 5 clusters forming. Using GMM, we vary the number of clusters and find with the help of the AIC-BIC plot and SSE plot that the optimal number of 5, which is the same as our intuitive guess. In terms of the data presented, we can conclude that Spending Score is maximum when Income is either maximum or minimum. The same pattern can be observed when we look at minimum Spending Score bracket. However, when income is in between the two extremes, the Spending Score is also around its mean.
- For Age and Spending Score, we can observe two clusters. However, when we use GMM, we find that the AIC and BIC values are minimum for $k = 3$. The same result can be observed in the SSE plot. This cluster could have been identified by simply observing the scatter plot. This shows that for customers with age less than the average age of the consumer pool, spending varies from minimum to maximum but for customers with age greater than the average, we find their Spending Score to be less than the average.
- In case of Age and Annual Income, there is no pattern observable to the naked eye based on the scatter plot. However, as we go through the analysis, we see there are two clusters (AIC and BIC are minimum for $k = 2$). Based on the scatter plot for the same, we can see that, on an average, consumers with age greater than the mean earn less than the other half.
- When we look at Gender and Annual Income, we see that mean income for men is greater than mean income for women with the standard deviations following the opposite trend.
- For Gender and Spending Score, we find the mean spending score of women to be greater than that of men while the standard deviations follow the opposite trend.
- When SSE is calculated using the approximate method explained in the previous question, we get results consistent that are with the K-means clustering.