**Name: Jim, Jamin Rahman**

**ID: 20-42732-1**

**Introduction to Data Science**

**Section: D**

**Final Term Project**

**Project Details:**

Here, I will apply K means clustering on an unsupervised dataset. To do so, first of all I have selected and downloaded a dataset named “vehicle.csv” from kaggle. In this dataset there are 847 instances and 19 attributes where 18 of them are numerical and only one attribute (“class”) is categorical, which is the target attribute of this dataset. To consider this dataset as an unsupervised dataset I won’t consider the “class” attribute while applying k means clustering. In the first 18 attribute there are various features and measurements of vehicle is mentioned. From this information I will visualize some clusters which will contain the same type of vehicle.

Dataset Link: <https://www.kaggle.com/datasets/yogendra1127/vehicle-unsupervised-learning-project>

**Installing Package:**

*install.packages("factoextra")*

Here “Factoextra” is used for visualizing the results of multivariate data analyses. The package provides functions for generating informative plots and tables for various types of clustering and dimensionality reduction methods.

**Importing Package:**

*library(factoextra)*

**Importing the data set as csv:**

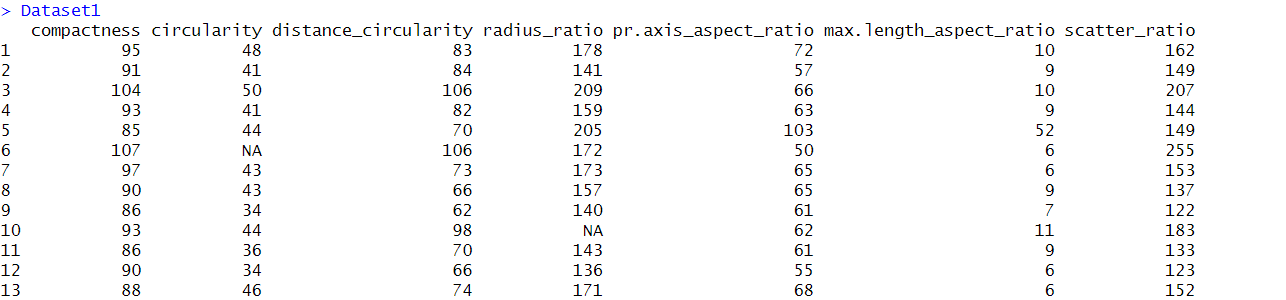
*Dataset1 <- read.csv("F://vehicle.csv", header = TRUE, sep = ",")*

Here I have import the dataset.

**Printing the data set:**

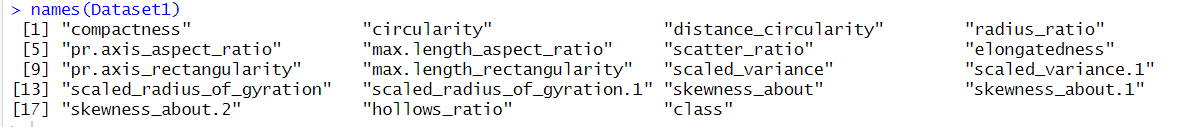
*Dataset1*

Here, I have print the dataset which was imported previously.



**Printing attributes name of dataset:**

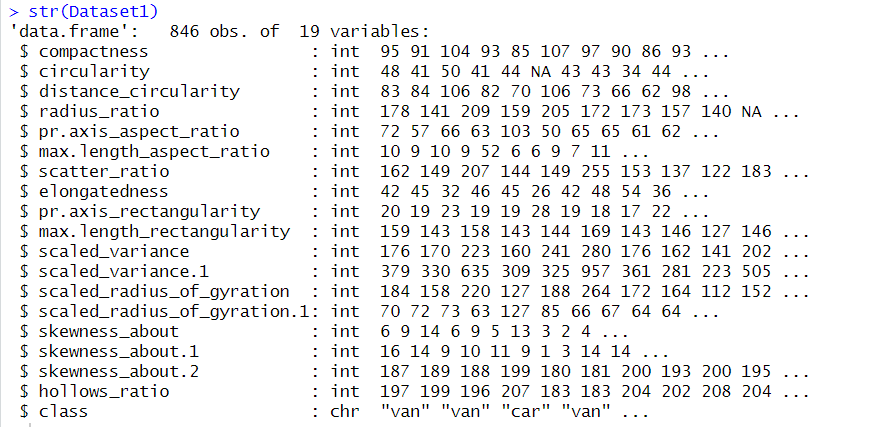
*names(Dataset1)*



Here, All the name of the attribute/Column was printed by calling names function.

**Structure of data set:**

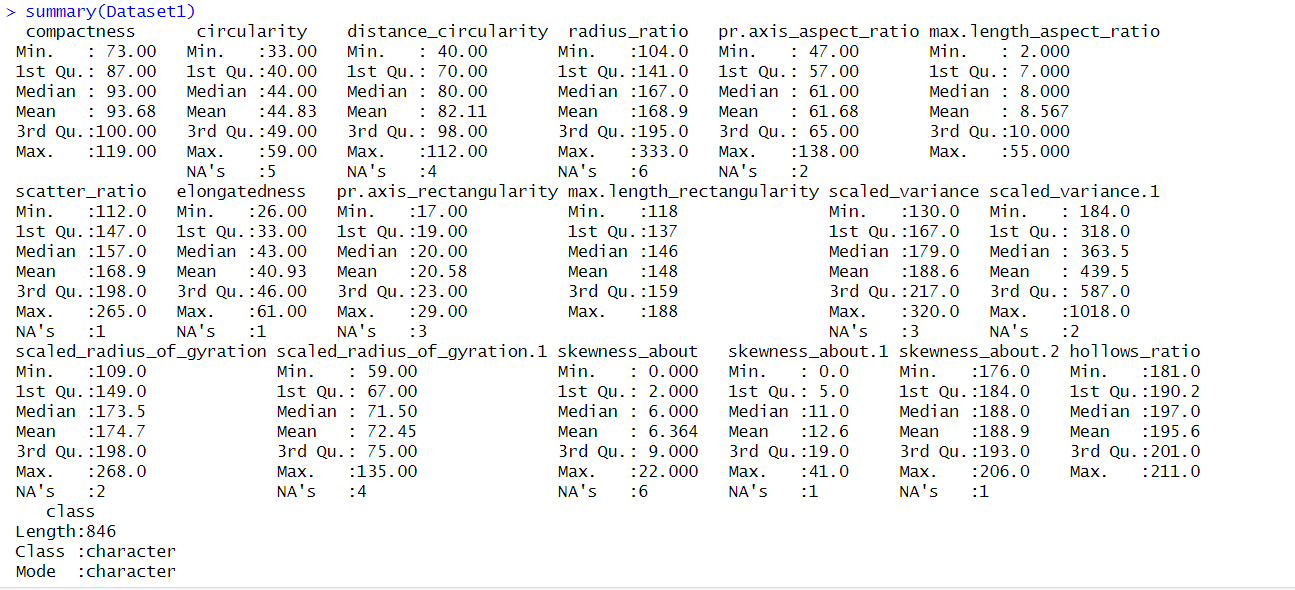
*str(Dataset1)*



Here, the structure of the dataset is visualized by calling the str function. Here we can see some values of each attribute and their data type.

**Descriptive Statistics Using summary() Function:**

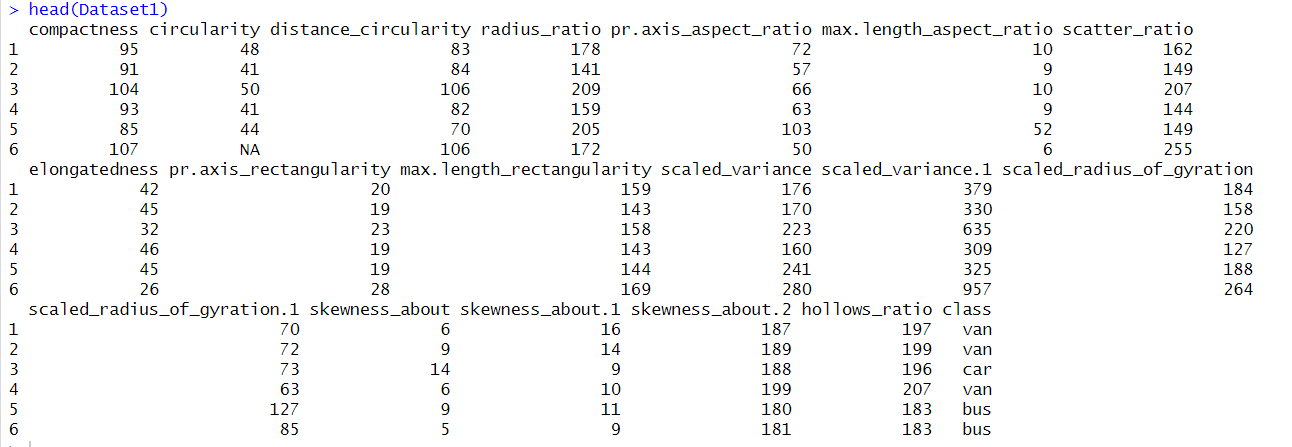
*summary(Dataset1)*



Here, by calling the summary function the descriptive statistics was visualized. From here we can identify the minimum, maximum, Mean value, Median value, value of 1st quadrant, value of 3rd quadrant of each attribute.

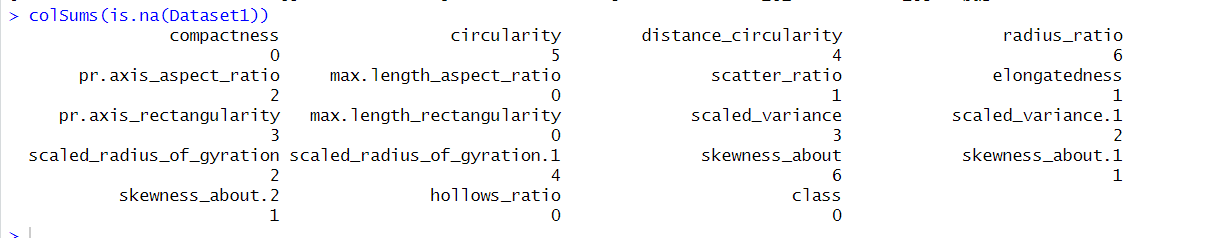
**Viewing first few values of each attribute:**

*head(Dataset1)*



**Number of null values in each column:**

*colSums(is.na(Dataset1))*



Here, to check the number of null value in each attribute, I use colSums(is.na(Dataset1)) where is.na detects if there is any missing value (NA) and colSums will return the sum of all the null values of an attribute.

**Removing null values and storing it in a new dataset:**

*Dataset1\_omit <- na.omit(Dataset1)*

By using na.omit() all the null values will be removed and the new dataset without any null value is stored as Dataset1\_omit.

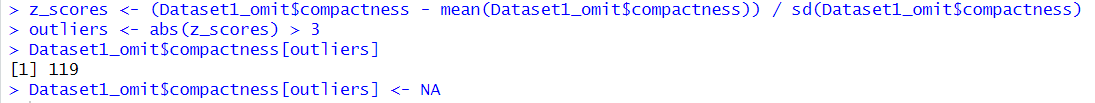
**Detecting outliers and replacing them with NA (For the attribute compactness):**

*z\_scores <- (Dataset1\_omit$compactness - mean(Dataset1\_omit$compactness)) / sd(Dataset1\_omit$compactness)*

*outliers <- abs(z\_scores) > 3*

*Dataset1\_omit$compactness[outliers]*

*Dataset1\_omit$compactness[outliers] <- NA*



Here, to detect if there is any outlier I have used the z-score which indicates how many standard deviations an observation is above or below the mean of the dataset. The z-score is calculated by subtracting the mean of the dataset from the observation, and then dividing the result by the standard deviation of the dataset:

z = (x - mean) / standard deviation

Any observation with a z-score greater than 3 or less than -3 is considered an outlier. Here I have used the abs() function which takes the absolute value of the z-scores.

There is one outlier which is 119. So I have replaced it with NA.

The same way all the attributes outliers were detected and replaced by NA.

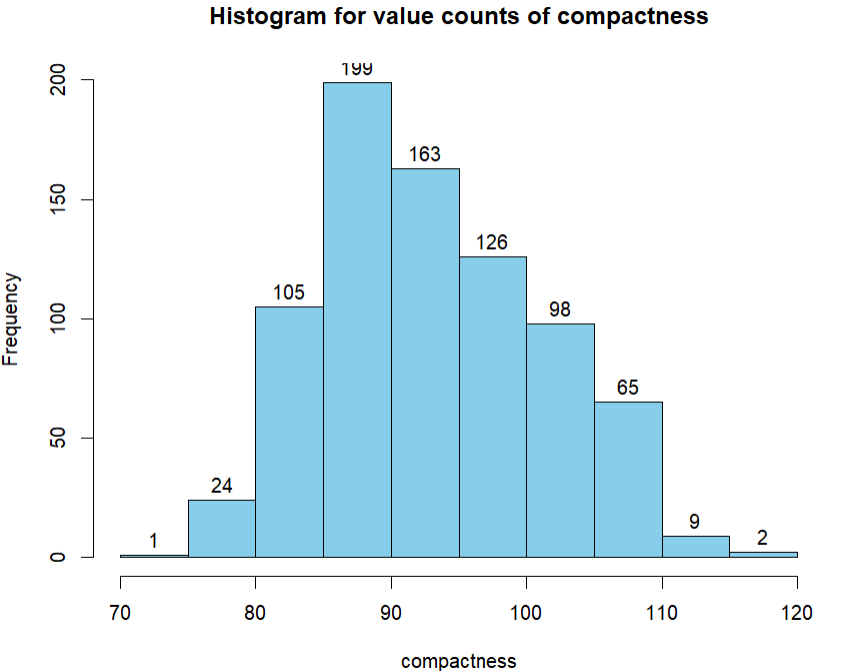
**Removing the NA values which were generated from previous step:**

*Dataset1\_omit1 <- na.omit(Dataset1\_omit)*

*Dataset1\_omit1*

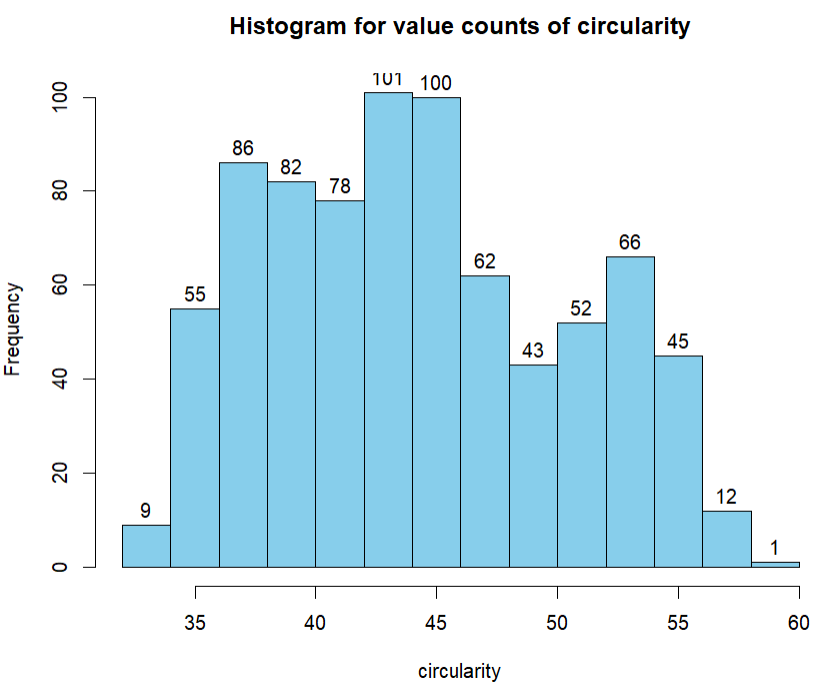
**Histogram of compactness attribute:**

*hist(Dataset1\_omit1$compactness, col="Sky Blue", main="Histogram for value counts of compactness", xlab="compactness", ylab="Frequency", labels=TRUE)*



**Histogram of circularity attribute:**

*hist(Dataset1\_omit1$circularity, col="Sky Blue", main="Histogram for value counts of circularity", xlab="circularity", ylab="Frequency", labels=TRUE)*



In the same way all the attributes histogram were also plotted.

From the histogram it was identified that all the attributes mostly contains scatter integer values. So, I have decided to replace the missing values with the median value of each attribute. However it can be also done by using mean values.

**Median Values of Each Attribute (Without class):**

*median\_of\_compactness <- median(Dataset1\_omit1$compactness)*

*median\_of\_circularity <- median(Dataset1\_omit1$circularity)*

*median\_of\_distance\_circularity <- median(Dataset1\_omit1$distance\_circularity)*

*median\_of\_radius\_ratio <- median(Dataset1\_omit1$radius\_ratio)*

*median\_of\_pr.axis\_aspect\_ratio <- median(Dataset1\_omit1$pr.axis\_aspect\_ratio)*

*median\_of\_max.length\_aspect\_ratio <- median(Dataset1\_omit1$max.length\_aspect\_ratio)*

*median\_of\_scatter\_ratio <- median(Dataset1\_omit1$scatter\_ratio)*

*median\_of\_elongatedness <- median(Dataset1\_omit1$elongatedness)*

*median\_of\_pr.axis\_rectangularity <- median(Dataset1\_omit1$pr.axis\_rectangularity)*

*median\_of\_max.length\_rectangularity <- median(Dataset1\_omit1$max.length\_rectangularity)*

*median\_of\_scaled\_variance <- median(Dataset1\_omit1$scaled\_variance)*

*median\_of\_scaled\_variance.1 <- median(Dataset1\_omit1$scaled\_variance.1)*

*median\_of\_scaled\_radius\_of\_gyration <- median(Dataset1\_omit1$scaled\_radius\_of\_gyration)*

*median\_of\_scaled\_radius\_of\_gyration.1 <- median(Dataset1\_omit1$scaled\_radius\_of\_gyration.1)*

*median\_of\_skewness\_about <- median(Dataset1\_omit1$skewness\_about)*

*median\_of\_skewness\_about.1 <- median(Dataset1\_omit1$skewness\_about.1)*

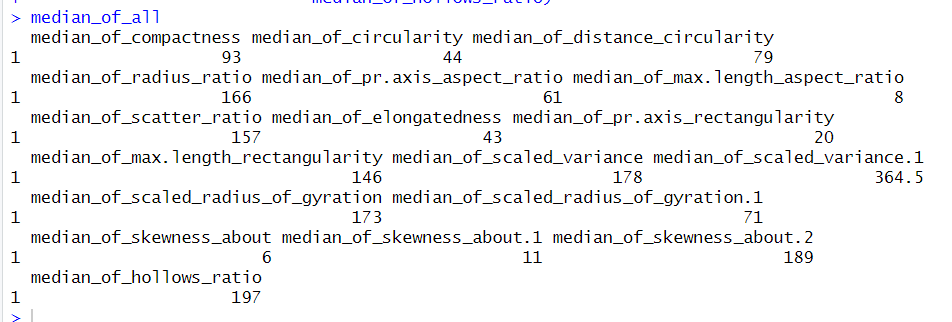
*median\_of\_skewness\_about.2 <- median(Dataset1\_omit1$skewness\_about.2)*

*median\_of\_hollows\_ratio <- median(Dataset1\_omit1$hollows\_ratio)*

*median\_of\_all <- data.frame(median\_of\_compactness, median\_of\_circularity, median\_of\_distance\_circularity, median\_of\_radius\_ratio, median\_of\_pr.axis\_aspect\_ratio, median\_of\_max.length\_aspect\_ratio, median\_of\_scatter\_ratio, median\_of\_elongatedness, median\_of\_pr.axis\_rectangularity, median\_of\_max.length\_rectangularity, median\_of\_scaled\_variance, median\_of\_scaled\_variance.1, median\_of\_scaled\_radius\_of\_gyration, median\_of\_scaled\_radius\_of\_gyration.1, median\_of\_skewness\_about, median\_of\_skewness\_about.1, median\_of\_skewness\_about.2,*

*median\_of\_hollows\_ratio)*

*median\_of\_all*



Here, all the attributes median values was first stored in a separate variable and then all of them are showed together.

**Recovering Missing values with Median values in a new dataset:**

*recover\_missing\_with\_median <- Dataset1*

*recover\_missing\_with\_median$compactness[is.na(recover\_missing\_with\_median$compactness)] <- median\_of\_compactness*

*recover\_missing\_with\_median$circularity [is.na(recover\_missing\_with\_median$circularity)] <- median\_of\_circularity*

*recover\_missing\_with\_median$distance\_circularity[is.na(recover\_missing\_with\_median$distance\_circularity)] <- median\_of\_distance\_circularity*

*recover\_missing\_with\_median$radius\_ratio[is.na(recover\_missing\_with\_median$radius\_ratio)] <- median\_of\_radius\_ratio*

*recover\_missing\_with\_median$pr.axis\_aspect\_ratio[is.na(recover\_missing\_with\_median$pr.axis\_aspect\_ratio)] <- median\_of\_pr.axis\_aspect\_ratio*

*recover\_missing\_with\_median$max.length\_aspect\_ratio[is.na(recover\_missing\_with\_median$max.length\_aspect\_ratio)] <- median\_of\_max.length\_aspect\_ratio*

*recover\_missing\_with\_median$scatter\_ratio[is.na(recover\_missing\_with\_median$scatter\_ratio)] <- median\_of\_scatter\_ratio*

*recover\_missing\_with\_median$elongatedness [is.na(recover\_missing\_with\_median$elongatedness)] <- median\_of\_elongatedness*

*recover\_missing\_with\_median$pr.axis\_rectangularity[is.na(recover\_missing\_with\_median$pr.axis\_rectangularity)] <- median\_of\_pr.axis\_rectangularity*

*recover\_missing\_with\_median$max.length\_rectangularity[is.na(recover\_missing\_with\_median$max.length\_rectangularity)] <- median\_of\_max.length\_rectangularity*

*recover\_missing\_with\_median$scaled\_variance[is.na(recover\_missing\_with\_median$scaled\_variance)] <- median\_of\_scaled\_variance*

*recover\_missing\_with\_median$scaled\_variance.1[is.na(recover\_missing\_with\_median$scaled\_variance.1)] <- median\_of\_scaled\_variance.1*

*recover\_missing\_with\_median$scaled\_radius\_of\_gyration[is.na(recover\_missing\_with\_median$scaled\_radius\_of\_gyration)] <- median\_of\_scaled\_radius\_of\_gyration*

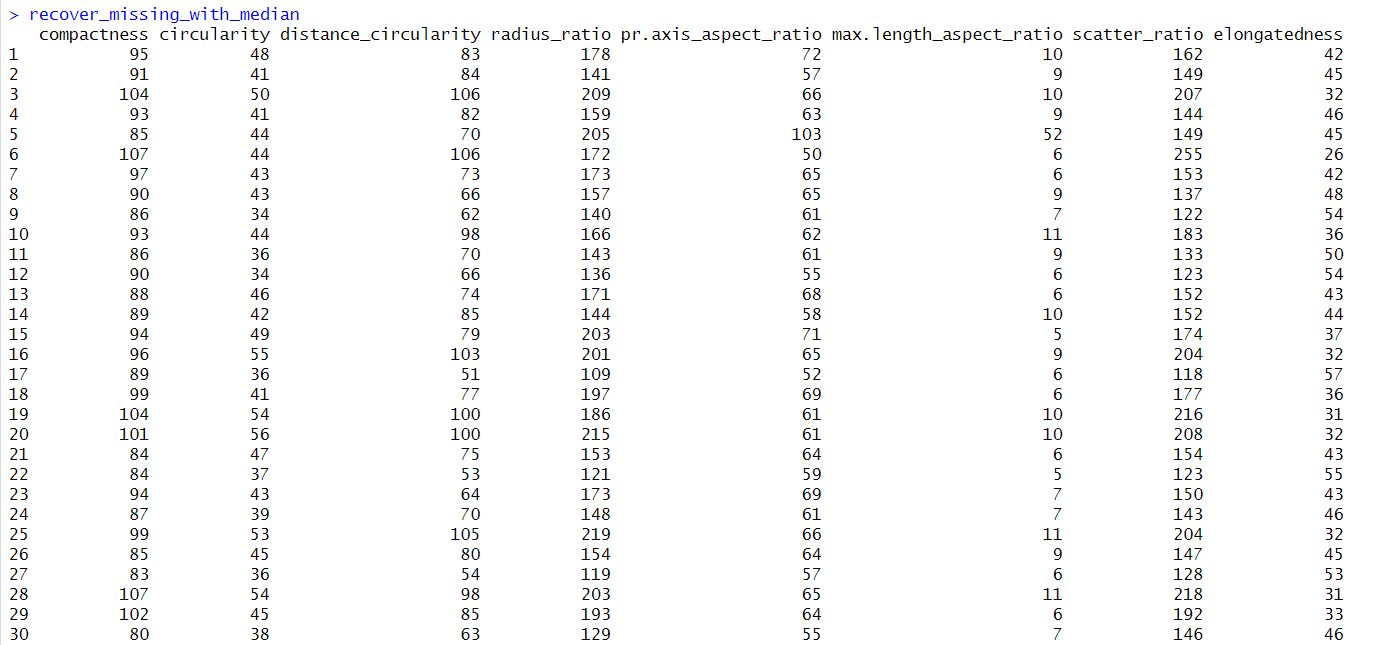
*recover\_missing\_with\_median$scaled\_radius\_of\_gyration.1[is.na(recover\_missing\_with\_median$scaled\_radius\_of\_gyration.1)] <- median\_of\_scaled\_radius\_of\_gyration.1*

*recover\_missing\_with\_median$skewness\_about[is.na(recover\_missing\_with\_median$skewness\_about)] <- median\_of\_skewness\_about*

*recover\_missing\_with\_median$skewness\_about.1 [is.na(recover\_missing\_with\_median$skewness\_about.1)] <- median\_of\_skewness\_about.1*

*recover\_missing\_with\_median$skewness\_about.2[is.na(recover\_missing\_with\_median$skewness\_about.2)] <- median\_of\_skewness\_about.2*

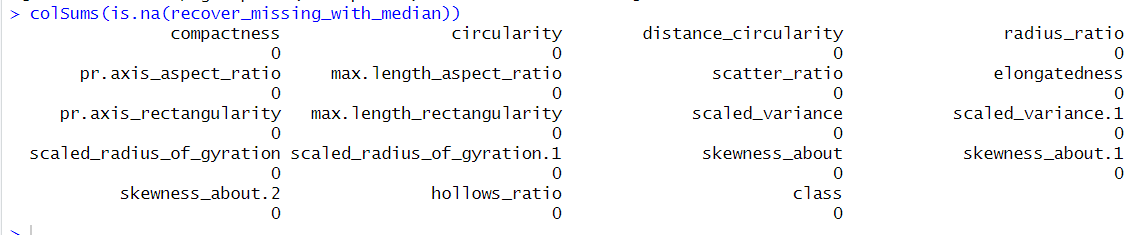
*recover\_missing\_with\_median$hollows\_ratio[is.na(recover\_missing\_with\_median$hollows\_ratio)] <- median\_of\_hollows\_ratio*



Here, for recovering the missing values I copied the dataset into a new dataset where the missing values will be replaced with median values of each attribute.For recovering the missing values of a particular attribute first it was checked if there is any missing value on that attribute. Then, if there is any missing values, it will be replaced by the median value of that attribute.Finally after replacing all the attributes missing values with their respective median value, the whole dataset was printed where it can be seen that there is no missing values in any attribute.

**Missing values in each attribute after recovering missing values with median value:**

*colSums(is.na(recover\_missing\_with\_median))*

****

Here, it can be seen that there is no missing value in any of the attribute.

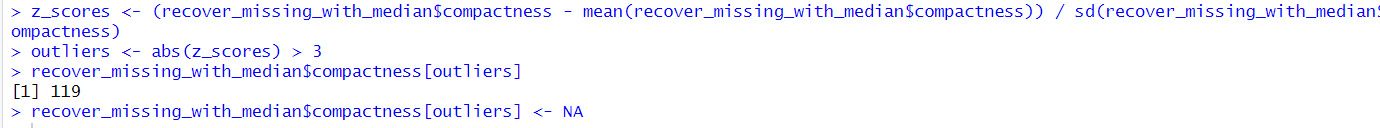
**Detecting outliers and replacing them with NA (For the attribute compactness):**

*z\_scores <- (recover\_missing\_with\_median$compactness - mean(recover\_missing\_with\_median$compactness)) / sd(recover\_missing\_with\_median$compactness)*

*outliers <- abs(z\_scores) > 3*

*recover\_missing\_with\_median$compactness[outliers]*

*recover\_missing\_with\_median$compactness[outliers] <- NA*



Here, to detect if there is any outlier I have used the z-score which I used previously so that after recovering all the missing values there is no outliers in any of the attribute.

There is one outlier which is 119. So I have replaced it with NA.

The same way all the attributes outliers were detected and replaced by NA.

**Removing the NA values which were generated from previous step:**

*Dataset1\_omit2 <- na.omit(recover\_missing\_with\_median)*

*Dataset1\_omit2*

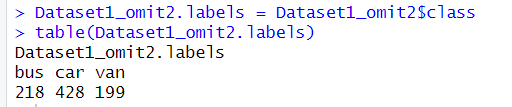
**Labeling the Class attribute:**

*Dataset1\_omit2.labels = Dataset1\_omit2$class*

Here I have assigned the values of class column in a new variable named Dataset1\_omit2.labels.

**Frequency of the values of class attribute:**

*table(Dataset1\_omit2.labels)*



Here the table shows that there are 218 bus, 428 car and 199 van in the class attribute.

**New dataset without class attribute:**

*Dataset1\_omit2\_data <- Dataset1\_omit2[1:18]*

Here as I will apply k means clustering in an unsupervised dataset, I have removed the class attribute from the dataset and stored it in a new data set as class is the target attribute of this dataset. 1:18 means that in the new dataset first 18 columns will be remain and class is 19th column in the dataset.

**Sacling:**

*Dataset1\_omit2\_data\_scale <- scale(Dataset1\_omit2\_data)*

Here I have now scaled the datasets values as Scaling involves centering the variables on their mean and scaling them to have a standard deviation of 1 and it helps to avoid issues with variables that have different scales, and allows for easier comparison between variables.

**Distance:**

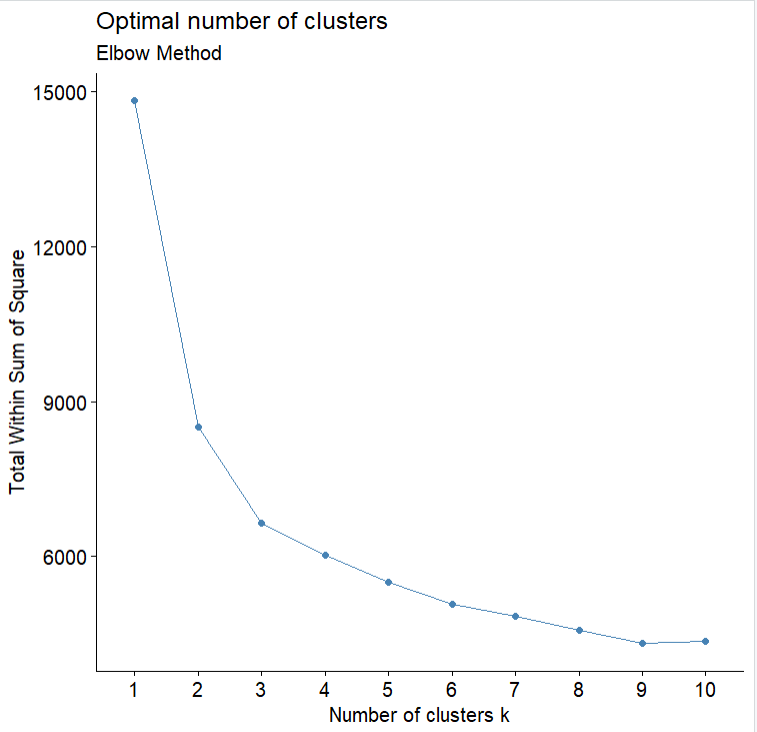
*Dataset1\_omit2\_data <- dist(Dataset1\_omit2\_data\_scale)*

Here by using the dist() function I have calculated the Euclidean distance .

**Calculating how many clusters I will need:**

*fviz\_nbclust(Dataset1\_omit2\_data\_scale, kmeans, method = "wss")+ labs(subtitle="Elbow Method")*

Here fviz\_nbclust() function from the factoextra package helps to generate an elbow plot for the k-means clustering algorithm. The method = "wss" argument specifies the elbow method to be used for determining the optimal number of clusters. The labs(subtitle="Elbow Method") function adds a subtitle to the plot to indicate that it is an elbow plot.



As the number of clusters (k) increases, the WSS tends to decrease. However, at some point, increasing the number of clusters does not lead to a significant decrease in WSS. This point is called the "elbow", and it represents the optimal number of clusters.

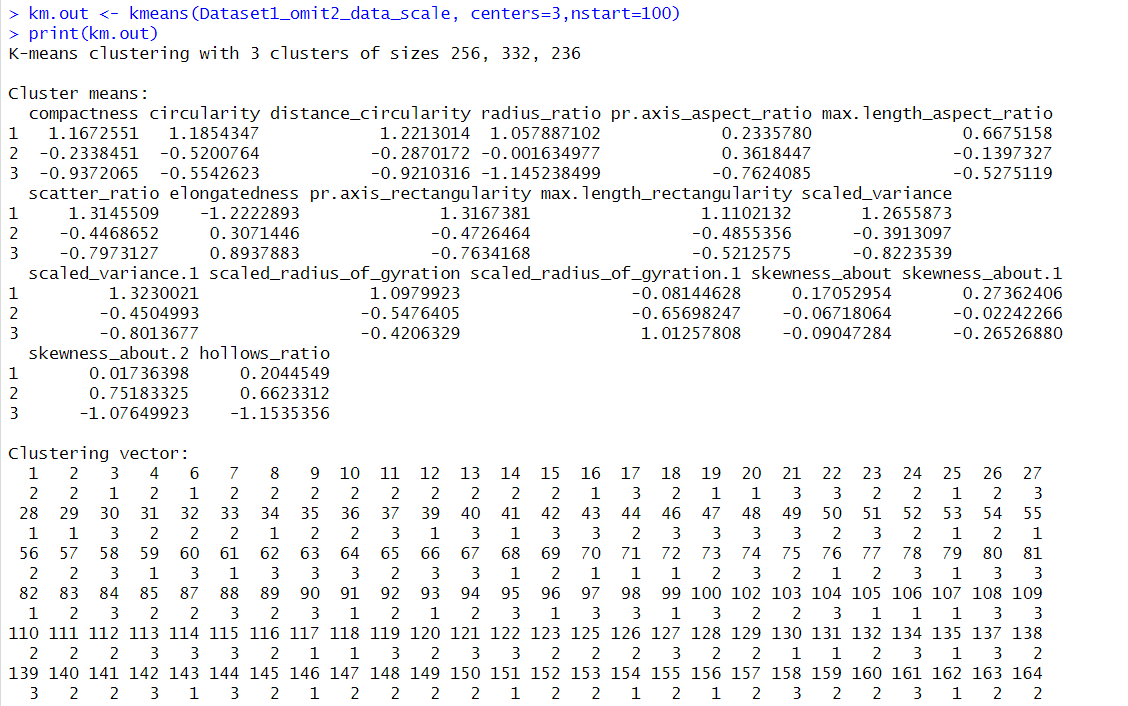
Here, to determine the optimal value of k from the elbow plot I found, I need to look for the point where the WSS begins to level off, forming an elbow shape. In the attached plot this point is k=3 where the WSS begins to level off dramatically and slowly it becomes close to zero. For optimal value I must consider the value which is not close to zero. That is why I have chosen k=3 for this dataset.

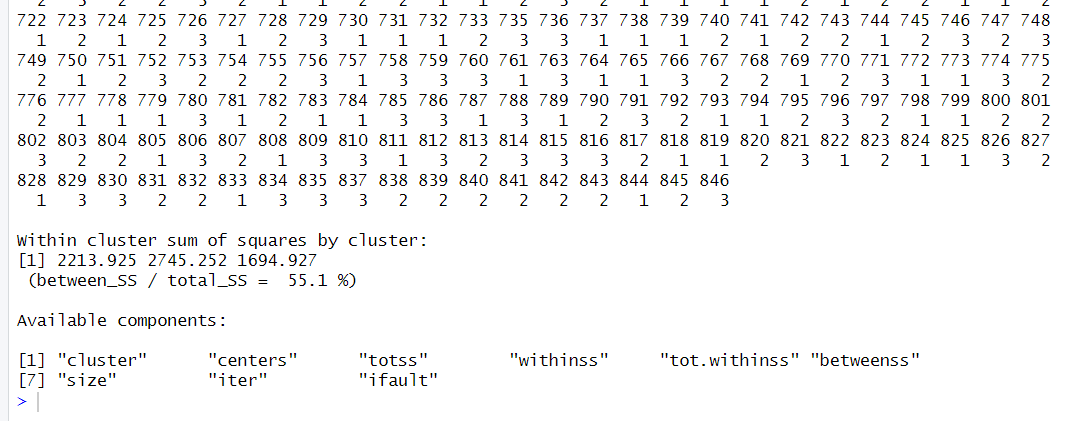
**Applying K means clustering algorithm:**

*km.out <- kmeans(Dataset1\_omit2\_data\_scale, centers=3,nstart=100)*

*print(km.out)*

Here, it performs k-means clustering with centers=3, the algorithm will group the observations into three clusters. The nstart parameter specifies the number of random initial configurations to be tried. The output of the clustering is stored in the km.out object.





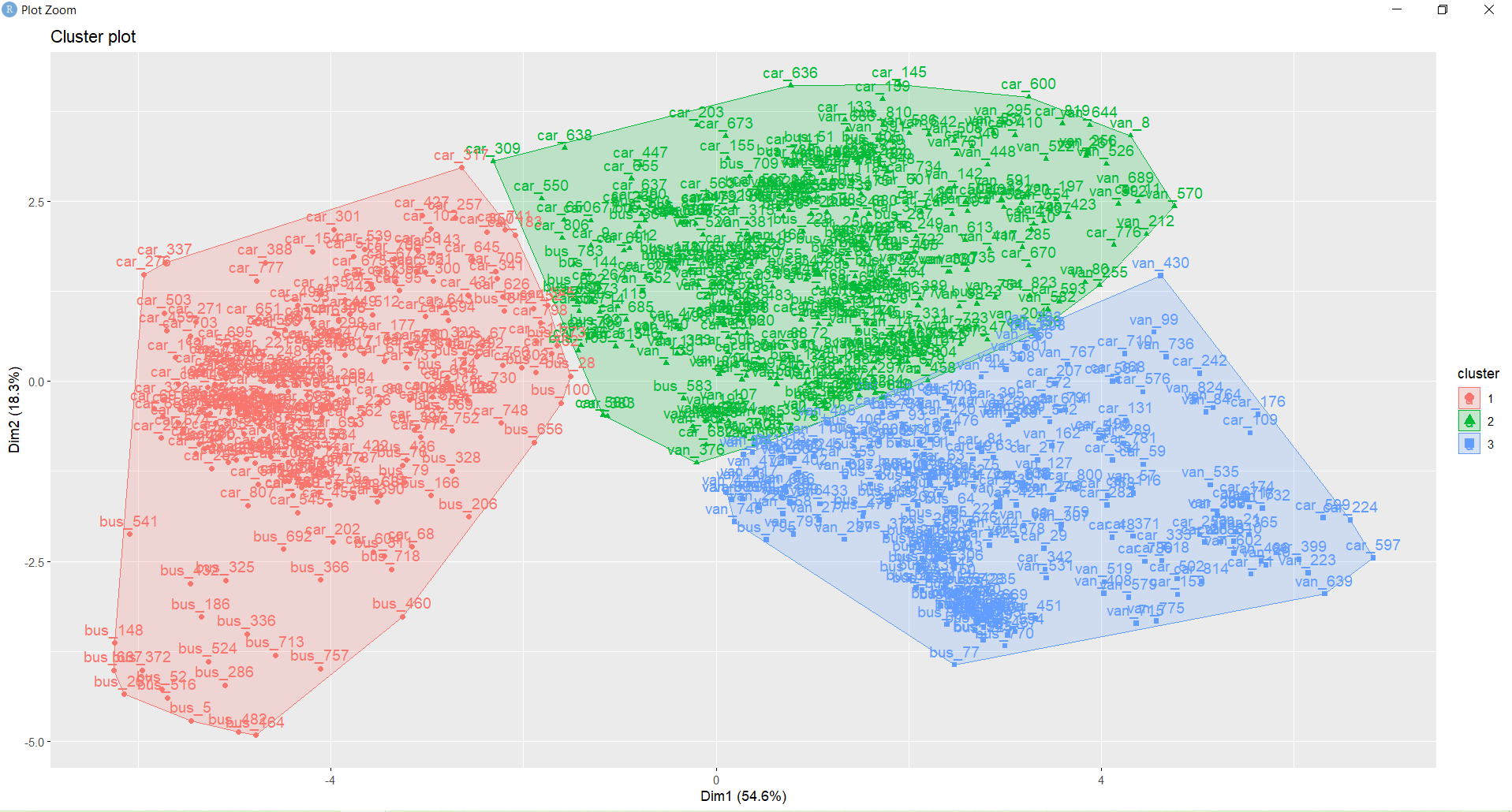
**Visualization of the clusters:**

*km.clusters<-km.out$cluster*

*rownames(Dataset1\_omit2\_data\_scale)<-paste(Dataset1\_omit2$class, 1:dim(Dataset1\_omit2)[1], sep = "\_")*

*fviz\_cluster(list(data=Dataset1\_omit2\_data\_scale, cluster = km.clusters))*

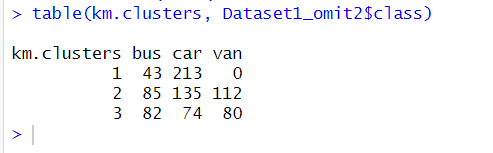
Here, the cluster plot has been generated using the factoextra package.

****

Here, it can be seen that there are three clusters and they are marked in red, blue and green.

**Contingency table:**

*T able(km.clusters, Dataset1\_omit2$class)*

****

Here, it will generate a table that will represent the number of instances of each class in the three clusters.