ADAPTABLITY LEVEL OF STUDENTS TO ONLINE LEARNING

Samue Ezembu & Muhammad Zarul

2023-08-20

data <- read.csv(“students\_adaptability\_level\_online\_education (1).csv”)

# Basic preprocessing

# Assuming you might need to convert factors to character for consistency

data <- data %>% mutate\_if(is.factor, as.character)

# Data visualization and EDA

# Summary statistics

summary(data)

box\_plots <- list()

# Iterate through each column and create a box plot

for (col\_name in colnames(data)) { if (col\_name != “Gender” && col\_name != “Location” && is.numeric(data[[col\_name]])) { plot <- ggplot(data, aes\_string(y = col\_name)) + geom\_boxplot(fill = “lightgreen”, color = “black”) + labs(title = paste(“Box Plot of”, col\_name), y = col\_name) + theme(axis.text.x = element\_blank(), axis.ticks.x = element\_blank()) # Remove x-axis labels and ticks

box\_plots[[col\_name]] <- plot

} }

# Create a list to store individual bar plots for non-numeric columns

bar\_plots <- list()

# Iterate through each column and create bar plots for non-numeric columns

for (col\_name in colnames(data)) { if (col\_name != “Age” && !is.numeric(data[[col\_name]])) { plot <- ggplot(data, aes\_string(x = col\_name)) + geom\_bar(fill = “purple”, color = “black”) + labs(title = paste(“Bar Plot of”, col\_name), x = col\_name, y = “Count”) + theme(axis.text.x = element\_blank(), axis.ticks.x = element\_blank()) # Remove x-axis labels and ticks

bar\_plots[[col\_name]] <- plot

} }

# Arrange the bar plots in a grid

grid\_arrange\_bar <- do.call(grid.arrange, c(bar\_plots, ncol = 2)) grid\_arrange\_box <- do.call(grid.arrange, c(box\_plots, ncol = 2))

install.packages(c("arules", "arulesViz"), repos = "https://cran.r-project.org")

## Installing packages into 'C:/Users/DELL PC/AppData/Local/R/win-library/4.3'  
## (as 'lib' is unspecified)

## package 'arules' successfully unpacked and MD5 sums checked  
## package 'arulesViz' successfully unpacked and MD5 sums checked  
##   
## The downloaded binary packages are in  
## C:\Users\DELL PC\AppData\Local\Temp\RtmpKWnpOB\downloaded\_packages

library(arules)

## Loading required package: Matrix

##   
## Attaching package: 'arules'

## The following object is masked from 'package:dplyr':  
##   
## recode

## The following objects are masked from 'package:base':  
##   
## abbreviate, write

library(arulesViz)  
  
  
dataAR <- read.csv("adapt\_processed.csv")  
  
# Set Apriori parameters  
rules <- apriori(dataAR, parameter = list(supp = 0.65, conf = 0.85, minlen = 2))

## Warning: Column(s) 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13 not logical or  
## factor. Applying default discretization (see '? discretizeDF').

## Warning in discretize(x = c(1, 1, 1, -1, 1, -1, -1, -1, 1, -1, 1, 1, -1, : The calculated breaks are: -1, -1, 1, 1  
## Only unique breaks are used reducing the number of intervals. Look at ? discretize for details.

## Apriori  
##   
## Parameter specification:  
## confidence minval smax arem aval originalSupport maxtime support minlen  
## 0.85 0.1 1 none FALSE TRUE 5 0.65 2  
## maxlen target ext  
## 10 rules TRUE  
##   
## Algorithmic control:  
## filter tree heap memopt load sort verbose  
## 0.1 TRUE TRUE FALSE TRUE 2 TRUE  
##   
## Absolute minimum support count: 980   
##   
## set item appearances ...[0 item(s)] done [0.00s].  
## set transactions ...[34 item(s), 1508 transaction(s)] done [0.00s].  
## sorting and recoding items ... [8 item(s)] done [0.00s].  
## creating transaction tree ... done [0.00s].  
## checking subsets of size 1 2 3 done [0.00s].  
## writing ... [30 rule(s)] done [0.00s].  
## creating S4 object ... done [0.00s].

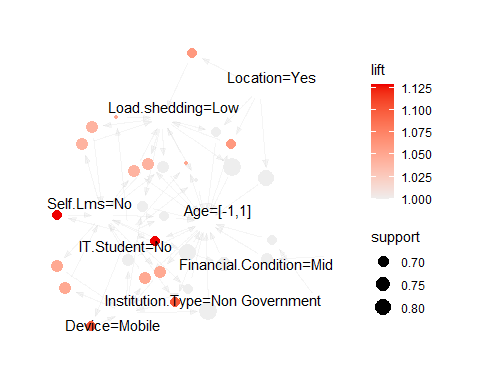
# Inspect the generated rules  
inspect(rules)

## lhs rhs support   
## [1] {Institution.Type=Non Government} => {Age=[-1,1]} 0.6883289  
## [2] {Financial.Condition=Mid} => {Age=[-1,1]} 0.7228117  
## [3] {IT.Student=No} => {Self.Lms=No} 0.6863395  
## [4] {IT.Student=No} => {Load.shedding=Low} 0.6538462  
## [5] {IT.Student=No} => {Device=Mobile} 0.6876658  
## [6] {IT.Student=No} => {Age=[-1,1]} 0.7446950  
## [7] {Location=Yes} => {Load.shedding=Low} 0.6929708  
## [8] {Location=Yes} => {Age=[-1,1]} 0.7818302  
## [9] {Self.Lms=No} => {Load.shedding=Low} 0.7108753  
## [10] {Load.shedding=Low} => {Self.Lms=No} 0.7108753  
## [11] {Self.Lms=No} => {Device=Mobile} 0.7181698  
## [12] {Device=Mobile} => {Self.Lms=No} 0.7181698  
## [13] {Self.Lms=No} => {Age=[-1,1]} 0.8163130  
## [14] {Load.shedding=Low} => {Age=[-1,1]} 0.8355438  
## [15] {Device=Mobile} => {Age=[-1,1]} 0.8381963  
## [16] {IT.Student=No, Self.Lms=No} => {Age=[-1,1]} 0.6863395  
## [17] {Age=[-1,1], IT.Student=No} => {Self.Lms=No} 0.6863395  
## [18] {IT.Student=No, Load.shedding=Low} => {Age=[-1,1]} 0.6538462  
## [19] {Age=[-1,1], IT.Student=No} => {Load.shedding=Low} 0.6538462  
## [20] {IT.Student=No, Device=Mobile} => {Age=[-1,1]} 0.6876658  
## [21] {Age=[-1,1], IT.Student=No} => {Device=Mobile} 0.6876658  
## [22] {Location=Yes, Load.shedding=Low} => {Age=[-1,1]} 0.6929708  
## [23] {Age=[-1,1], Location=Yes} => {Load.shedding=Low} 0.6929708  
## [24] {Load.shedding=Low, Self.Lms=No} => {Age=[-1,1]} 0.7108753  
## [25] {Age=[-1,1], Self.Lms=No} => {Load.shedding=Low} 0.7108753  
## [26] {Age=[-1,1], Load.shedding=Low} => {Self.Lms=No} 0.7108753  
## [27] {Self.Lms=No, Device=Mobile} => {Age=[-1,1]} 0.7181698  
## [28] {Age=[-1,1], Self.Lms=No} => {Device=Mobile} 0.7181698  
## [29] {Age=[-1,1], Device=Mobile} => {Self.Lms=No} 0.7181698  
## [30] {Load.shedding=Low, Device=Mobile} => {Age=[-1,1]} 0.7035809  
## confidence coverage lift count  
## [1] 1.0000000 0.6883289 1.000000 1038   
## [2] 1.0000000 0.7228117 1.000000 1090   
## [3] 0.9216385 0.7446950 1.129026 1035   
## [4] 0.8780053 0.7446950 1.050819 986   
## [5] 0.9234194 0.7446950 1.101674 1037   
## [6] 1.0000000 0.7446950 1.000000 1123   
## [7] 0.8863444 0.7818302 1.060799 1045   
## [8] 1.0000000 0.7818302 1.000000 1179   
## [9] 0.8708367 0.8163130 1.042240 1072   
## [10] 0.8507937 0.8355438 1.042240 1072   
## [11] 0.8797725 0.8163130 1.049602 1083   
## [12] 0.8568038 0.8381963 1.049602 1083   
## [13] 1.0000000 0.8163130 1.000000 1231   
## [14] 1.0000000 0.8355438 1.000000 1260   
## [15] 1.0000000 0.8381963 1.000000 1264   
## [16] 1.0000000 0.6863395 1.000000 1035   
## [17] 0.9216385 0.7446950 1.129026 1035   
## [18] 1.0000000 0.6538462 1.000000 986   
## [19] 0.8780053 0.7446950 1.050819 986   
## [20] 1.0000000 0.6876658 1.000000 1037   
## [21] 0.9234194 0.7446950 1.101674 1037   
## [22] 1.0000000 0.6929708 1.000000 1045   
## [23] 0.8863444 0.7818302 1.060799 1045   
## [24] 1.0000000 0.7108753 1.000000 1072   
## [25] 0.8708367 0.8163130 1.042240 1072   
## [26] 0.8507937 0.8355438 1.042240 1072   
## [27] 1.0000000 0.7181698 1.000000 1083   
## [28] 0.8797725 0.8163130 1.049602 1083   
## [29] 0.8568038 0.8381963 1.049602 1083   
## [30] 1.0000000 0.7035809 1.000000 1061

# Plot the rules as a graph  
plot(rules, method = "graph", control = list(type = "items"))

## Warning: Unknown control parameters: type

## Available control parameters (with default values):  
## layout = stress  
## circular = FALSE  
## ggraphdots = NULL  
## edges = <environment>  
## nodes = <environment>  
## nodetext = <environment>  
## colors = c("#EE0000FF", "#EEEEEEFF")  
## engine = ggplot2  
## max = 100  
## verbose = FALSE



# Install and load the necessary packages  
install.packages(c("klaR", "caret"), repos = "https://cran.r-project.org")

## Installing packages into 'C:/Users/DELL PC/AppData/Local/R/win-library/4.3'  
## (as 'lib' is unspecified)

## package 'klaR' successfully unpacked and MD5 sums checked  
## package 'caret' successfully unpacked and MD5 sums checked  
##   
## The downloaded binary packages are in  
## C:\Users\DELL PC\AppData\Local\Temp\RtmpKWnpOB\downloaded\_packages

library(klaR)

## Loading required package: MASS

##   
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':  
##   
## select

library(caret)

## Loading required package: lattice

# Read the dataset "adapt\_processed.csv" into 'dataNB'  
dataNB <- read.csv("adapt\_processed.csv")  
  
# Split the data into training and test sets  
set.seed(123) # For reproducibility  
inx <- sample(nrow(dataNB), round(nrow(dataNB) \* 0.7))  
train <- dataNB[inx, ]  
test <- dataNB[-inx, ]  
  
# Separate features (x) and target variable (y) for training set  
x\_train <- train[, -13] # Excluding column 13  
y\_train <- train$Adaptivity.Level  
  
# Separate features (x) and target variable (y) for test set  
x\_test <- test[, -13]  
y\_test <- test$Adaptivity.Level  
  
# Train a Naive Bayes model using cross-validation  
nb\_model <- train(x\_train, y\_train, method = 'nb',  
 trControl = trainControl(method = 'cv', number = 5))

# Create a confusion matrix to evaluate the performance of the model  
tbl <- table(Actual = y\_test, Predicted = preds)  
print(tbl)

## Predicted  
## Actual High Low Moderate  
## High 19 25 8  
## Low 1 144 24  
## Moderate 6 186 39

# Calculate and print the accuracy of the Naive Bayes model  
accuracy <- sum(diag(tbl)) / sum(tbl)  
cat("Accuracy:", accuracy, "\n")

## Accuracy: 0.4469027

# Create a confusion matrix and calculate additional metrics using caret's confusionMatrix function  
conf\_matrix <- confusionMatrix(table(preds, y\_test))  
print(conf\_matrix)

## Confusion Matrix and Statistics  
##   
## y\_test  
## preds High Low Moderate  
## High 19 1 6  
## Low 25 144 186  
## Moderate 8 24 39  
##   
## Overall Statistics  
##   
## Accuracy : 0.4469   
## 95% CI : (0.4004, 0.4941)  
## No Information Rate : 0.5111   
## P-Value [Acc > NIR] : 0.9973   
##   
## Kappa : 0.1071   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Statistics by Class:  
##   
## Class: High Class: Low Class: Moderate  
## Sensitivity 0.36538 0.8521 0.16883  
## Specificity 0.98250 0.2544 0.85520  
## Pos Pred Value 0.73077 0.4056 0.54930  
## Neg Pred Value 0.92254 0.7423 0.49606  
## Prevalence 0.11504 0.3739 0.51106  
## Detection Rate 0.04204 0.3186 0.08628  
## Detection Prevalence 0.05752 0.7854 0.15708  
## Balanced Accuracy 0.67394 0.5532 0.51202

# Install and load the necessary packages  
install.packages(c("randomForest", "caret"), repos = "https://cran.r-project.org")

## Warning: package 'caret' is in use and will not be installed

## Installing package into 'C:/Users/DELL PC/AppData/Local/R/win-library/4.3'  
## (as 'lib' is unspecified)

## package 'randomForest' successfully unpacked and MD5 sums checked  
##   
## The downloaded binary packages are in  
## C:\Users\DELL PC\AppData\Local\Temp\RtmpKWnpOB\downloaded\_packages

library(caret)  
library(randomForest)

## randomForest 4.7-1.1

## Type rfNews() to see new features/changes/bug fixes.

##   
## Attaching package: 'randomForest'

## The following object is masked from 'package:gridExtra':  
##   
## combine

## The following object is masked from 'package:ggplot2':  
##   
## margin

## The following object is masked from 'package:dplyr':  
##   
## combine

# Read the CSV file "adapt\_processed.csv" into the data  
dataRF <- read.csv("adapt\_processed.csv")  
  
# Assuming Adaptivity.Level is categorical  
# Encode the categorical variable using ordinal encoding  
dataRF$Adaptivity.Level <- as.integer(factor(dataRF$Adaptivity.Level))  
  
# Fit a Random Forest ensemble model with 300 trees using the encoded Adaptivity.Level as the response variable  
RFensemble300 <- randomForest(factor(Adaptivity.Level) ~ ., data = dataRF, ntree = 300)  
  
# Display the summary of the RFensemble300 model  
summary(RFensemble300)

## Length Class Mode   
## call 4 -none- call   
## type 1 -none- character  
## predicted 1508 factor numeric   
## err.rate 1200 -none- numeric   
## confusion 12 -none- numeric   
## votes 4524 matrix numeric   
## oob.times 1508 -none- numeric   
## classes 3 -none- character  
## importance 12 -none- numeric   
## importanceSD 0 -none- NULL   
## localImportance 0 -none- NULL   
## proximity 0 -none- NULL   
## ntree 1 -none- numeric   
## mtry 1 -none- numeric   
## forest 14 -none- list   
## y 1508 factor numeric   
## test 0 -none- NULL   
## inbag 0 -none- NULL   
## terms 3 terms call

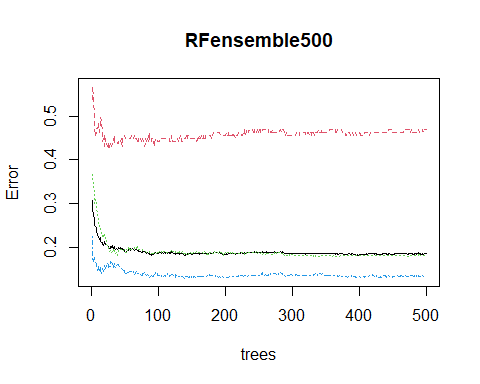
# Fit a Random Forest ensemble model with 300 trees and mtry=1 (consider only 1 feature at each split)  
RFensemble300.mtry1 <- randomForest(factor(Adaptivity.Level) ~ ., data = dataRF, ntree = 300, mtry = 1)  
  
# Display the summary of the RFensemble300.mtry1 model  
summary(RFensemble300.mtry1)

## Length Class Mode   
## call 5 -none- call   
## type 1 -none- character  
## predicted 1508 factor numeric   
## err.rate 1200 -none- numeric   
## confusion 12 -none- numeric   
## votes 4524 matrix numeric   
## oob.times 1508 -none- numeric   
## classes 3 -none- character  
## importance 12 -none- numeric   
## importanceSD 0 -none- NULL   
## localImportance 0 -none- NULL   
## proximity 0 -none- NULL   
## ntree 1 -none- numeric   
## mtry 1 -none- numeric   
## forest 14 -none- list   
## y 1508 factor numeric   
## test 0 -none- NULL   
## inbag 0 -none- NULL   
## terms 3 terms call

# Fit a Random Forest ensemble model with 500 trees using the encoded Adaptivity.Level as the response variable  
RFensemble500 <- randomForest(factor(Adaptivity.Level) ~ ., data = dataRF, ntree = 500)  
  
# Display the summary of the RFensemble500 model  
summary(RFensemble500)

## Length Class Mode   
## call 4 -none- call   
## type 1 -none- character  
## predicted 1508 factor numeric   
## err.rate 2000 -none- numeric   
## confusion 12 -none- numeric   
## votes 4524 matrix numeric   
## oob.times 1508 -none- numeric   
## classes 3 -none- character  
## importance 12 -none- numeric   
## importanceSD 0 -none- NULL   
## localImportance 0 -none- NULL   
## proximity 0 -none- NULL   
## ntree 1 -none- numeric   
## mtry 1 -none- numeric   
## forest 14 -none- list   
## y 1508 factor numeric   
## test 0 -none- NULL   
## inbag 0 -none- NULL   
## terms 3 terms call

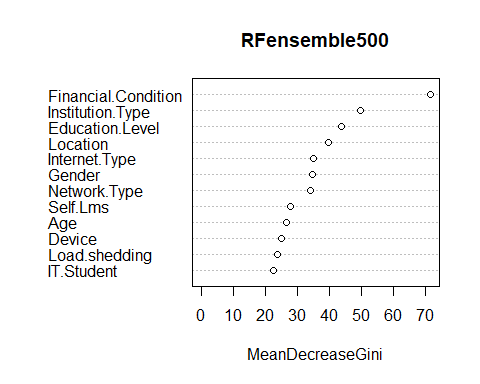
# Create a plot of the RFensemble500 model  
plot(RFensemble500)



# Calculate feature importance for the RFensemble500 model  
importance(RFensemble500)

## MeanDecreaseGini  
## Gender 34.53441  
## Age 26.63351  
## Education.Level 43.65248  
## Institution.Type 49.72422  
## IT.Student 22.57743  
## Location 39.53135  
## Load.shedding 23.61729  
## Financial.Condition 71.48241  
## Internet.Type 35.06856  
## Network.Type 33.99900  
## Self.Lms 27.80801  
## Device 24.94342

# Create a variable importance plot for the RFensemble500 model  
varImpPlot(RFensemble500)



#SAMUEL  
install.packages(c("cluster", "factoextra"), repos = "https://cran.r-project.org")

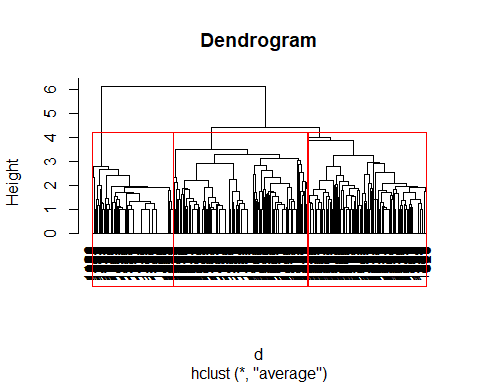
## Installing packages into 'C:/Users/DELL PC/AppData/Local/R/win-library/4.3'  
## (as 'lib' is unspecified)

## package 'cluster' successfully unpacked and MD5 sums checked  
## package 'factoextra' successfully unpacked and MD5 sums checked  
##   
## The downloaded binary packages are in  
## C:\Users\DELL PC\AppData\Local\Temp\RtmpKWnpOB\downloaded\_packages

library(cluster)  
library(factoextra)

## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa

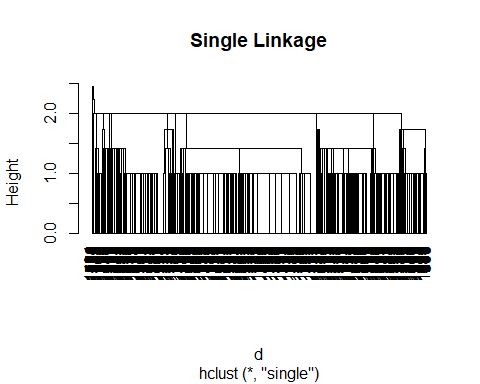
# Load data  
data <- read.csv("converted\_data.csv")  
  
data[, 1:(ncol(data) - 1)] <- lapply(data[, 1:(ncol(data) - 1)], function(x) as.numeric(as.factor(x)))  
numeric\_data <- data[, sapply(data, is.numeric)]  
  
# Impute missing values with column means  
numeric\_data[is.na(numeric\_data)] <- colMeans(numeric\_data, na.rm = TRUE)  
  
# Distance Matrix  
dist\_matrix <- dist(numeric\_data)  
d = dist\_matrix  
hc <- hclust(d, method="average")  
  
# Plot dendrogram  
plot(hc, hang=-1, main="Dendrogram")  
  
# Color labels by cluster (k=3)  
clusters <- cutree(hc, k=3)   
rect.hclust(hc, k=3, border="red")



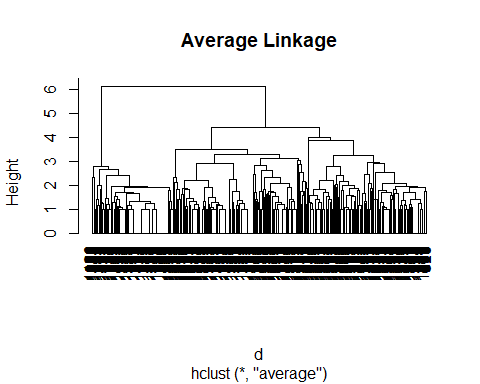
labels\_col <- rainbow(3)[clusters]  
labels(hc, col=labels\_col)

## [1] "merge" "height" "order" "labels" "method"   
## [6] "call" "dist.method"

# Silhouette analysis  
sil <- silhouette(clusters, dist=d)  
mean\_sil <- mean(sil[,3]) # Average silhouette width  
  
# Compare methods   
hc\_single <- hclust(d, method="single")  
hc\_avg <- hclust(d, method="average")  
  
plot(hc\_single, main="Single Linkage", hang=-1)



plot(hc\_avg, main="Average Linkage", hang=-1)



#SAMUEL  
install.packages("mclust", repos = "https://cran.r-project.org")

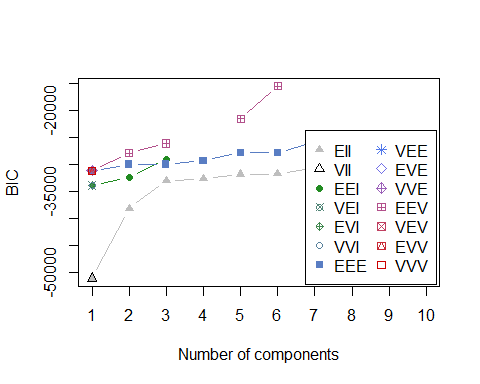
## Installing package into 'C:/Users/DELL PC/AppData/Local/R/win-library/4.3'  
## (as 'lib' is unspecified)

## package 'mclust' successfully unpacked and MD5 sums checked  
##   
## The downloaded binary packages are in  
## C:\Users\DELL PC\AppData\Local\Temp\RtmpKWnpOB\downloaded\_packages

library(mclust)

## Package 'mclust' version 6.0.0  
## Type 'citation("mclust")' for citing this R package in publications.

# Load data  
data <- read.csv("converted\_data.csv")  
  
  
# Preprocess data  
data[, 1:(ncol(data) - 1)] <- lapply(data[, 1:(ncol(data) - 1)], function(x) as.numeric(as.factor(x)))  
numeric\_data <- data[, sapply(data, is.numeric)]  
numeric\_data[is.na(numeric\_data)] <- colMeans(numeric\_data, na.rm = TRUE)  
  
  
# Fit GMM model  
gmm <- Mclust(numeric\_data, G = 1:10)  
  
  
# Calculate silhouette scores  
num\_clusters <- 1:10  
  
  
# Plot BIC and   
plot(gmm, what = "BIC")



#SAMUEL  
install.packages("neuralnet", repos = "https://cran.r-project.org")

## Installing package into 'C:/Users/DELL PC/AppData/Local/R/win-library/4.3'  
## (as 'lib' is unspecified)

## package 'neuralnet' successfully unpacked and MD5 sums checked  
##   
## The downloaded binary packages are in  
## C:\Users\DELL PC\AppData\Local\Temp\RtmpKWnpOB\downloaded\_packages

library(neuralnet)

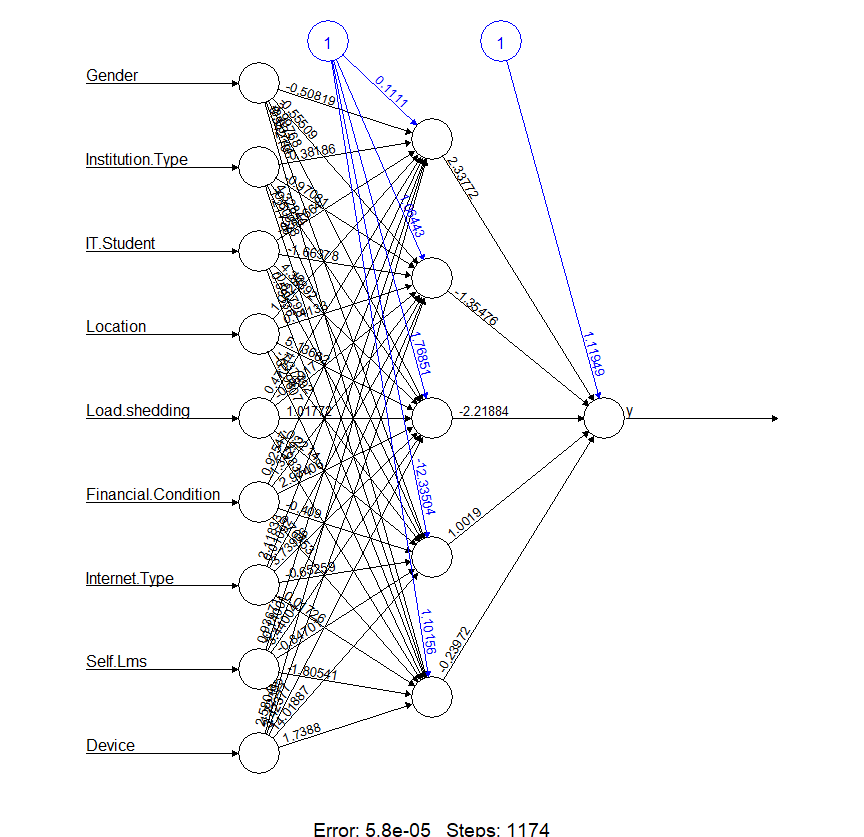
##   
## Attaching package: 'neuralnet'

## The following object is masked from 'package:dplyr':  
##   
## compute

# Read the CSV file  
data <- read.csv("students\_adaptability\_level\_online\_education (1).csv")  
  
# Convert all columns to numeric except the last column  
data[, 1:(ncol(data) - 1)] <- lapply(data[, 1:(ncol(data) - 1)], function(x) as.numeric(as.factor(x)))  
  
  
# Assign input and test variable x and y  
y = as.matrix(data[,12])  
y[which(y=="Low")] = 0  
y[which(y=="High")] = 1  
y = as.numeric(y)  
x = as.numeric(as.matrix(data[,2:11]))  
x = matrix(as.numeric(x), ncol=10)  
  
  
nn <- neuralnet(y ~ Gender + `Institution.Type` + `IT.Student` +  
 Location + `Load.shedding` + `Financial.Condition` +  
 `Internet.Type` + `Self.Lms` + Device,  
 data = data, hidden = 5)  
  
# Predict results  
yy = nn$net.result[[1]]  
yhat = matrix(0, length(y), 1)  
yhat[which(yy > mean(yy))] = 1  
yhat[which(yy <= mean(yy))] = 0  
cm = print(table(y, yhat))

## yhat  
## y 0 1  
## 1 244 0  
## 2 0 1264

# Plot Model  
plot(nn)

  
  
# Model accuracy  
print(sum(diag(cm)) / sum(cm))

## [1] 1