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

```
# Q1

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import random
%matplotlib inline
from sklearn.preprocessing import StandardScaler
import seaborn as sns
from sklearn.metrics import mean_squared_error as mse
from sklearn.metrics import accuracy_score
from scipy.stats import mode
from sklearn.cluster import KMeans
from sklearn.mixture import GaussianMixture
import warnings
warnings.filterwarnings('ignore')
from scipy.stats import multivariate_normal as mvn


#Ref: https://scikit-learn.org/stable/auto_examples/mixture/plot_gmm_covariances.html#sphx-glr
    -auto-examples-mixture-plot-gmm-covariances-py
from matplotlib.patches import Ellipse
def plot_2D(data, centers, y_pred, gmm, axes_label):
    plt.figure(figsize=[10,6])
    ax = plt.gca()

```

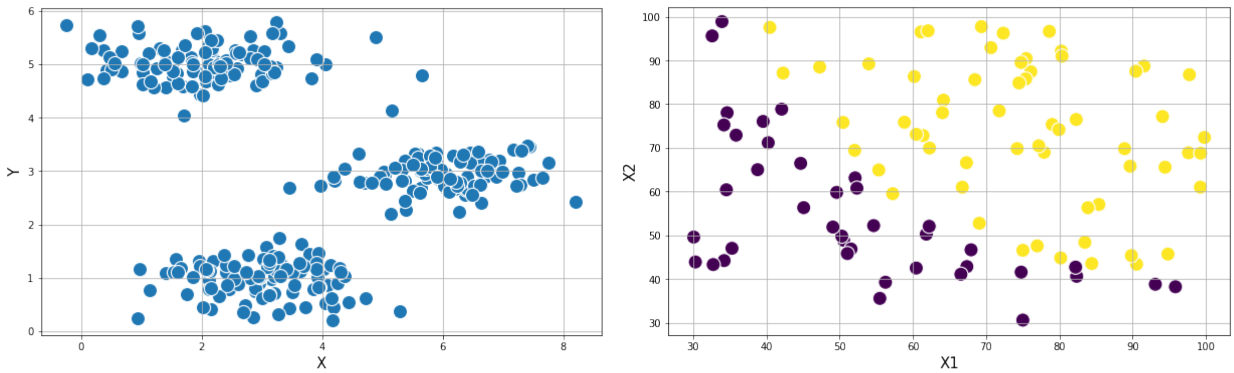
```
    # Plot points according to the value of their max probability
    size = 100 * gmm.predict_proba(data).max(1) ** 2  # square emphasizes differences
    ax.scatter(data[:, 0], data[:, 1], c=y_pred, cmap='viridis', s=size)

    # Plot the cluster area depending on the mean and covariance
    for pos, covar, w in zip(gmm.means_, gmm.covariances_, gmm.weights_):
        # Convert covariance to principal axes
        U, s, Vt = np.linalg.svd(covar)
        angle = 180*(np.arctan2(U[1, 0], U[0, 0]))/np.pi
        width, height = 2 * np.sqrt(s)

        # Draw the Ellipse for n*sigma
        for n in range(1, 4):
            ax.add_patch(Ellipse(pos, n*width, n*height, angle, alpha=0.1*(w/gmm.weights_.max
    ())))

    ax.axis('equal')
    plt.grid()
    plt.title('K='+str(i),fontsize=15)
    plt.xlabel(axes_label[0],fontsize=15)
    plt.ylabel(axes_label[1],fontsize=15)
    plt.show()

def find_inertia(k,centroids,data):
    sse_error=0
    for i in range(len(data)):
        min_val= np.inf
        for j in range(k):
            temp = (data[i][0]-centroids[j][0])**2 + (data[i][1]-centroids[j][1])**2
            min_val = min(min_val,temp)
        sse_error = sse_error + min_val
    return sse_error

# Q1 a)

data = pd.read_excel('Question2a.xlsx')
data.head()

plt.figure(figsize=(10,6))
sns.scatterplot(data['x'],data['y'],s=200)
plt.xlabel('X',fontsize=15)
plt.ylabel('Y',fontsize=15)
plt.grid()

# Standardize data
df=[]
df.append(data['x'])
df.append(data['y'])
df=np.array(df).transpose()
scaler = StandardScaler()
df_scaled = scaler.fit_transform(df)

#Q1a 3) Visualize the cluster formation for each k

SSE_q1a = []
SSE_q1a_kmeans=[]
q1a_aic = []
q1a_bic = []
axes_label=['X','Y']
for i in range(1,11):
  # GMM
  gmm = GaussianMixture(n_components=i, covariance_type='full', random_state=0).fit(df_scaled)
  y_pred = gmm.predict(df_scaled)
  probs = gmm.predict_proba(df_scaled)

  # K-Means
  kmeans = KMeans(n_clusters = i, init='k-means++')
  kmeans.fit(df_scaled)

  # Creating plot
  centers = np.zeros((i,2))
  for j in range(i):
      density = mvn(cov=gmm.covariances_[j], mean=gmm.means_[j]).logpdf(df_scaled)
      centers[j, :] = df_scaled[np.argmax(density)]
  plot_2D(df_scaled, centers, y_pred, gmm, axes_label)
```

```
  #Calculate SSE
  SSE_q1a.append(find_inertia(i,centers,df_scaled))
  SSE_q1a_kmeans.append(kmeans.inertia_)
  q1a_aic.append(gmm.aic(df_scaled))
  q1a_bic.append(gmm.bic(df_scaled))

#Q1a 4) Plot elbow curve

x_axis=np.arange(1,11)
plt.figure(figsize=[10,6])
plt.plot(x_axis,q1a_aic,'b-*')
plt.plot(x_axis,q1a_bic,'k-o')
plt.grid()
plt.legend(['AIC','BIC'],fontsize=15)
plt.xlabel('K(number of clusters)',fontsize=15)

x_axis=np.arange(1,11)
plt.figure(figsize=[10,6])
plt.plot(x_axis,SSE_q1a,'b-*')
plt.plot(x_axis,SSE_q1a_kmeans,'k-o')
plt.grid()
plt.ylabel('SSE',fontsize=15)
plt.xlabel('K(number of clusters)',fontsize=15)
plt.title('Elbow Curve',fontsize=15)
plt.legend(['GMM','K-Means'],fontsize=15)

# Q1b

data2 = pd.read_excel('Question2b.xls')
data2.head()

plt.figure(figsize=(10,6))
sns.scatterplot(data2['x1'],data2['x2'],s=200,c=data2['y'])
plt.xlabel('X1',fontsize=15)
plt.ylabel('X2',fontsize=15)
plt.grid()

# Standardize data
df2=[]
df2.append(data2['x1'])
df2.append(data2['x2'])
df2=np.array(df2).transpose()
scaler = StandardScaler()
df_scaled2 = scaler.fit_transform(df2)

#Q1b 3) Visualize the cluster formation for each k

SSE_q1b = []
SSE_q1b_kmeans=[]
q1b_aic=[]
q1b_bic=[]
n_iter = []
axes_label=['X1','X2']
for i in range(1,11):
  # GMM
  gmm = GaussianMixture(n_components=i, covariance_type='full', random_state=0).fit(df_scaled2
    )
  y_pred = gmm.predict(df_scaled2)
  probs = gmm.predict_proba(df_scaled2)
  n_iter.append(gmm.n_iter_)

  # K-Means
  kmeans = KMeans(n_clusters = i, init='k-means++')
  kmeans.fit(df_scaled2)

  # Creating plot
  centers = np.zeros((i,2))
  for j in range(i):
      density = mvn(cov=gmm.covariances_[j], mean=gmm.means_[j]).logpdf(df_scaled2)
      centers[j, :] = df_scaled2[np.argmax(density)]
  plot_2D(df_scaled2, centers, y_pred, gmm, axes_label)

  if i==2:
    print('Accuracy for k=2: ',accuracy_score(data2['y'],y_pred))
```

```
  #Calculate SSE
  SSE_q1b.append(find_inertia(i,centers,df_scaled2))
  SSE_q1b_kmeans.append(kmeans.inertia_)
  q1b_aic.append(gmm.aic(df_scaled2))
  q1b_bic.append(gmm.bic(df_scaled2))

#Q1b 4) Plot elbow curve

x_axis=np.arange(1,11)
plt.figure(figsize=[10,6])
plt.plot(x_axis,q1b_aic,'b-*')
plt.plot(x_axis,q1b_bic,'k-o')
plt.grid()
plt.legend(['AIC','BIC'],fontsize=15)
plt.xlabel('K(number of clusters)',fontsize=15)

x_axis=np.arange(1,11)
plt.figure(figsize=[10,6])
plt.plot(x_axis,SSE_q1b,'b-*')
plt.plot(x_axis,SSE_q1b_kmeans,'k-o')
plt.grid()
plt.ylabel('SSE',fontsize=15)
plt.xlabel('K(number of clusters)',fontsize=15)
plt.title('Elbow Curve',fontsize=15)
plt.legend(['GMM','K-Means'],fontsize=15)
```

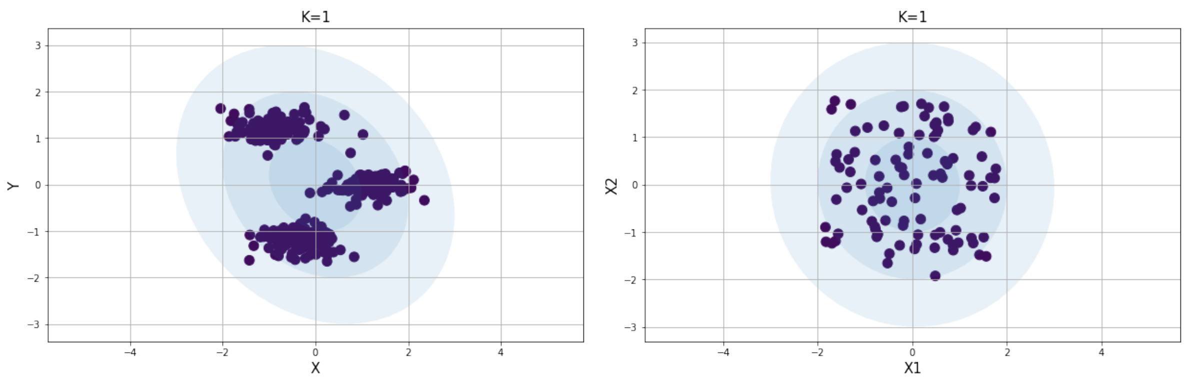
Listing 1: Question 1

Result



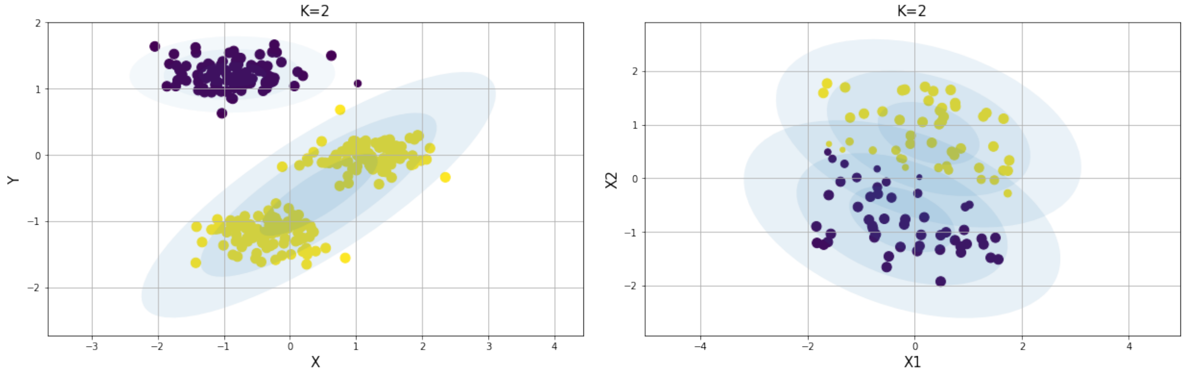
(a) (b)

Figure 1: Initial data



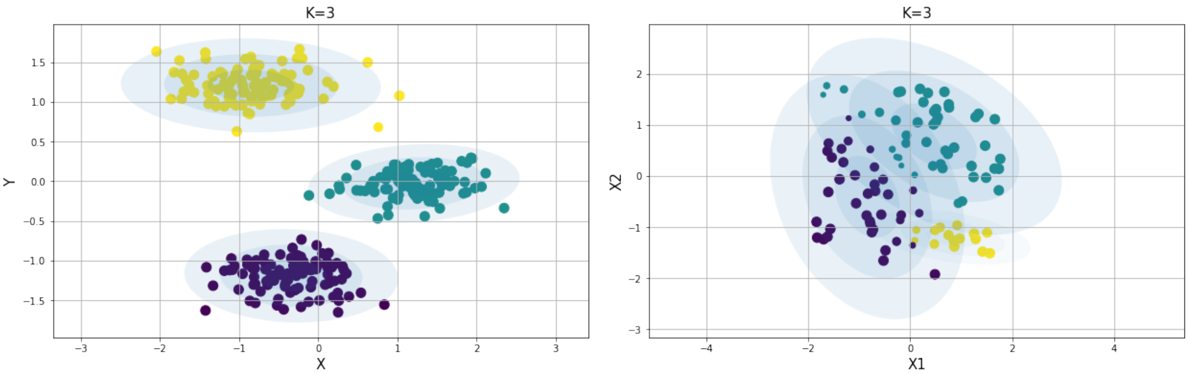
(a) (b)

Figure 2: K = 1



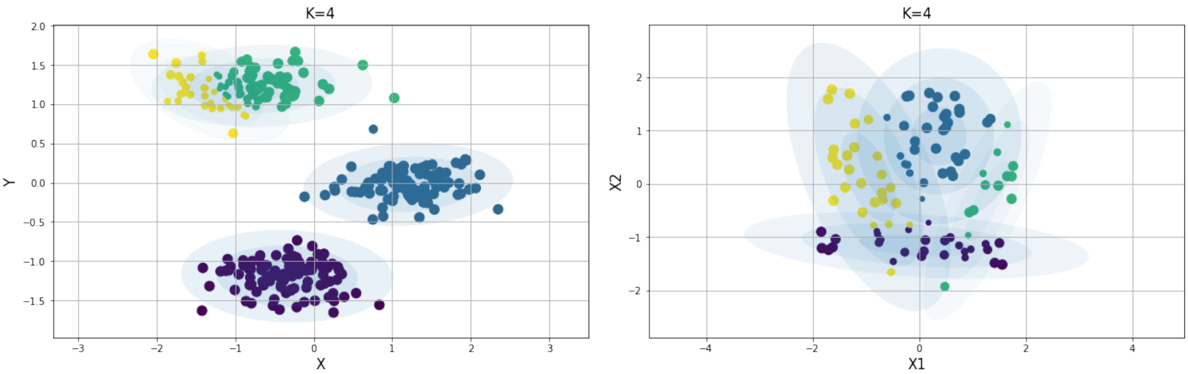
(a) (b)

Figure 3: K = 2



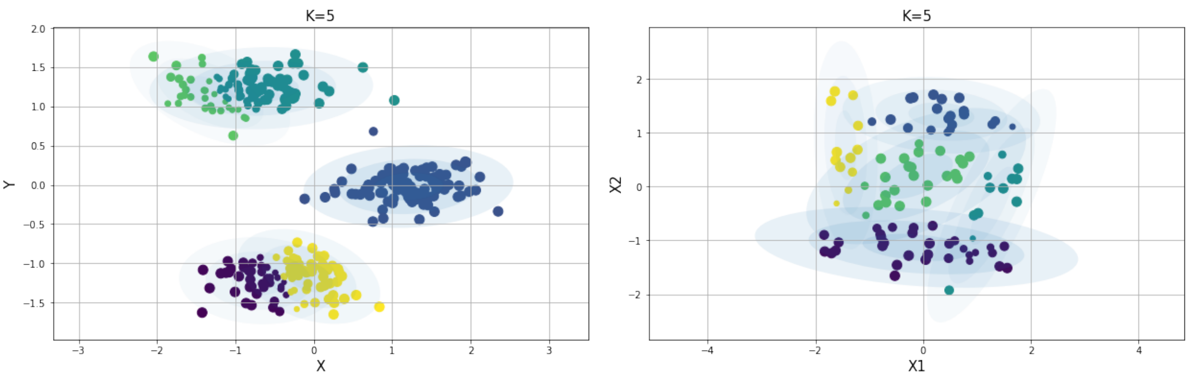
(a) (b)

Figure 4: K = 3



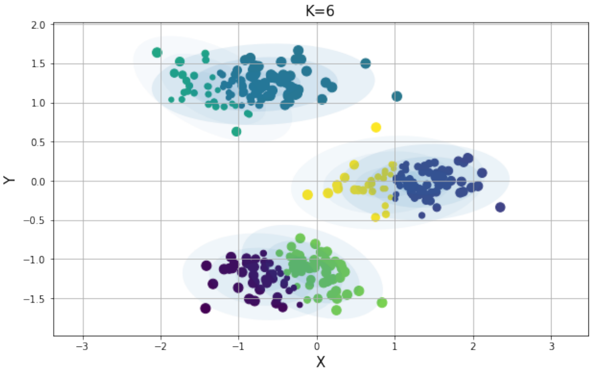
(a) (b)

Figure 5: K = 4

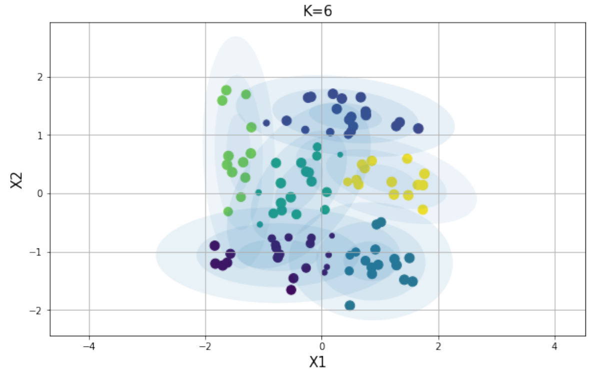


(a) (b)

Figure 6: K = 5

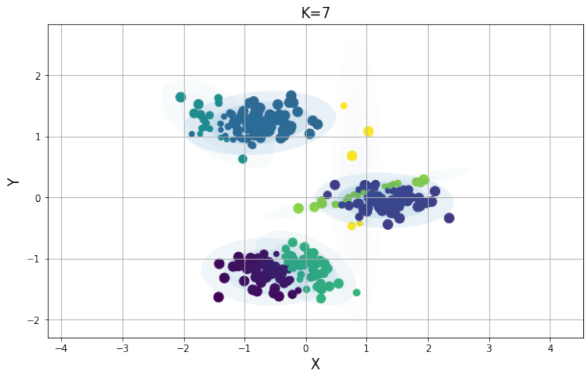


(a)

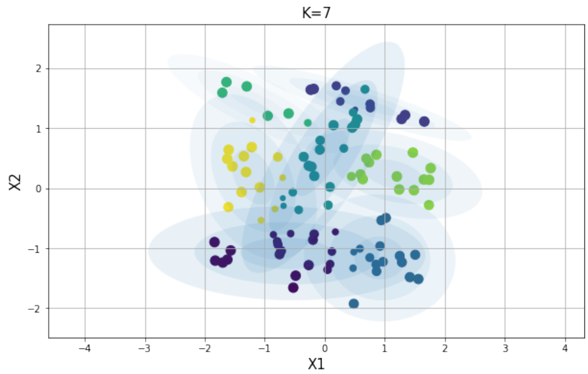


(b)

Figure 7: K = 6

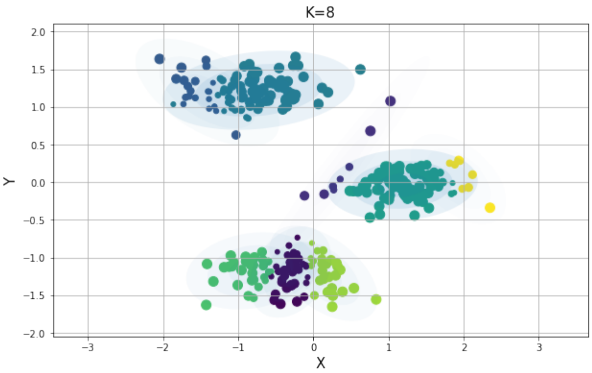


(a)

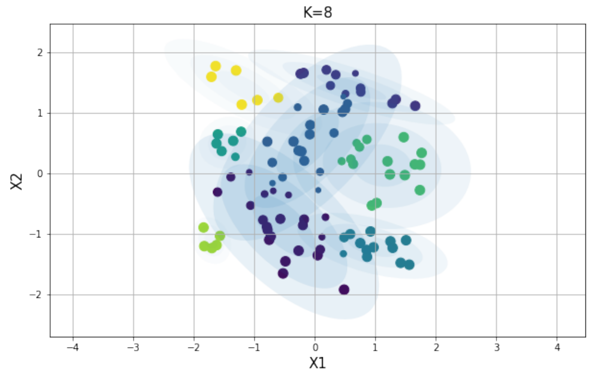


(b)

Figure 8: K = 7

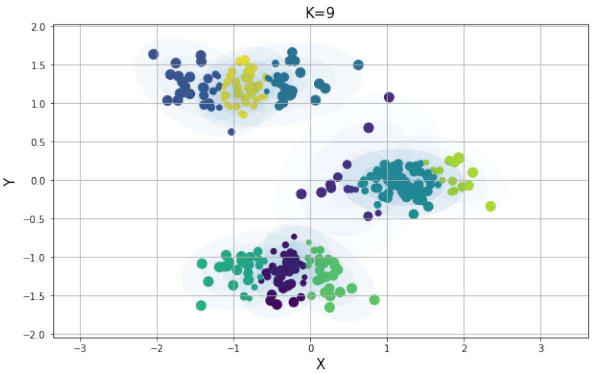


(a)

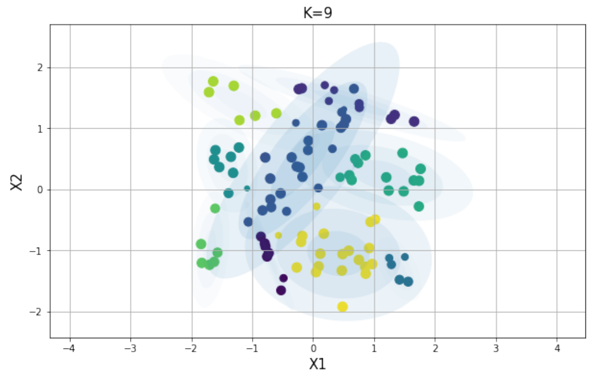


(b)

Figure 9: K = 8

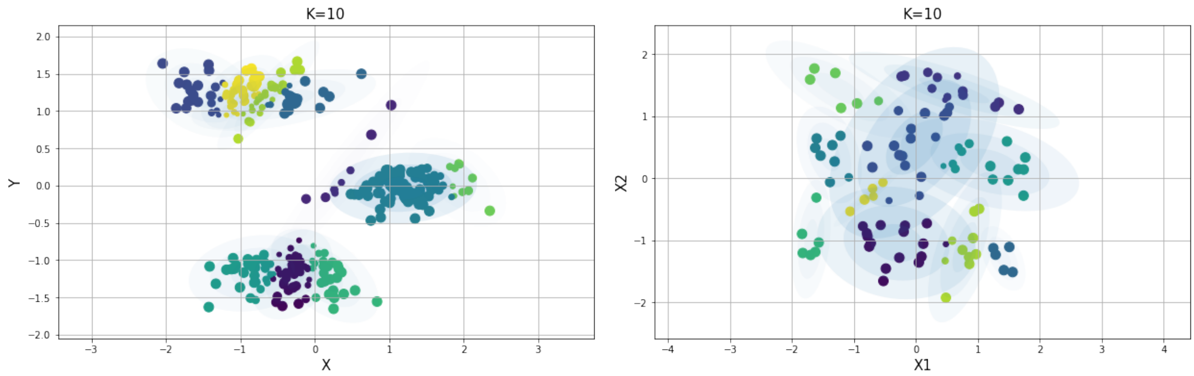


(a)



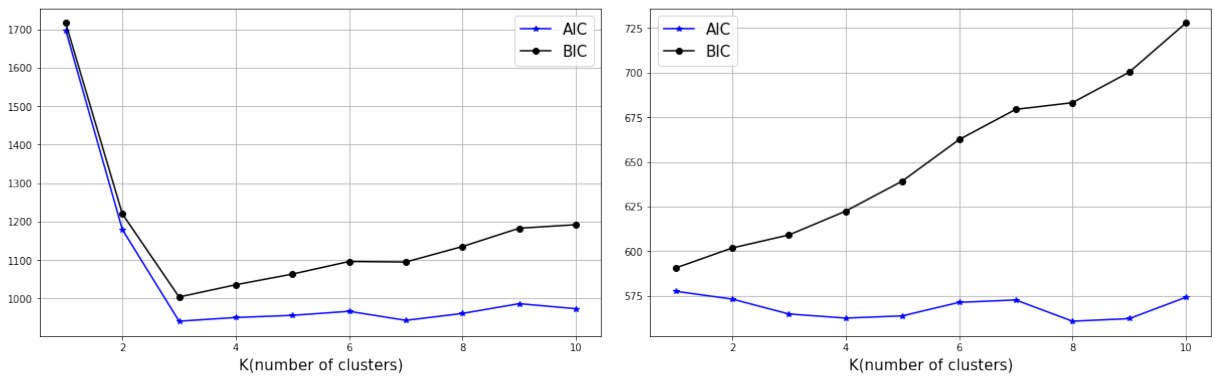
(b)

Figure 10: K = 9



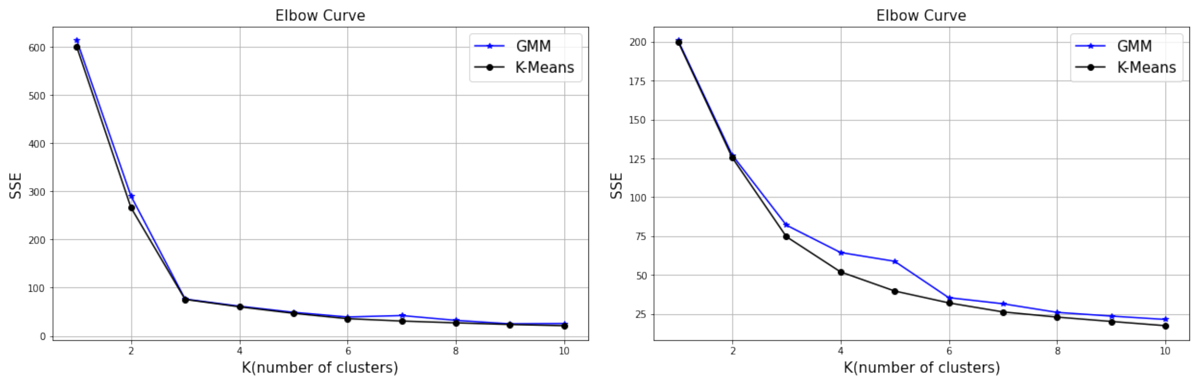
(a) (b)

Figure 11: K = 10



(a) (b)

Figure 12: Elbow curve(AIC/BIC)



(a) (b)

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

```
# import libaries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.mixture import GaussianMixture
from sklearn.preprocessing import StandardScaler


# function to find inertia for GMM
def find_inertia(k, centroids, data):
    sse_error = 0
    for i in range(len(data)):
        min_val = np.inf
        for j in range(k):
            temp = (data[i][0]-centroids[j][0])**2 + \
                (data[i][1]-centroids[j][1])**2
            min_val = min(min_val, temp)
        sse_error = sse_error + min_val
    return sse_error


# import datafile
data = pd.read_csv('dataset3.csv')
data.head()


# Annual Income and Spending Score

# scatter plot
plt.figure(figsize=(10, 6))
sns.scatterplot(x=data['Annual Income (k$)'],
                y=data['Spending Score (1-100)'], color='b', s=50)
plt.xlabel('Annual Income (k$)', fontsize=15)
plt.ylabel('Spending Score (1-100)', fontsize=15)
plt.grid()


# centroids for varying k
df = []
df.append(data['Annual Income (k$)'])
df.append(data['Spending Score (1-100)'])
df = np.array(df).transpose()
scaler = StandardScaler()
df_scaled = scaler.fit_transform(df)

k_set = [int(i) for i in range(1, 11)]

for k in k_set:
    gm = GaussianMixture(n_components=k).fit(df_scaled)
    y_pred = gm.predict(df_scaled)

    plt.figure(figsize=[10, 8])
    sns.scatterplot(x=df_scaled[:, 0], y=df_scaled[:, 1], c=y_pred, s=100)
    centers = gm.means_
    sns.scatterplot(x=centers[:, 0], y=centers[:, 1],
                    color='.2', marker='*', s=500)
    plt.grid()
```

```
    plt.title('K='+str(k), fontsize=15)
    plt.xlabel('Annual Income (k$)', fontsize=15)
    plt.ylabel('Spending Score (1-100)', fontsize=15)
    plt.show()


# plot aic and bic
sum_bic = []
sum_aic = []

k_set = range(1, 11)
for k in k_set:
    gm = GaussianMixture(n_components=k).fit(df_scaled)
    sum_bic.append(gm.bic(df_scaled))
    sum_aic.append(gm.aic(df_scaled))

x_axis = np.arange(1, 11)
plt.figure(figsize=[10, 6])
plt.plot(x_axis, sum_aic, 'b-*', label='AIC')
plt.plot(x_axis, sum_bic, 'r-o', label='BIC')
plt.grid()
plt.title('AIC and BIC for different numbers of k', fontsize=15)
plt.xlabel('K(number of clusters)', fontsize=15)
plt.ylabel('Value', fontsize=15)
plt.legend(loc='upper right')
plt.show()


# plot inertia
sse = []

k_set = range(1, 11)
for k in k_set:
    gm = GaussianMixture(n_components=k).fit(df_scaled)
    centers = gm.means_
    sse.append(find_inertia(k, centers, df_scaled))

x_axis = np.arange(1, 11)
plt.figure(figsize=[10, 6])
plt.plot(x_axis, sse, 'b-*')
plt.grid()
plt.title('Elbow Curve', fontsize=15)
plt.xlabel('K(number of clusters)', fontsize=15)
plt.ylabel('SSE', fontsize=15)
plt.show()


# Age and Spending

# scatter plot
plt.figure(figsize=(10, 6))
sns.scatterplot(
    x=data['Age'], y=data['Spending Score (1-100)'], color='b', s=50)
plt.xlabel('Age', fontsize=15)
plt.ylabel('Spending Score (1-100)', fontsize=15)
plt.grid()


# centroids for varying k
df = []
df.append(data['Age'])
df.append(data['Spending Score (1-100)'])
df = np.array(df).transpose()
scaler = StandardScaler()
df_scaled = scaler.fit_transform(df)

k_set = [int(i) for i in range(1, 11)]

for k in k_set:
    gm = GaussianMixture(n_components=k).fit(df_scaled)
    y_pred = gm.predict(df_scaled)

    plt.figure(figsize=[10, 8])
    sns.scatterplot(x=df_scaled[:, 0], y=df_scaled[:, 1], c=y_pred, s=100)
    centers = gm.means_
```

```
    sns.scatterplot(x=centers[:, 0], y=centers[:, 1],
                    color='.2', marker='*', s=500)
    plt.grid()
    plt.title('K='+str(k), fontsize=15)
    plt.xlabel('Age', fontsize=15)
    plt.ylabel('Spending Score (1-100)', fontsize=15)
    plt.show()


# plot aic and bic
sum_bic = []
sum_aic = []

k_set = range(1, 11)
for k in k_set:
    gm = GaussianMixture(n_components=k).fit(df_scaled)
    sum_bic.append(gm.bic(df_scaled))
    sum_aic.append(gm.aic(df_scaled))

x_axis = np.arange(1, 11)
plt.figure(figsize=[10, 6])
plt.plot(x_axis, sum_aic, 'b-*', label='AIC')
plt.plot(x_axis, sum_bic, 'r-o', label='BIC')
plt.grid()
plt.title('AIC and BIC for different numbers of k', fontsize=15)
plt.xlabel('K(number of clusters)', fontsize=15)
plt.ylabel('Value', fontsize=15)
plt.legend(loc='upper right')
plt.show()


# plot inertia
sse = []

k_set = range(1, 11)
for k in k_set:
    gm = GaussianMixture(n_components=k).fit(df_scaled)
    centers = gm.means_
    sse.append(find_inertia(k, centers, df_scaled))

x_axis = np.arange(1, 11)
plt.figure(figsize=[10, 6])
plt.plot(x_axis, sse, 'b-*')
plt.grid()
plt.title('Elbow Curve', fontsize=15)
plt.xlabel('K(number of clusters)', fontsize=15)
plt.ylabel('SSE', fontsize=15)
plt.show()


# Age and Annual Income

# scatter plot
plt.figure(figsize=(10, 6))
sns.scatterplot(x=data['Age'], y=data['Annual Income (k$)'], color='b', s=50)
plt.xlabel('Age', fontsize=15)
plt.ylabel('Annual Income (k$)', fontsize=15)
plt.grid()


# centroids for varying k
df = []
df.append(data['Age'])
df.append(data['Annual Income (k$)'])
df = np.array(df).transpose()
scaler = StandardScaler()
df_scaled = scaler.fit_transform(df)

k_set = [int(i) for i in range(1, 11)]

for k in k_set:
    gm = GaussianMixture(n_components=k).fit(df_scaled)
    y_pred = gm.predict(df_scaled)

    plt.figure(figsize=[10, 8])
```

```python
    sns.scatterplot(x=df_scaled[:, 0], y=df_scaled[:, 1], c=y_pred, s=100)
    centers = gm.means_
    sns.scatterplot(x=centers[:, 0], y=centers[:, 1],
                    color='.2', marker='*', s=500)
    plt.grid()
    plt.title('K='+str(k), fontsize=15)
    plt.xlabel('Age', fontsize=15)
    plt.ylabel('Annual Income (k$)', fontsize=15)
    plt.show()


# plot aic and bic
sum_bic = []
sum_aic = []

k_set = range(1, 11)
for k in k_set:
    gm = GaussianMixture(n_components=k).fit(df_scaled)
    sum_bic.append(gm.bic(df_scaled))
    sum_aic.append(gm.aic(df_scaled))

x_axis = np.arange(1, 11)
plt.figure(figsize=[10, 6])
plt.plot(x_axis, sum_aic, 'b-*', label='AIC')
plt.plot(x_axis, sum_bic, 'r-o', label='BIC')
plt.grid()
plt.title('AIC and BIC for different numbers of k', fontsize=15)
plt.xlabel('K(number of clusters)', fontsize=15)
plt.ylabel('Value', fontsize=15)
plt.legend(loc='upper right')
plt.show()


# plot inertia
sse = []

k_set = range(1, 11)
for k in k_set:
    gm = GaussianMixture(n_components=k).fit(df_scaled)
    centers = gm.means_
    sse.append(find_inertia(k, centers, df_scaled))

x_axis = np.arange(1, 11)
plt.figure(figsize=[10, 6])
plt.plot(x_axis, sse, 'b-*')
plt.grid()
plt.title('Elbow Curve', fontsize=15)
plt.xlabel('K(number of clusters)', fontsize=15)
plt.ylabel('SSE', fontsize=15)
plt.show()


# Gender and Annual Income

# scatter plot
plt.figure(figsize=(10, 6))
sns.scatterplot(x=data['Gender'],
                y=data['Annual Income (k$)'], color='b', s=50)
plt.xlabel('Gender', fontsize=15)
plt.ylabel('Annual Income (k$)', fontsize=15)
plt.grid()


# mean and std for male
print(np.mean(data[data['Gender'] == 'Male']['Annual Income (k$)']))
print(np.std(data[data['Gender'] == 'Male']['Annual Income (k$)']))


# mean and std for female
print(np.mean(data[data['Gender'] == 'Female']['Annual Income (k$)']))
print(np.std(data[data['Gender'] == 'Female']['Annual Income (k$)']))


# Gender and Spending Score

```

```
plt.figure(figsize=(10, 6))
sns.scatterplot(x=data['Gender'],
                y=data['Spending Score (1-100)'], color='b', s=50)
plt.xlabel('Gender', fontsize=15)
plt.ylabel('Spending Score (1-100)', fontsize=15)
plt.grid()


# mean and std for male
print(np.mean(data[data['Gender'] == 'Male']['Spending Score (1-100)']))
print(np.std(data[data['Gender'] == 'Male']['Spending Score (1-100)']))


# mean and std for female
print(np.mean(data[data['Gender'] == 'Female']['Spending Score (1-100)']))
print(np.std(data[data['Gender'] == 'Female']['Spending Score (1-100)']))
```

Listing 2: Question 2

Result

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

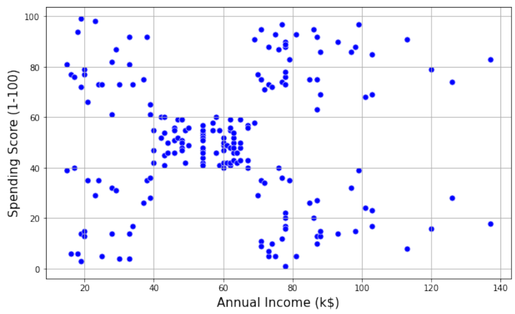
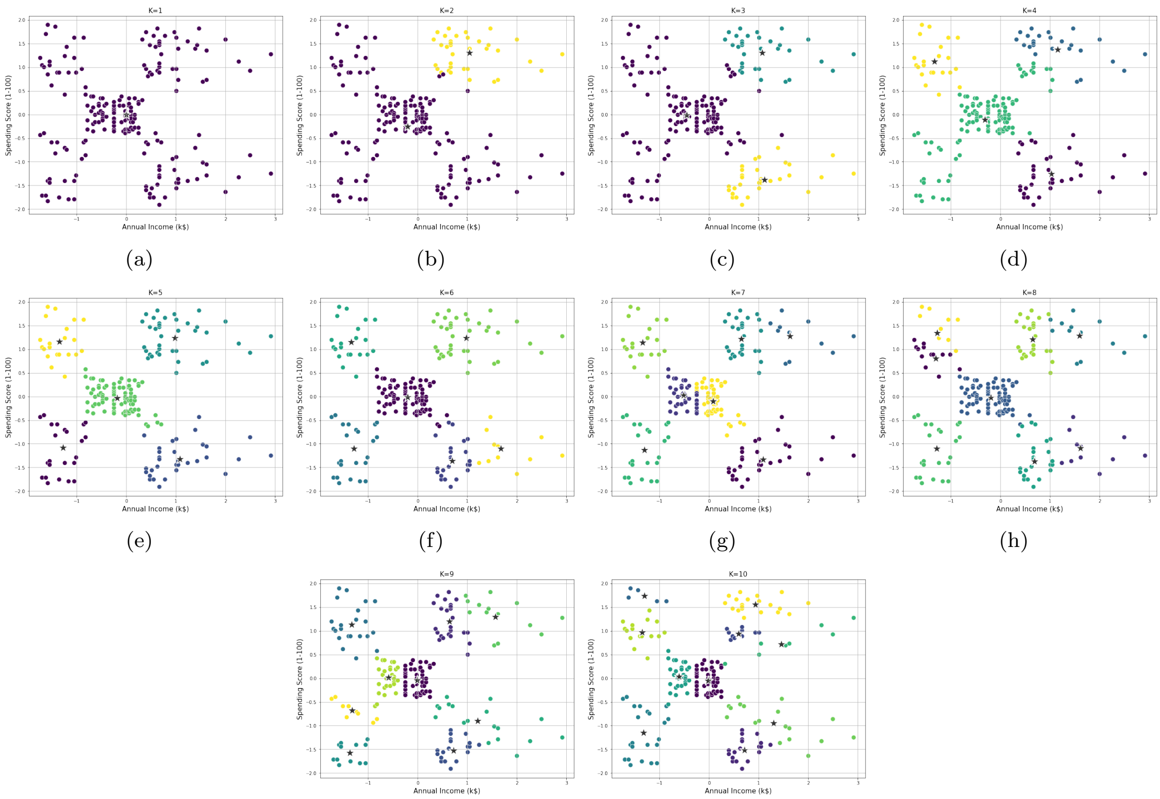


Figure 14: Scatter Plot

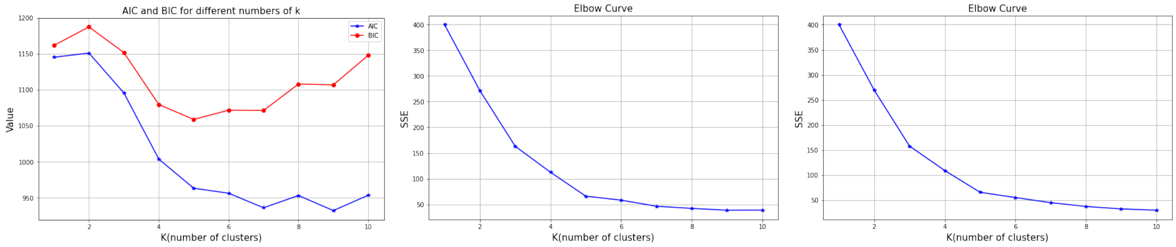


(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j)

Figure 15: varying K



(a) (b) (c)

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)

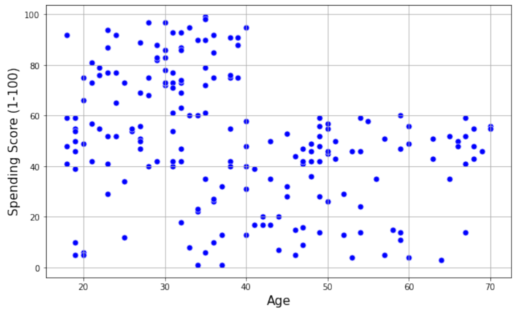
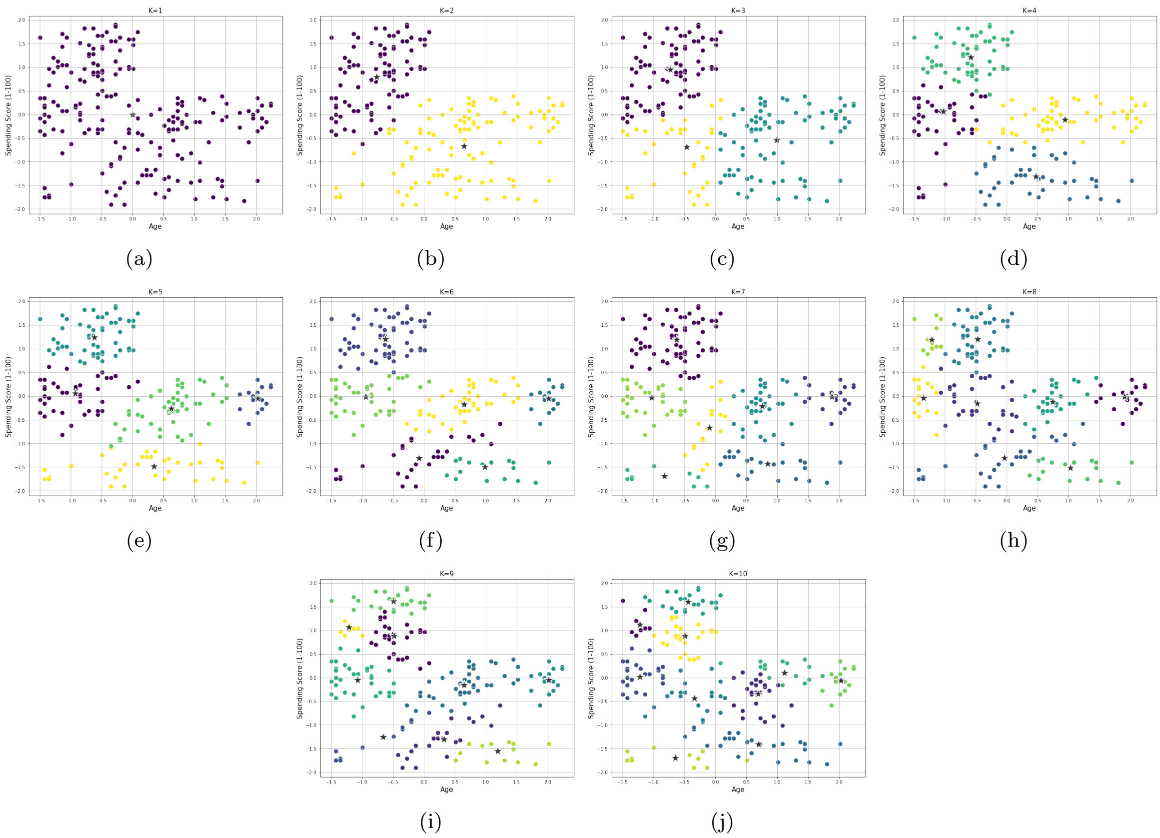
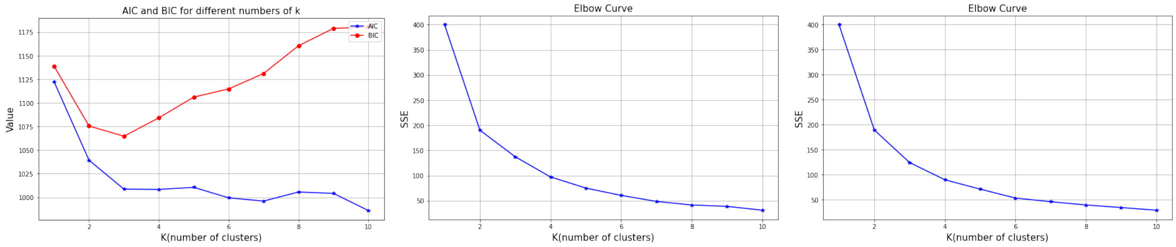


Figure 17: Scatter Plot



(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Figure 18: varying K



(a) (b) (c)

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

Age and Annual Income (k$)

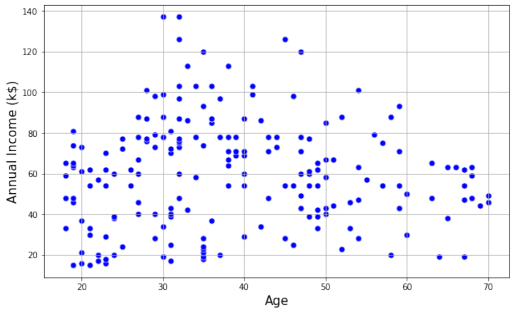
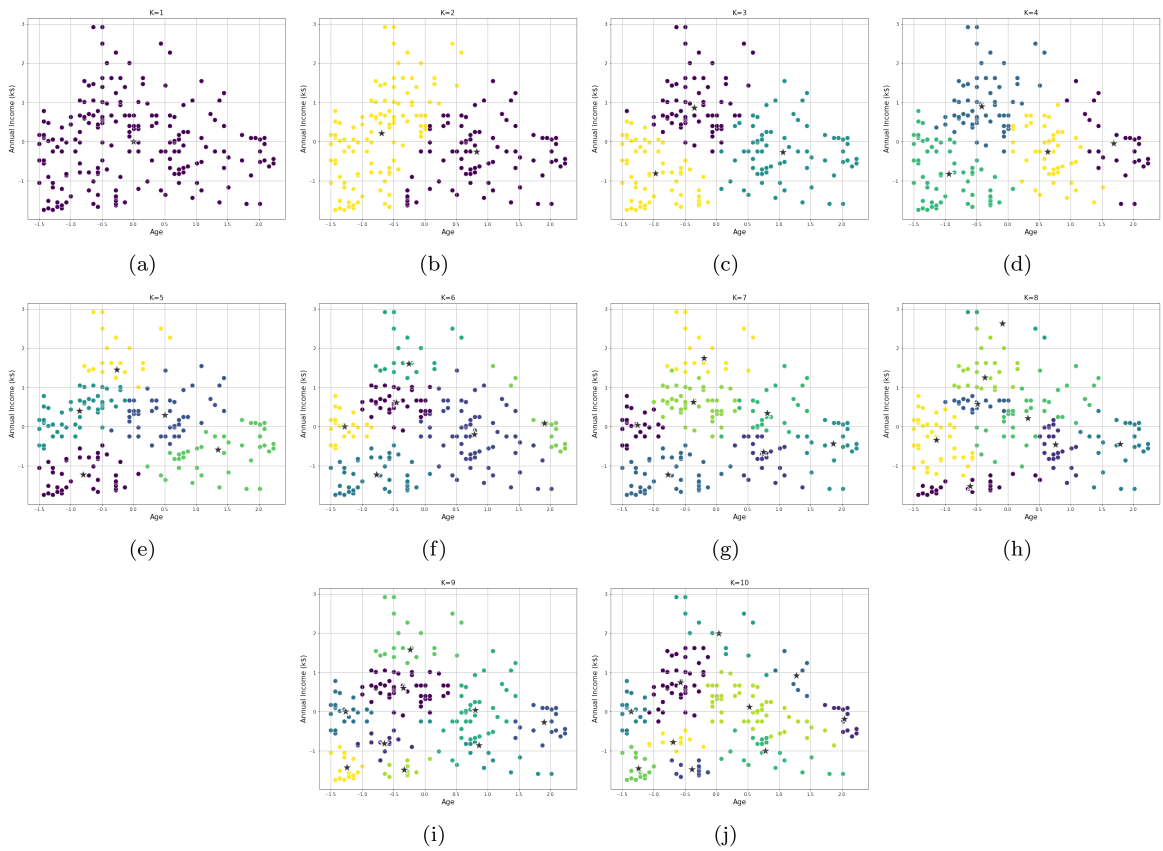
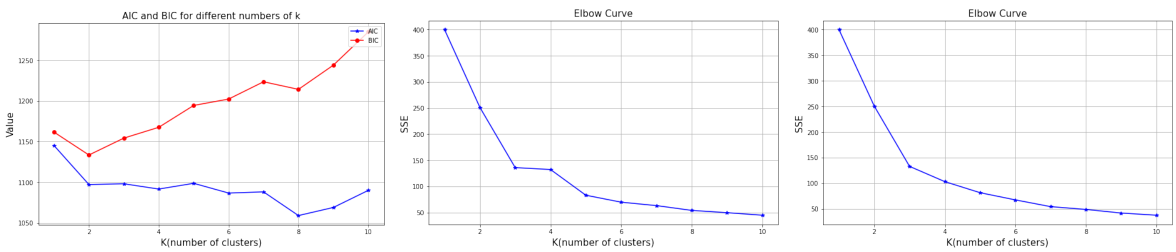


Figure 20: Scatter Plot



(a) (b) (c) (d)
(e) (f) (g) (h)
(i) (j)

Figure 21: varying K



(a) (b) (c)

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 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.