Harshit Mishra(19BCE0799)

CSE3024 – WEB MINING

LAB ASSIGNMENT 10

Ques: Illustrate the K-means clustering to cluster the data points for at least five epochs properly.

1. Use the elbow method to determine the optimal number of clusters.
2. Visualize the clusters.
3. Plot the centroids of each cluster.

Dataset Used: Shopping-data.csv from kaggle

Procedure:

- We first import the dataset into our workspace using pandas.
- We next define the set of independent attributes. Since it is unsupervised learning, we don't have dependent attribute.
- Next, we plot the graph of elbow method to find the optimal number of clusters.
- We then train our k-means clustering model with the optimal number of clusters as input.
- We can print the results of each input as predicted by our model, that is the cluster they belong to.
- Finally, we visualize our clusters and their centroids by plotting a scatter plot.

Harshit Mishra(19BCE0799)

Code:

```
#Importing Libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

#Importing the Datasets
dataset = pd.read_csv('shopping-data.csv')
X = dataset.iloc[:, 3:].values

#Elbow method to find the optimal number of clusters
from sklearn.cluster import KMeans
wcss = []
for i in range(1, 11):
   kmeans = KMeans(n_clusters=i, init='k-means++', max_iter=300, n_init=10)
   kmeans.fit(X)
   wcss.append(kmeans.inertia_)
plt.plot(range(1, 11), wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.show()


#Applying Kmeans to the dataset
kmeans = KMeans(n_clusters=5, init='k-means++', max_iter=300, n_init=10);
y_kmeans = kmeans.fit_predict(X)


#Printing out the cluster each input belongs to
y_kmeans


# Visualising the clusters
plt.scatter(X[y_kmeans == 0, 0], X[y_kmeans == 0, 1], s = 100, c = 'red', label = 'Standard Customers')
plt.scatter(X[y_kmeans == 1, 0], X[y_kmeans == 1, 1], s = 100, c = 'blue', label = 'Careless Customers')
plt.scatter(X[y_kmeans == 2, 0], X[y_kmeans == 2, 1], s = 100, c = 'cyan', label = 'Target Customers')
plt.scatter(X[y_kmeans == 3, 0], X[y_kmeans == 3, 1], s = 100, c = 'magenta', label = 'Sensible
Customers')
plt.scatter(X[y_kmeans == 4, 0], X[y_kmeans == 4, 1], s = 100, c = 'green', label = 'Careful Customers')
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s = 300, c = 'yellow', label =
'Centroids')
plt.title ('Clusters of Clients')
plt.xlabel ('Annual Income (k$)')
plt.ylabel ('Spending Score (1-100)')
plt.legend()
plt.show()
```

Code Snippet and Explanation:

```
In [1]: #Importing Libraries
        import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
```

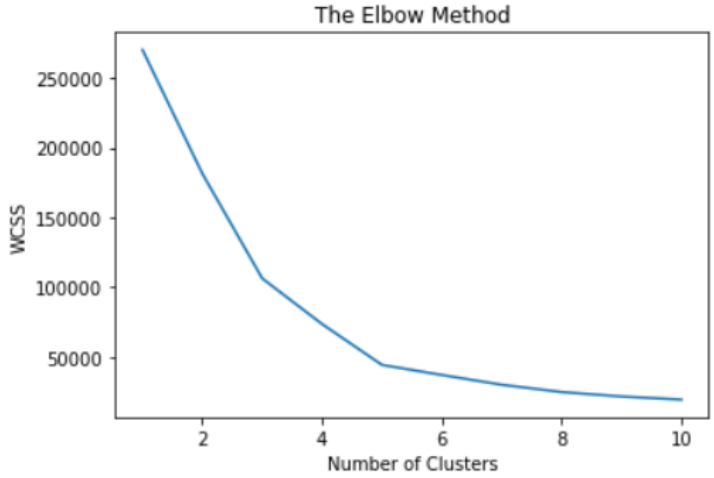
Here we are importing the necessary libraries in our workspace. The pandas library will be used to import the dataset into out workspace. numpy library is used to convert arrays and list and reshape them according to the need of input parameters to a method. matplotlib is used to visualize our result, we will be using its pyplot sub-library to do the same.

```
In [2]: #Importing the Datasets
        dataset = pd.read_csv('shopping-data.csv')
        X = dataset.iloc[:, 3:].values
```

Here we are importing the dataset into our workspace and are assigning the income attribute along with shopping score as independent variables.

In [3]:
```
#Elbow method to find the optimal number of clusters
from sklearn.cluster import KMeans
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', max_iter=300, n_init=10)
    kmeans.fit(X)
    wcss.append(kmeans.inertia_)

plt.plot(range(1, 11), wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.show()
```



Here we are plotting a graph that marks Within Cluster Sum of Squares (WCSS) with the increase in number of clusters. We can see an elbow formation when the number of clusters is 5 and hence, we assume that optimal number of clusters in our dataset is 5.

In [4]:
```
#Applying Kmeans to the dataset
kmeans = KMeans(n_clusters=5, init='k-means++', max_iter=300, n_init=10);
y_kmeans = kmeans.fit_predict(X)
```

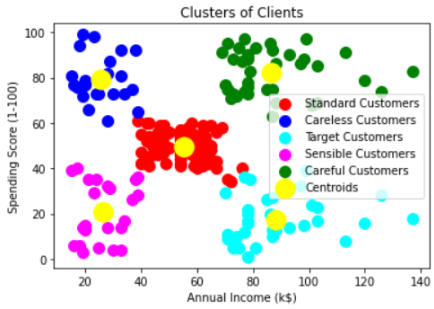
Here we are training our k-means model with 5 clusters. We are also generating the y_kmeans array that stores the cluster index of each input attribute from 0 to 4.

```
In [5]: #Printing out the cluster each input belongs to
        y_kmeans

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

Here we are printing our y_kmeans array and we can see that each input cell is assigned a value between 0 and 4, both inclusive. This corresponds to the cluster index of each input.

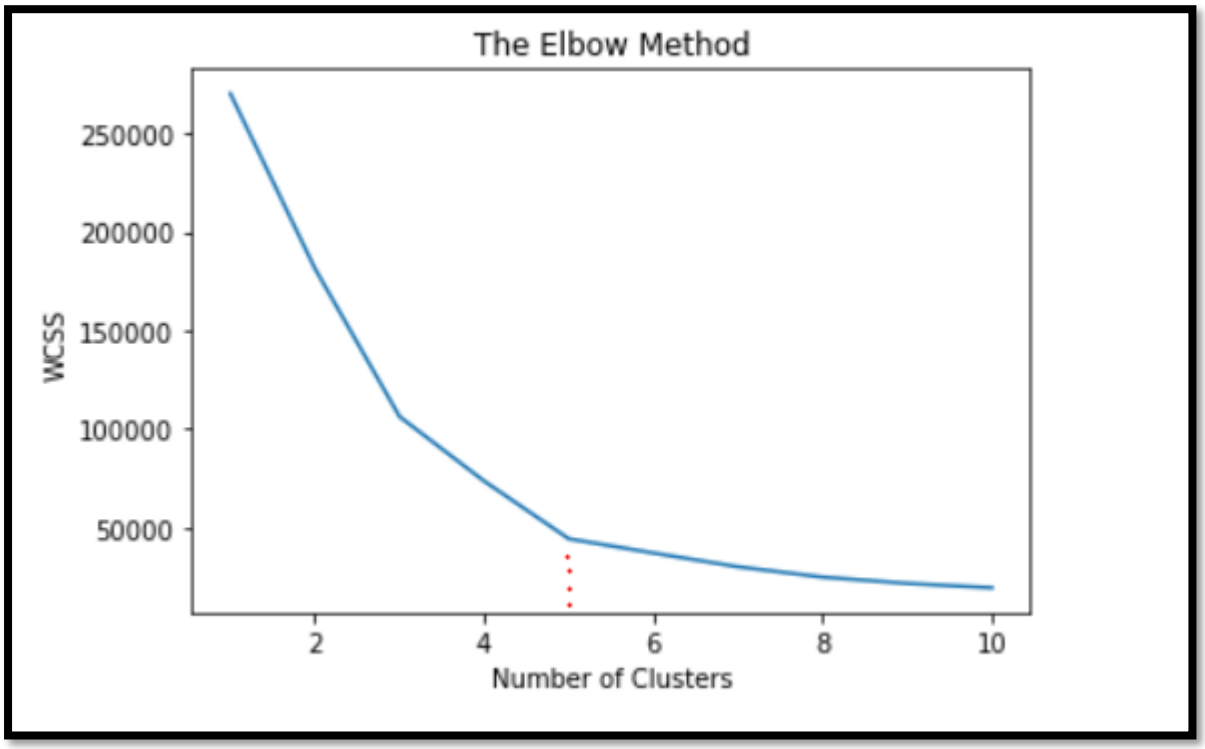
```
In [6]: # Visualising the clusters
        plt.scatter(X[y_kmeans == 0, 0], X[y_kmeans == 0, 1], s = 100, c = 'red', label = 'Standard Customers')
        plt.scatter(X[y_kmeans == 1, 0], X[y_kmeans == 1, 1], s = 100, c = 'blue', label = 'Careless Customers')
        plt.scatter(X[y_kmeans == 2, 0], X[y_kmeans == 2, 1], s = 100, c = 'cyan', label = 'Target Customers')
        plt.scatter(X[y_kmeans == 3, 0], X[y_kmeans == 3, 1], s = 100, c = 'magenta', label = 'Sensible Customers')
        plt.scatter(X[y_kmeans == 4, 0], X[y_kmeans == 4, 1], s = 100, c = 'green', label = 'Careful Customers')
        plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s = 300, c = 'yellow', label = 'Centroids')
        plt.title ('Clusters of Clients')
        plt.xlabel ('Annual Income (k$)')
        plt.ylabel ('Spending Score (1-100)')
        plt.legend()
        plt.show()
```



Here we have visualized our results. We have labelled different clusters as blue, green, pink, red and cyan. Each cluster correspond to different category of target audience. We have also marked centroids of each cluster which are yellow in colour.

Results and Conclusion:

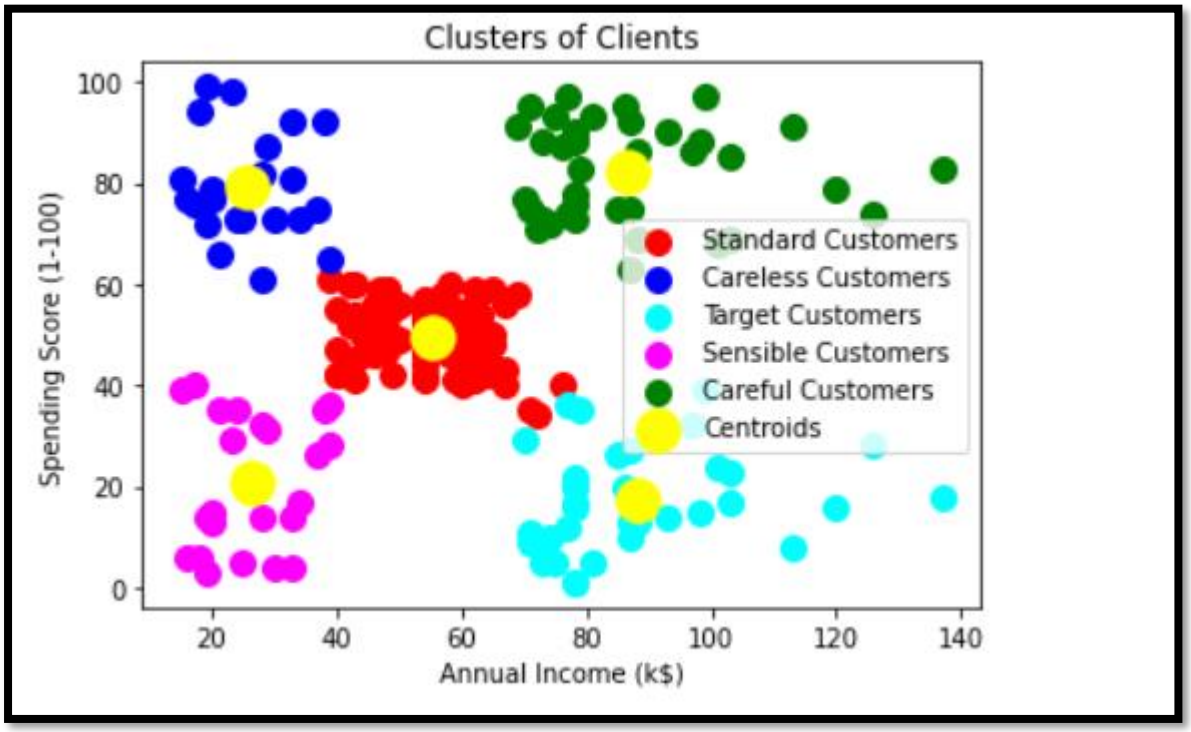
Elbow Method Graph:



We can here see that graph for Within Cluster sum of squares (WCSS) and Number of clusters take a bend when number of clusters is 5. Hence, we assume that the optimal number of clusters are 5.

Harshit Mishra(19BCE0799)

Clustering Graph:



Here different clusters are marked as blue, red, pink, cyan and green. The yellow dot over each cluster represents its centroid. We can categorise these clusters as:

- Blue Cluster corresponds to careless customers as they have low income but high spending.
- Pink Cluster as Sensible customers, becoz they have low income and low spending.
- Red Clusters are standard cluster that suggest they have median income and median spending.
- The cyan coloured cluster correspond to Target Customers, as they have high income but low spending, the shopping company can give them offers and attractions as they are capable of spending more but they aren't doing it currently.
- Finally, the Green coloured clusters are Careful customers. They have high income and thus high spending as well.