

**Informatics Institute of Technology**

**Department of Computing**

BSc in Computer Science

**Module: 5DATA001C**

**Machine Learning & Data Mining**

**Module Leader: Mr. Achala Aponso**

**COURSE WORK REPORT**

**Tutorial Group** : SE/CS Group B

**Student IIT ID** : 20191191

**Student UoW ID**  : W17903352

**Student First Name** : Mahfoos

**Student Surname** : Ahamed

**Table of Contents**

[**Objective 1 (Partitioning clustering)** 4](#_Toc71066294)

[**Methodologies used for reducing the input dimensionality.** 4](#_Toc71066295)

[**Pre- Processing tasks** 4](#_Toc71066296)

[**Code and plot of Outliers detection** 4](#_Toc71066297)

[**Bus Class** 4](#_Toc71066298)

[**Van Class** 5](#_Toc71066299)

[**Saab Class** 5](#_Toc71066300)

[**Opel Class** 6](#_Toc71066301)

[**Code of Removing the outliers** 6](#_Toc71066302)

[**Scaling the data** 7](#_Toc71066303)

[**Number of cluster centers** 7](#_Toc71066304)

[**Silhouette Method** 7](#_Toc71066305)

[**Code** 7](#_Toc71066306)

[**NbClust** 8](#_Toc71066307)

[**Code** 8](#_Toc71066308)

[8](#_Toc71066309)

[**K-means analysis for each attempt** 9](#_Toc71066310)

[**Evaluation of the produced outputs against 19th column** 9](#_Toc71066311)

[**Final Winner cluster case** 9](#_Toc71066312)

[**Illustrate of coordinates of each center for each clustering group** 9](#_Toc71066313)

[**Objective 2 (MLP)** 10](#_Toc71066314)

[**Methods for defining the input vector in time-series problems.** 10](#_Toc71066315)

[**Evidence of various adopted Input vectors and the related input/output metrices** 10](#_Toc71066316)

[**Evidence of correct normalization** 10](#_Toc71066317)

[**Implement Number of MLPs, using various structures (layers/nodes)** 11](#_Toc71066318)

[**Discussion of the meaning of these stat. indices** 11](#_Toc71066319)

[**Discuss the issue of “efficiency” with your two best NN structures** 11](#_Toc71066320)

[**best results (prediction output vs. desired output)** 11](#_Toc71066321)

[**Appendix** 12](#_Toc71066322)

[**Objective 1 Code** 12](#_Toc71066323)

[**Objective 2 Code** 17](#_Toc71066324)

**Table of Figure**

[Figure 1 Outlier Detection bus 4](file:///D:\Accadamic%20Materials\IIT\02nd%20Year\Second%20Semester\Machine%20Learnig\CourseWork\w1790335_ML_CWK.docx#_Toc70783157)

[Figure 2 Outlier Detection van 4](#_Toc70783158)

[Figure 3 Outlier Detection saab 5](file:///D:\Accadamic%20Materials\IIT\02nd%20Year\Second%20Semester\Machine%20Learnig\CourseWork\w1790335_ML_CWK.docx#_Toc70783159)

[Figure 4 Outlier Detection opel 5](#_Toc70783160)

[Figure 5 Silhouette method 7](file:///D:\Accadamic%20Materials\IIT\02nd%20Year\Second%20Semester\Machine%20Learnig\CourseWork\w1790335_ML_CWK.docx#_Toc70783161)

[Figure 6 NbClust 7](file:///D:\Accadamic%20Materials\IIT\02nd%20Year\Second%20Semester\Machine%20Learnig\CourseWork\w1790335_ML_CWK.docx#_Toc70783162)

# **Objective 1 (Partitioning clustering)**

## **Methodologies used for reducing the input dimensionality.**

* Here is a brief review of techniques for dimensionality reduction:
* **Missing Values Ratio**. Data columns with too many missing values are unlikely to carry much useful information. Thus, data columns with a ratio of missing values greater than a given threshold can be removed. The higher the threshold, the more aggressive the reduction.
* **Principal Component Analysis (PCA)**. Principal component analysis (PCA) is a statistical procedure that orthogonally transforms the original n numeric dimensions of a dataset into a new set of n dimensions called principal components. As a result of the transformation, the first principal component has the largest possible variance each succeeding principal component has the highest possible variance under the constraint that it is orthogonal to (i.e., uncorrelated with) the preceding principal components. Keeping only the first m < n principal components reduce the data dimensionality while retaining most of the data information, i.e., variation in the data. Notice that the PCA transformation is sensitive to the relative scaling of the original columns, and therefore, the data need to be normalized before applying PCA. Also notice that the new coordinates (PCs) are not real, system-produced variables anymore. Applying PCA to your dataset loses its interpretability. If interpretability of the results is important for your analysis, PCA is not the transformation that you should apply.

## **Pre- Processing tasks**

* In the preprocessing task first, identify and detection the outlier of each class (attribute).
* After that removed the outliers

### **Code and plot of Outliers detection**

### **Bus Class**

vehicles\_original %>%

pivot\_longer (2:19, names\_to = "labels") %>%

filter (class == "bus") %>%

mutate (class = fct\_reorder (class, value, median)) %>%

ggplot (aes (class, value, fill = reorder (labels, value))) +

geom\_boxplot() +

labs(title = "Outlier Detection for class: 'bus'")

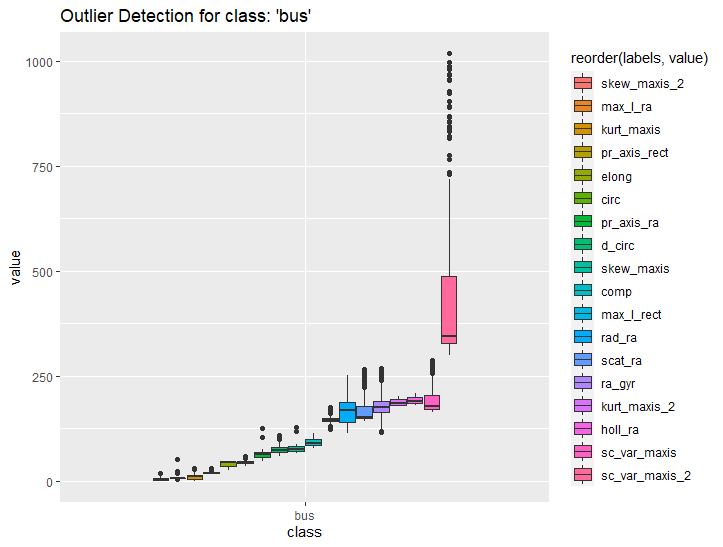


Figure 1 Outlier Detection bus

### **Van Class**

vehicles\_original %>%

pivot\_longer (2:19, names\_to = "labels") %>%

filter (class == "van") %>%

mutate (class = fct\_reorder (class, value, median)) %>%

ggplot (aes (class, value, fill = reorder (labels, value))) +

geom\_boxplot () +

labs (title = "Outlier Detection for class: 'van'")

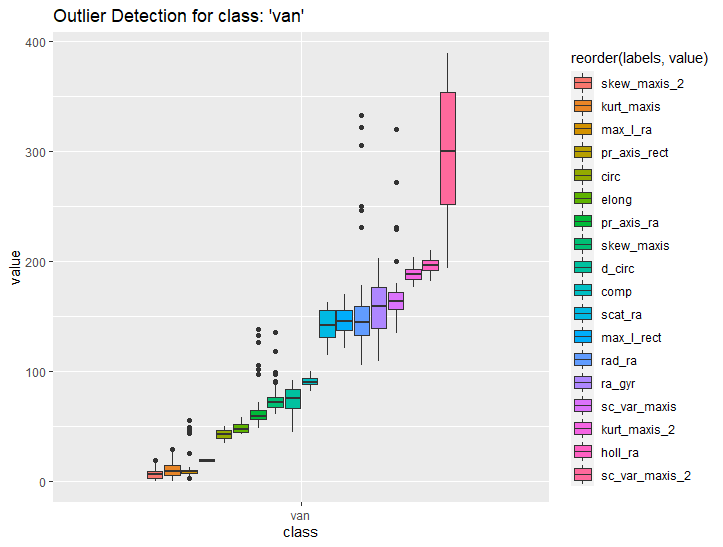


Figure 2 Outlier Detection van

### **Saab Class**

vehicles\_original %>%

pivot\_longer (2:19, names\_to = "labels") %>%

filter (class == "saab") %>%

mutate (class = fct\_reorder (class, value, median)) %>%

ggplot (aes (class, value, fill = reorder (labels, value))) +

geom\_boxplot () +

labs (title = "Outlier Detection for class: 'saab'")

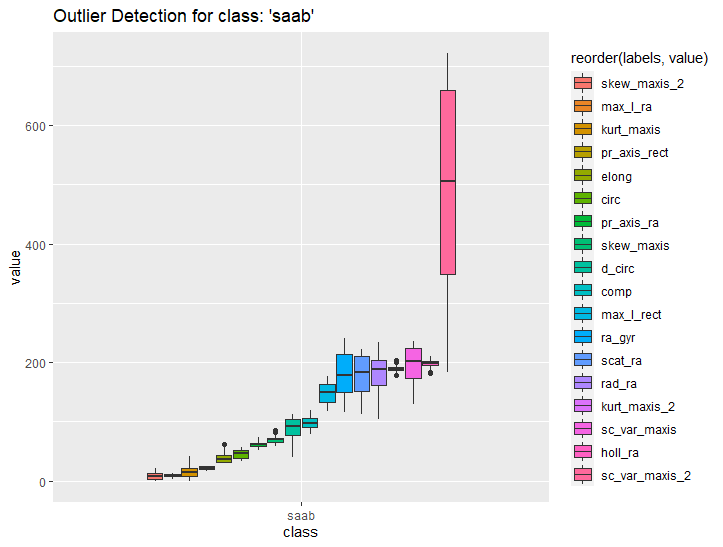


Figure 3 Outlier Detection saab

### **Opel Class**

vehicles\_original %>%

pivot\_longer (2:19, names\_to = "labels") %>%

filter (class == "opel") %>%

mutate (class = fct\_reorder (class, value, median)) %>%

ggplot (aes (class, value, fill = reorder (labels, value))) +

geom\_boxplot () +

labs (title = "Outlier Detection for class: 'opel'")

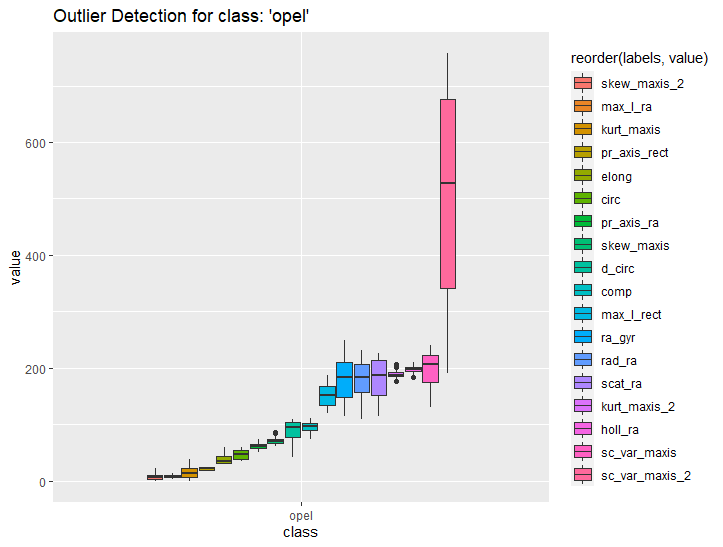


Figure 4 Outlier Detection opel

## **Code of Removing the outliers**

# Remove the Outlier

vehicles\_bus = vehicles\_original %>%

filter (class == "bus") %>%

mutate (across (2:19, ~squish (.x, quantile (.x, c(.05, .95)))))

vehicles\_van = vehicles\_original %>%

filter (class == "van") %>%

mutate (across (2:19, ~squish (.x, quantile(.x, c(.05, .95)))))

vehicles\_opel = vehicles\_original %>%

filter(class == "opel") %>%

mutate(across(2:19, ~squish(.x, quantile(.x, c(.05, .95)))))

vehicles\_saab = vehicles\_original %>%

filter(class == "saab") %>%

mutate(across(2:19, ~squish(.x, quantile(.x, c(.05, .95)))))

# Combining all class

combined = bind\_rows (list (vehicles\_bus, vehicles\_opel, vehicles\_saab, vehicles\_van)) %>%

arrange(samples)

### **Scaling the data**

* When you have variables, which are measured in different scales it is useful to scale the data.

vehicles\_scaled = vehicles\_data\_points %>%

mutate (across (everything (), scale))

## **Number of cluster centers**

* A variety of measures have been used in the partition clustering for evaluating clustering results. The term **clustering validation** is used to design the procedure of evaluating the results of a clustering algorithm. There are more than thirty indices and methods for identifying the optimal number of clusters.

### **Silhouette Method**

* This method can help determine the optimal number of clusters is called a silhouette method. Average silhouette method computes the average silhouette of observations for different values of k. The optimal number of clusters k is the one that maximize the average silhouette over a range of possible values for **k**.

### **Code**

fviz\_nbclust (vehicles\_scaled, kmeans, method = "silhouette", k.max = 24) + theme\_minimal () + ggtitle ("The Silhouette Plot")

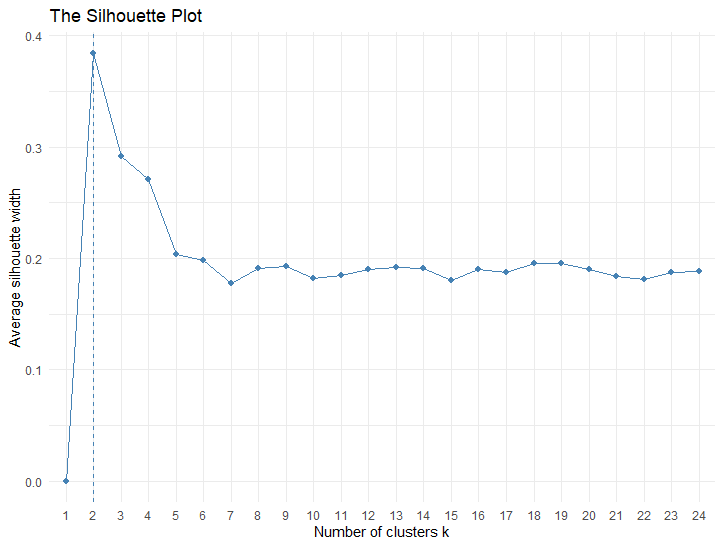


Figure 5 Silhouette method

### **NbClust**

* The **NbClust** package provides 30 indices for determining the relevant number of clusters and proposes to users the best clustering scheme from the different results obtained by varying all combinations of number of clusters, distance measures, and clustering methods.

### **Code**

# Use Euclidean for distance

cluster\_euclidean = NbClust (vehicles\_scaled, distance="euclidean", min.nc=2, max.nc=10, method="kmeans”, index="all")

# Plot the best cluster

factoextra::fviz\_nbclust(cluster\_euclidean) + theme\_minimal() +

ggtitle("NbClust's optimal number of clusters")

# 

Figure 6 NbClust

## **K-means analysis for each attempt**

## **Evaluation of the produced outputs against 19th column**

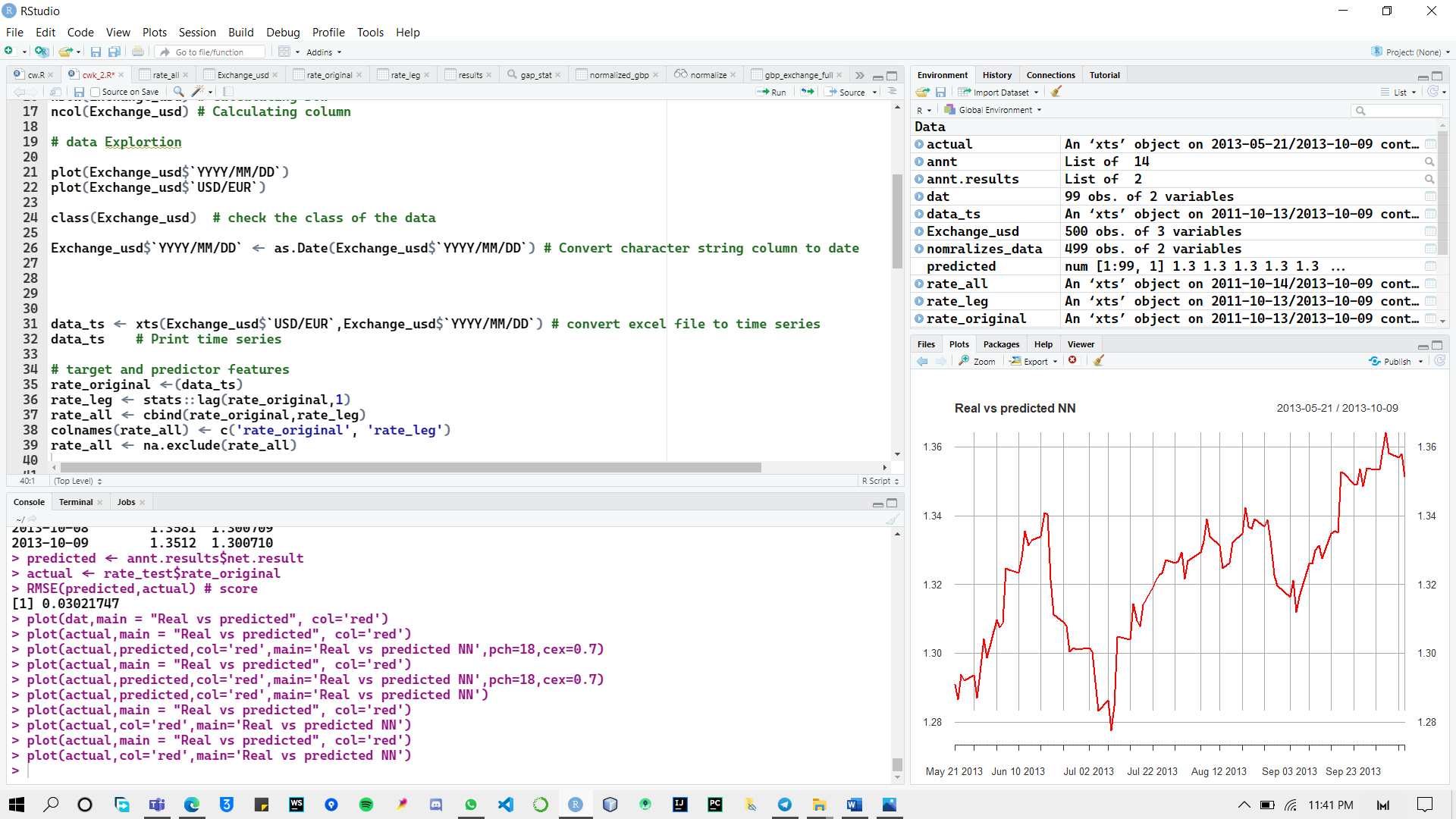
## **Final Winner cluster case**

## **Illustrate of coordinates of each center for each clustering group**

# **Objective 2 (MLP)**

## **Methods for defining the input vector in time-series problems.**

* Performed lag operation for input vectors.



## **Evidence of various adopted Input vectors and the related input/output metrices**

## **Evidence of correct normalization**

**Normalization**

* Normalization is a rescaling of the data from the original range so that all values are within the range of 0 and 1.
* Normalization requires that you know or can accurately estimate the minimum and maximum observable values.

**why normalization procedure is necessary.**

* Among the best practices for training a Neural Network is to normalize your data to obtain a mean close to 0. Normalizing the data generally speeds up learning and leads to faster convergence. Also, the (logistic) sigmoid function is hardly ever used anymore as an activation function in hidden layers of Neural Networks because the **tanh** function (among others) seems to be strictly superior.
* While this might not be immediately evident, there are very similar reasons for why this is the case. The **tanh** function is quite like the logistic sigmoid. The main difference, however, is that the tanh function outputs result between -1 and 1, while the sigmoid function outputs values that are between 0 and 1 — therefore they are always positive.

**Normalization code**

normalize <- function (x, na.rm = TRUE) {

return ((x- min(x)) /(max(x)-min(x)))

}

nomralizes\_data <- as.data.frame(lapply(rate\_all,normalize))

## **Implement Number of MLPs, using various structures (layers/nodes)**

## **Discussion of the meaning of these stat. indices**

**Discuss the issue of “efficiency” with your two best NN structures**.

## **best results (prediction output vs. desired output)**

# **Appendix**

## **Objective 1 Code**

library(tidyverse)

library(readxl) # using for read the excel file

library(factoextra)

library(NbClust) # For finding cluster

library(caret)

library(tidymodels)

# Read in the original excel datafile

vehicles\_original <- read\_excel("D:/Accadamic Materials/IIT/02nd Year/Second Semester/Machine Learnig/CourseWork/vehicles.xlsx") %>%

#plot(vehicles\_original)

janitor::clean\_names() %>%

#https://www.rdocumentation.org/packages/janitor/versions/1.2.0/topics/clean\_names

mutate(class = as\_factor(class))

# Get a birds eye view of how the dataset looks like and detect the Outlier

summary(vehicles\_original)

vehicles\_original %>%

pivot\_longer(2:19,names\_to = "labels") %>%

filter(class == "van") %>%

mutate(class = fct\_reorder(class,value,median)) %>%

ggplot(aes(class, value, fill = reorder(labels,value))) +

geom\_boxplot() +

labs(title = "Outlier Detection for class: 'van'")

vehicles\_original %>%

pivot\_longer(2:19,names\_to = "labels") %>%

filter(class == "bus") %>%

mutate(class = fct\_reorder(class,value,median)) %>%

ggplot(aes(class, value, fill = reorder(labels,value))) +

geom\_boxplot() +

labs(title = "Outlier Detection for class: 'bus'")

vehicles\_original %>%

pivot\_longer(2:19,names\_to = "labels") %>%

filter(class == "saab") %>%

mutate(class = fct\_reorder(class,value,median)) %>%

ggplot(aes(class, value, fill = reorder(labels,value))) +

geom\_boxplot() +

labs(title = "Outlier Detection for class: 'saab'")

vehicles\_original %>%

pivot\_longer(2:19,names\_to = "labels") %>%

filter(class == "opel") %>%

mutate(class = fct\_reorder(class,value,median)) %>%

ggplot(aes(class, value, fill = reorder(labels,value))) +

geom\_boxplot() +

labs(title = "Outlier Detection for class: 'opel'")

# Remove the Outlier

vehicles\_bus = vehicles\_original %>%

filter(class == "bus") %>%

mutate(across(2:19, ~squish(.x, quantile(.x, c(.05, .95)))))

vehicles\_van = vehicles\_original %>%

filter(class == "van") %>%

mutate(across(2:19, ~squish(.x, quantile(.x, c(.05, .95)))))

vehicles\_opel = vehicles\_original %>%

filter(class == "opel") %>%

mutate(across(2:19, ~squish(.x, quantile(.x, c(.05, .95)))))

vehicles\_saab = vehicles\_original %>%

filter(class == "saab") %>%

mutate(across(2:19, ~squish(.x, quantile(.x, c(.05, .95)))))

# Combined the all claas

combined = bind\_rows(list(vehicles\_bus,vehicles\_opel,vehicles\_saab,vehicles\_van)) %>%

arrange(samples)

print(combined)

# Transformed Outliers Class

combined %>%

pivot\_longer(2:19,names\_to = "labels") %>%

filter(class == "bus") %>%

mutate(class = fct\_reorder(class,value,median)) %>%

ggplot(aes(class, value, fill = reorder(labels,value))) +

geom\_boxplot() +

labs(title = "Transformed Outliers class: 'bus'")

combined %>%

pivot\_longer(2:19,names\_to = "labels") %>%

filter(class == "van") %>%

mutate(class = fct\_reorder(class,value,median)) %>%

ggplot(aes(class, value, fill = reorder(labels,value))) +

geom\_boxplot() +

labs(title = "Transformed Outliers class: 'van'")

combined %>%

pivot\_longer(2:19,names\_to = "labels") %>%

filter(class == "saab") %>%

mutate(class = fct\_reorder(class,value,median)) %>%

ggplot(aes(class, value, fill = reorder(labels,value))) +

geom\_boxplot() +

labs(title = "Transformed Outliers for class: saab")

combined %>%

pivot\_longer(2:19,names\_to = "labels") %>%

filter(class == "opel") %>%

mutate(class = fct\_reorder(class,value,median)) %>%

ggplot(aes(class, value, fill = reorder(labels,value))) +

geom\_boxplot() +

labs(title = "Transformed Outliers for class: opel")

# Remove the sample name and the class name. Both of these will be remove so that only n

#Numerical data is left for the algorithm.

vehicles\_data\_points = combined %>%

select(-samples, -class)

# Now that we have the "vehicles\_data\_points" dataset, scaling is performed

vehicles\_scaled = vehicles\_data\_points %>%

mutate(across(everything(), scale))

set.seed(123)

#Determining Optimal Number of Clusters

# Use Euclidean for distance

cluster\_euclidean = NbClust(vehicles\_scaled,distance="euclidean", min.nc=2,max.nc=10,method="kmeans",index="all")

# Plot the best cluster

factoextra::fviz\_nbclust(cluster\_euclidean) + theme\_minimal() +

ggtitle("NbClust's optimal number of clusters")

# Use manhattan for distance

cluster\_manhattan = NbClust(vehicles\_scaled,distance="manhattan", min.nc=2,max.nc=15,method="kmeans",index="all")

# Plot the best cluster

factoextra::fviz\_nbclust(cluster\_manhattan) + theme\_minimal() +

ggtitle("NbClust's optimal number of clusters")

# The Silhouette Method

fviz\_nbclust(vehicles\_scaled, kmeans, method = "silhouette", k.max = 24) + theme\_minimal() + ggtitle("The Silhouette Plot")

#compute k-means in R with the kmeans() function:

result<- kmeans(vehicles\_scaled ,centers = 2, nstart = 30) # aplly k-means algorithm with no. of k = 2

# Cluster plot

fviz\_cluster(result, data = vehicles\_scaled)

#Printing the table

table(vehicles\_original$class,result$cluster)

## **Objective 2 Code**

library(tidyverse)

library(tseries)

library(readxl)

library(neuralnet)

library(quantmod)

library(xts)

library(funtimes)

library(dplyr)

library(MLmetrics)

Exchange\_usd <- read\_excel("D:/Accadamic Materials/IIT/02nd Year/Second Semester/Machine Learnig/CourseWork/ExchangeUSD.xlsx")

# data Explortion

plot(Exchange\_usd$`YYYY/MM/DD`) # Plotting the YYYY/MM/DD Column

plot(Exchange\_usd$`USD/EUR`) # Plotting the USD/EUR Column

Exchange\_usd$`YYYY/MM/DD` <- as.Date(Exchange\_usd$`YYYY/MM/DD`) # Convert character string column to date

timeseries\_data <- xts(Exchange\_usd$`USD/EUR`,Exchange\_usd$`YYYY/MM/DD`) # convert excel file to time series

timeseries\_data # Print time series

# target and predictor features

rate\_original <-(timeseries\_data)

rate\_leg <- stats::lag(rate\_original,1)

rate\_all <- cbind(rate\_original,rate\_leg)

colnames(rate\_all) <- c('rate\_original', 'rate\_leg')

rate\_all <- na.exclude(rate\_all)

# Normalizing

normalize <- function(x, na.rm = TRUE) {

return((x- min(x)) /(max(x)-min(x)))

}

nomralizes\_data <- as.data.frame(lapply(rate\_all,normalize))

# Training and testing ranges

rate\_train <- window(rate\_all, end = '2013-05-21')

rate\_test <- window(rate\_all, start = '2013-05-21')

plot(rate\_train) # plotting the train data

set.seed(123)

# ANN Regression Fitting

nueral\_fit <- neuralnet(rate\_original~rate\_leg, data=rate\_train, hidden=1, act.fct= tanh) #ternage

nueral\_fit$result.matrix

# Graphic Neural network

plot(nueral\_fit)

# Test the accuracy of the model

temp\_test <- subset(rate\_test, select = 'rate\_leg')

head(temp\_test)

nueral\_fit.results <- compute(nueral\_fit, temp\_test)

results <- data.frame(actual = rate\_test$rate\_original, predicted= nueral\_fit.results$net.result)

results

predicted <- annt.results$net.result

actual <- rate\_test$rate\_original

RMSE(predicted,actual) # score

plot(actual,main = "Real vs predicted", col='red')