Introduction

Cluster analysis/clustering is an unsupervised machine learning algorithm used to group unlabeled datasets. the main aim is to form clusters or groups using the data points given in a dataset in such that there is high intra-cluster similarity and low inter-cluster similarity. The terms clustering goals at forming subsets/groups within a dataset comprising of data points which are similar to each other and the clusters/subsets formed can be knowingly distinguished from each other [1].

Why we use Clustering?

Let's take up we have a dataset and we don't recognize anything about it. So, a clustering algorithm can determine groups of objects where the regular distances between the data points of each cluster are nearer than to data points in other clusters.

Practical applications of Clustering in real life:

1) **Customer Segmentation**: Finding a cluster of customers with similar behavior given a large database of customers (e.g: Banking)

2) **Classifying network traffic**: Group together features of the traffic sources. Traffic categories can be easily categorized using clusters.

3) **Email Spam filter**: In grouped data looking at various sections (header, content etc) and then can use categorize which of them are spam.

4)**City-Planning**: Group no of houses according to their geo-location, house type and values.

Different Clustering Algorithms we are using:

1) **K-means Clustering** – By using this algorithm, we classify a given data set through a certain number of predetermined clusters or "k" clusters.

2) **Hierarchical Clustering** – It follows two approaches Divisive and Agglomerative.

3) **Fuzzy means Clustering** – The function of this Algorithm is almost the same as the k-means clustering, the basic difference is: in Fuzzy a data point can be put into more than one cluster.

4) **Density-Based Clustering** – It is useful in the application areas where we require non-linear cluster structures, purely based on density.

k-means Clustering:

K-Means Clustering is an Unsupervised Learning algorithm, which is used to group the unlabeled dataset to different clusters/subsets. In this algorithm 'k' states the amount of pre-defined clusters that need to be shaped in the procedure of clustering say if k=2, there will be two clusters, for k=3, there will be three clusters, and so on [2]. As it is a centroid-based algorithm, 'means' in k-means clustering is associated to the centroid of data points where each cluster is related with a centroid.

K mean mainly perform two actions:

- It Determines the most optimal value for K center points or centroids by a repetitive process.
- Assigns each data point to its closest k-center. Cluster is created with data points which are near to the particular k-center.

K means clustering steps:

- Choose a random number of centroids in the data. i.e k=3.
- Choose the same number of random points as centroids.
- Calculate the distance of each data point from the centroids.
- Allocate the data point to a cluster where its distance from the centroid is minimum.
- Recalculate the new centroids.
- Recalculate the distance from each data point to new centroids.
- Repeat the steps from point 3, till no data point change its cluster.

Advantages of k-means:

- It is easy to implement.
- Even with a large no of variables, K-Means could be computationally faster than hierarchical (if K is small).
- Convergence is guaranteed
- k-Means may yield Higher clusters than hierarchical.

Disadvantages of k-means:

- It is difficult to predict the number of clusters (K-Value).
- Initial seeds have a strong impact on the final results.
- Very sensitive to outliers

I did the K-mean algorithm Implementation in R studio using R language.

Steps I followed for K-mean clustering in R studio.

- **Step-1:** First of all, I loaded a data set as .txt upon which I am interested to perform k mean.

```
#import test data
d1 <- read.table("test2_data.txt", quote="\"", comment.char="")
d1 <- as.matrix(test1_data)
```

- **Step-2:** After loading the dataset define the no of clusters and choose the initial centroids

```
#number of cluster
cluster <- 3
c_initial <- c(0,1,2)
#first centroid = first 2 data
centroid <- test1_data[1:cluster,]
cent_DF <- data.frame(c = c_initial, centroid)
d <- matrix(0, nrow = nrow(d1), ncol = cluster)
```

- **Step-3:** Then define a stopping criteria for centroids and provide variable for this.

```
#stopping criteria for converged centroids
status <- 10
#variable for counting iteration until the cluster converged
itr <- 0
df <- data.frame(d1)
while (status != 0){
  itr <- itr + 1
  fn <- paste("Data 2,", cluster, "Cluster - itr", itr, ".png", sep = " ")}
  itr <- 0
df <- data.frame(d1)
p1 <- ggplot(df,aes(V1, V2)) +
geom_point(size=3)+ geom_point(data=cent_DF, color=c(2,3,4) , size=10)
ggsave(p1, file = "Data 2, 4 Cluster - initial.png", width = 40, height = 20, units = "cm")
```

- **Step-4:** In step 4, calculated the distance of each data to each centroids

```
#calculate distance of each data - centroids
for (j in 1:cluster){
for (i in 1:nrow(d1)){
d[i,j] = sqrt(sum((d1[i,1:ncol(d1)] - centroid[j,1:ncol(centroid)])^2))
}}
```

- **Step-5:** Now, plot the clustering results

```
df <- data.frame(c,d1)
cent_DF <- data.frame(c = c_initial, centroid)
  gg <- merge(df, cent_DF, by="c")
 plot <- ggplot(gg, aes(V1.x, V2.x, color=factor(c))) + geom_point(size=3) +
 geom_point(aes(x = V1.y, y = V2.y),size=5) +
geom_segment(aes(x = V1.y, y = V2.y, xend = V1.x, yend = V2.x))
```

- **Step-6:** In this step I calculated the new centroid based on new clustered data

```
#calculate the new centroids
compare <- cbind(d1, c)
for (i in 1:cluster){
x <- subset(compare[,1:2], compare[,3] == i-1)
for(j in 1:ncol(test1_data)){
updCentroid[i,j] <- mean(x[,j])
}}
```

- **Step-7:** Now, update the current centroid

```
#update the current centroid
if(all(updCentroid == centroid)){
status = 0}
else {
status = 1
for (i in 1:cluster){
for (j in 1:ncol(centroid)){
centroid[i,j] <- updCentroid[i,j]
}} }
```

- **Step-8:** Save the Final plot of clustering

  ggsave(plot, file = fn, width = 40, height = 20, units = "cm")

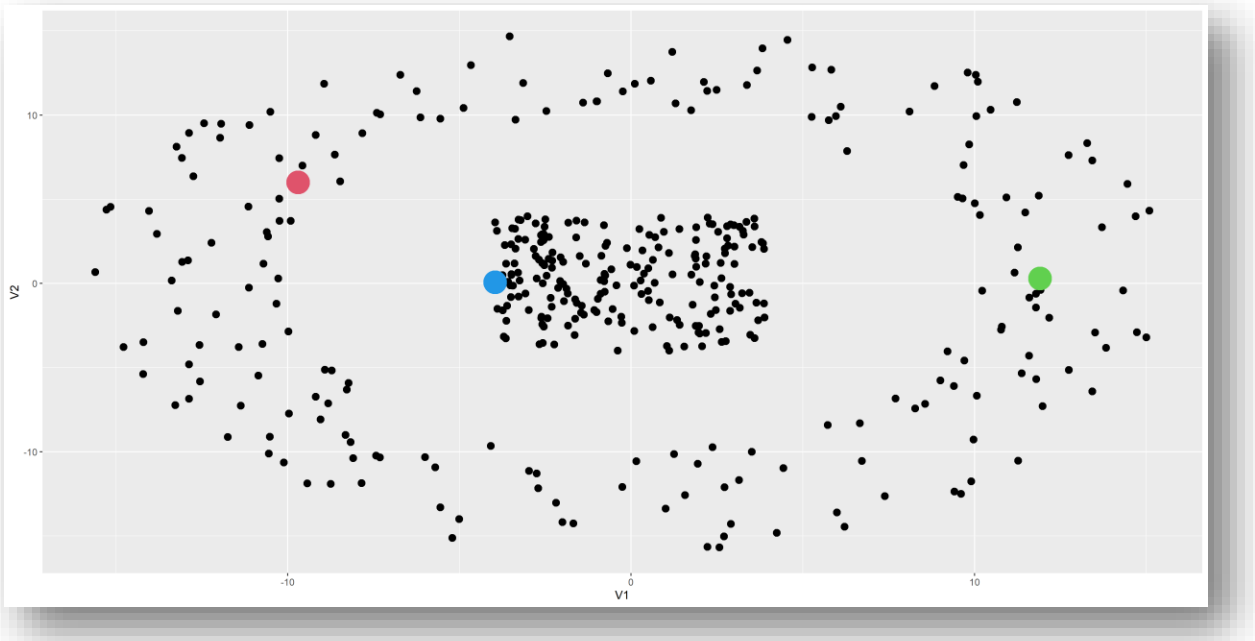

Figure 1: Initial Cluster with Centroids

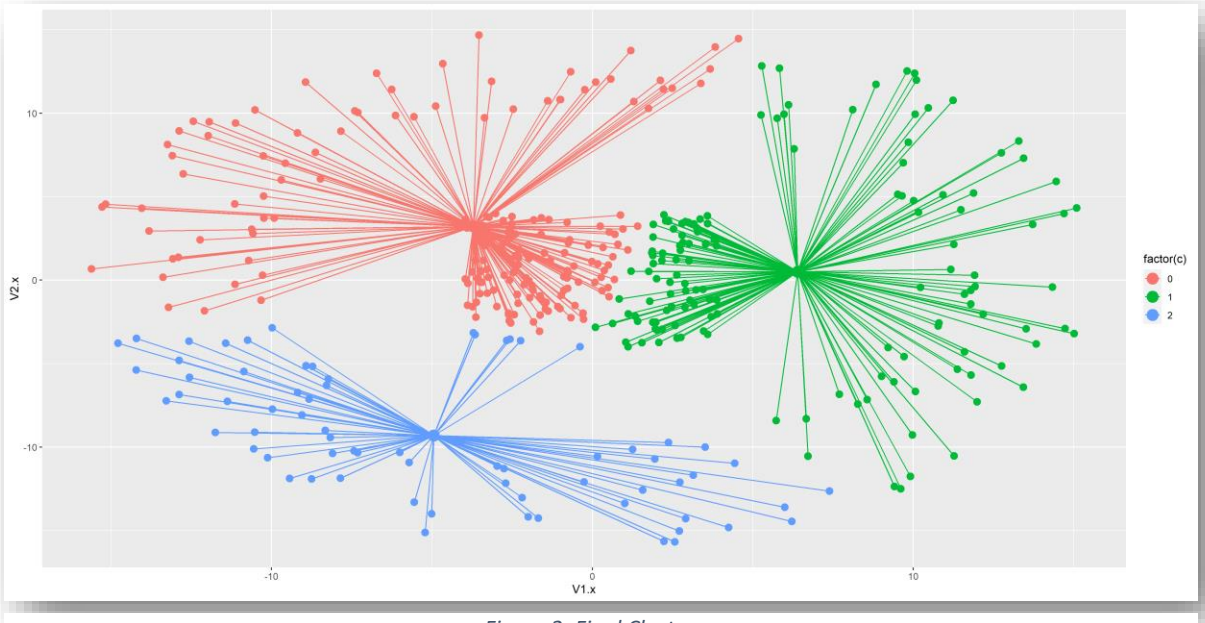

Figure 2: Final Cluster

References:

[1] https://www.analyticsvidhya.com/blog/2020/12/a-detailed-introduction-to-k-means-clustering-in-python/

[2] https://towardsdatascience.com/k-means-clustering-with-python-code-explained-5a792bd19548

[3] https://www.kdnuggets.com/2020/06/centroid-initialization-k-means-clustering.html