GroupData<-read.csv(file.choose(),header=T)

head(GroupData)

install.packages('dplyr')

library(dplyr)

library(reshape2)

GroupCor<-as.matrix(cor(GroupData))

library(nnet)

set.seed(1000)

GroupNN<-nnet(HighValueCar ~., data=GroupData,size=8,maxit=10000)

GroupNN<-nnet(HighValueCar ~., data=GroupData,size=8,maxit=100000)

GroupData<-read.csv(file.choose(),header=T)

head(GroupData)

GroupNN<-nnet(HighValueCar ~., data=GroupData,size=8,maxit=100000)

GroupNN <- nnet(HighValueCar ~ ., data = GroupData, size = 4, maxit = 100000, decay = 0.01)

GroupData<-read.csv(file.choose(),header=T)

GroupNN <- nnet(HighValueCar ~ ., data = GroupData, size = 4, maxit = 100000, decay = 0.01)

library(NeuralNetTools)

plotnet(GroupData)

plotnet(GroupNN)

install.packages("e1071")

set.seed(123)

sample\_size <- floor(0.7 \* nrow(GroupData))

train\_data <- GroupData[sample(sample\_size, nrow(GroupData)), ]

data <- GroupData[, c("HighValueCar", "brand", "model", "model\_year", "milage", "accident", )]

data <- GroupData[, c("HighValueCar", "brand", "model", "model\_year", "milage", "accident" )]

head(GroupData)

formula <- HighValueCar ~ brand + model\_year + mileage + accident

sample\_size <- floor(0.7 \* nrow(GroupData))

train\_data <- GroupData[sample(sample\_size, nrow(GroupData)), ]

set.seed(123)

sample\_indices <- sample(nrow(GroupData), size = sample\_size, replace = TRUE)

train\_data <- GroupData[sample\_indices, ]

test\_data <- GroupData[-sample\_indices, ]

naive\_bayes\_model <- naiveBayes(formula, data = train\_data)

library(e1071)

naive\_bayes\_model <- naiveBayes(formula, data = train\_data)

formula <- HighValueCar ~ brand + model\_year + mileage + accident

naive\_bayes\_model <- naiveBayes(formula, data = train\_data)

formula <- HighValueCar ~ brand + model\_year + milage + accident

naive\_bayes\_model <- naiveBayes(formula, data = train\_data)

predictions <- predict(naive\_bayes\_model, newdata = test\_data, type = "class")

confusion\_matrix <- table(Actual = test\_data$HighValueCar, Predicted = predictions)

accuracy <- sum(diag(confusion\_matrix)) / sum(confusion\_matrix)

print(confusion\_matrix)

cat("Accuracy:", accuracy)

summary(predictions)

predictions\_all <- predict(naive\_bayes\_model, newdata = GroupData, type = "class")

GroupData$Predicted\_HighValueCar <- predictions\_all

# Naive Bayes Model Evaluation:

predictions <- predict(naive\_bayes\_model, newdata = test\_data, type = "class")

confusion\_matrix <- table(Actual = test\_data$HighValueCar, Predicted = predictions)

# Calculate Precision, Recall, and F1-score

precision <- confusion\_matrix[2, 2] / sum(confusion\_matrix[, 2])

recall <- confusion\_matrix[2, 2] / sum(confusion\_matrix[2, ])

f1\_score <- 2 \* (precision \* recall) / (precision + recall)

# Print the confusion matrix and additional metrics

print(confusion\_matrix)

cat("Accuracy:", accuracy, "\n")

cat("Precision:", precision, "\n")

cat("Recall:", recall, "\n")

cat("F1-score:", f1\_score, "\n")

write.csv(GroupData, "GroupData\_with\_predictions.csv", row.names = FALSE)

# Neural Network Model Evaluation:

GroupData$HighValueCar <- as.factor(GroupData$HighValueCar)

# Check for missing values

missing\_values <- sum(is.na(GroupData))

if (missing\_values > 0) {

cat("Dataset contains missing values. Imputation or data cleaning is required.\n")

} else {

tryCatch({

# Train neural network (intentionally causing an error)

GroupNN <- nnet(HighValueCar ~ ., data = GroupData, size = 8, maxit = 100000)

}, error = function(e) {

cat("Neural network training encountered an error. See details below:\n")

cat(paste("Error: ", e$message, "\n"))

cat("Neural network evaluation is not working on this database.\n")

})

}