KMeans Clustering Algorithm

Problem Overview:

K-Means clustering algorithm is unsupervised learning algorithm used when we have data without a specific group or category.
The goal of K-Means algorithm is to find the best division of *n* entities in *k* groups, so that the total distance between the group's members and its corresponding centroid, representative of the group, is minimized.

The output of the K-Means clustering algorithm would be:

1. Grouping the data points with respect to their cluster centroid.
2. Clusters formed with similarity feature data.

The least K value will be taken with respect to the Elbow graph.

Steps followed:

Packages used:

Numpy: numpy is a package used for scientific computing in python. It contains linear algebra, fourier transform and random number capabilities, a powerful N-Dimensional array object and much more.

Pandas: pandas provide high-performance data structures and data analysis tool which is a BSD- licence library for python.

Matplotlib.pyplot: is a collection of functions that make it work like the MATLAB. The pyplot function makes changes to the figure.

Sklearn.cluster: the sklearn.cluster imports KMeans method which has arguments such as number of clusters, init, random state etc. to implement the Kmeans clustering algorithm

Code:

```
from matplotlib import pyplot as pt
import numpy as np
```

KMeans Clustering Using Python

```
from sklearn.cluster import KMeans
import pandas as pd
df = pd.read_csv("E:/college/Data_KMeans.csv", sep=',')

f1 = df['Distance_Feature'].values
f2 = df['Speeding_Feature'].values

X=np.matrix(list(zip(f1,f2)))
pt.scatter(f1,f2,c='Blue',s=15)
pt.title('Cluster Formation Before generating the Centroids ')
pt.xlabel('Distance Feature')
pt.ylabel('Speeding Feature')

pt.show()
kmeans = KMeans(n_clusters=2,init='k-means++',random_state=0)
kmeans.fit(X)
```

1. Importing the libraries and packages.

```
from matplotlib import pyplot as pt
import numpy as np
from sklearn.cluster import KMeans
import pandas as pd
```

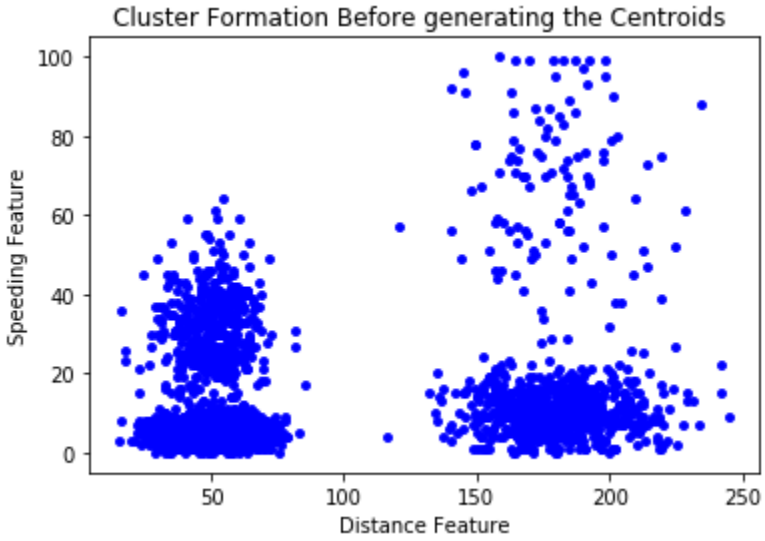
2. Importing data, getting values and plotting the data.

```
from matplotlib import pyplot as pt
import numpy as np
from sklearn.cluster import KMeans
import pandas as pd
df = pd.read_csv("E:/college/Data_KMeans.csv", sep=',')

f1 = df['Distance_Feature'].values
f2 = df['Speeding_Feature'].values

X=np.matrix(list(zip(f1,f2)))
pt.scatter(f1,f2,c='Blue',s=15)
pt.title('Cluster Formation Before generating the Centroids ')
pt.xlabel('Distance Feature')
pt.ylabel('Speeding Feature')

pt.show()
kmeans = KMeans(n_clusters=2,init='k-means++',random_state=0)
kmeans.fit(X)
```

Achyuth Pilly

KMeans Clustering Using Python

Below is the graph that shows the plotting of two clusters.



Code for cluster formation and plotting the centroids with data clusters:

```
from matplotlib import pyplot as pt
import numpy as np
from sklearn.cluster import KMeans
import pandas as pd
df = pd.read_csv("E:/college/Data_KMeans.csv", sep=',')

f1 = df['Distance_Feature'].values
f2 = df['Speeding_Feature'].values

X=np.matrix(list(zip(f1,f2)))
pt.scatter(f1,f2,c='Blue',s=15)
pt.title('Cluster Formation Before generating the Centroids ')
pt.xlabel('Distance Feature')
pt.ylabel('Speeding Feature')

pt.show()
kmeans = KMeans(n_clusters=2,init='k-means++',random_state=0)
kmeans.fit(X)
```

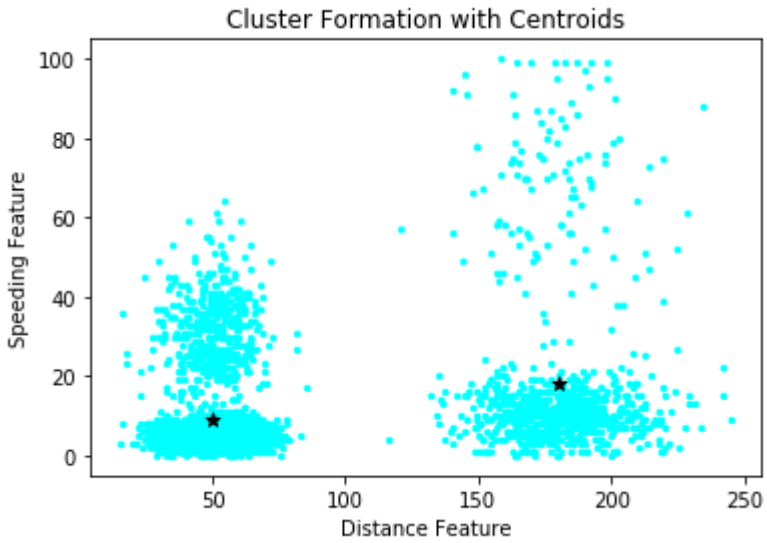
Screenshot:

Achyuth Pilly

KMeans Clustering Using Python

```
from matplotlib import pyplot as pt
import numpy as np
from sklearn.cluster import KMeans
import pandas as pd

df = pd.read_csv("E:/college/Data_KMeans.csv")

f1 = df['Distance_Feature'].values
f2 = df['Speeding_Feature'].values

kmeans = KMeans(n_clusters=2,init='k-means++',random_state=0)
Y= kmeans.fit_predict(X)
c= np.matrix(list(zip(f1, f2)), dtype=np.float32)
print(c.shape)
print(c)

# Plotting along with the Centroids
pt.scatter(f1, f2, c='cyan', s=7)
pt.scatter(kmeans.cluster_centers_[:,0], kmeans.cluster_centers_[:,1], marker='*', s=50, c='black')
pt.title('Cluster Formation with Centroids')
pt.xlabel('Distance Feature')
pt.ylabel('Speeding Feature')
pt.show()
```

Below is the graph that pointed the cluster data points.



Code:

from matplotlib import pyplot as pt
import numpy as np
from sklearn.cluster import KMea
import pandas as pd
df = pd.read_csv("E:/college/Data_KMeans.csv", sep=',')

Achyuth Pilly

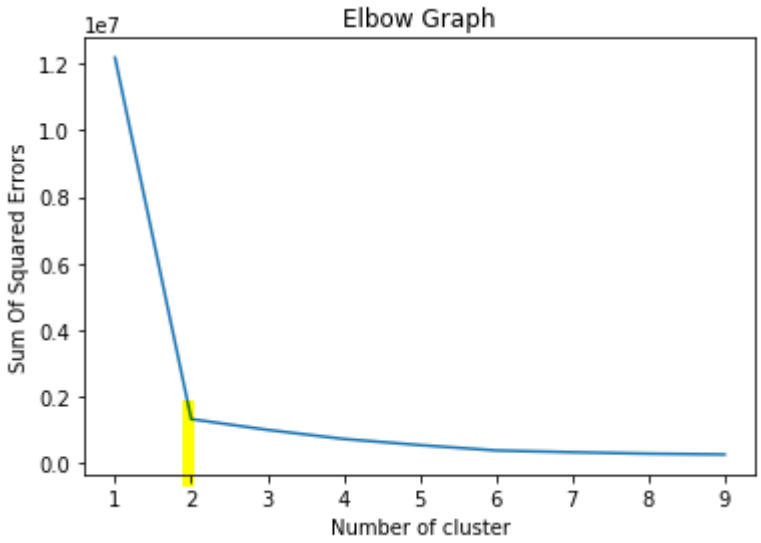
```
f1 = df['Distance_Feature'].values
f2 = df['Speeding_Feature'].values

X=np.array(list(zip(f1,f2)))

sse = {}
for k in range(1, 10):
    kmeans = KMeans(n_clusters=k, max_iter=1000).fit(X)
    df["clusters"] = kmeans.labels_
    #print(data["clusters"])
    sse[k] = kmeans.inertia_ # Inertia: Sum of distances of samples to their closest cluster center
pt.figure()
pt.plot(list(sse.keys()), list(sse.values()))
pt.title('Elbow Graph')
pt.xlabel("Number of cluster")
pt.ylabel("Sum Of Squared Errors")
pt.show()
```

3. Generating the Elbow graph by taking different K values k=1 to 10.

```
from matplotlib import pyplot as pt
import numpy as np
from sklearn.cluster import KMeans
import pandas as pd
df = pd.read_csv("E:/college/Data_KMeans.csv", sep=',')

f1 = df['Distance_Feature'].values
f2 = df['Speeding_Feature'].values

X=np.array(list(zip(f1,f2)))

sse = {}
for k in range(1, 10):
    kmeans = KMeans(n_clusters=k, max_iter=1000).fit(X)
    df["clusters"] = kmeans.labels_
    #print(data["clusters"])
    sse[k] = kmeans.inertia_ # Inertia: Sum of distances of samples to their closest cluster center
pt.figure()
pt.plot(list(sse.keys()), list(sse.values()))
pt.title('Elbow Graph')
pt.xlabel("Number of cluster")
pt.ylabel("Sum Of Squared Errors")
pt.show()
```

4. Below is the Elbow Graph generated :

KMeans Clustering Using Python



Conclusion:

By the above graph the least K value is 2. You can't find major change in the ratio if you increase the k value.

References:

https://www.datascience.com/blog/introduction-to-k-means-clustering-algorithm-learn-data-science-tutorials

https://www.chalkstreet.com/k-means-clustering-tutorial-python

http://benalexkeen.com/k-means-clustering-in-python/

http://madhugnadig.com/articles/machine-learning/2017/03/04/implementing-k-means-clustering-from-scratch-in-python.html

Achyuth Pilly