ERF Paper

Tegveer Ghura

2023-02-10

# df import and subset  
df <- read\_csv('Data/REKT\_Database\_Clean\_Python.csv')   
df <- subset(df, select = -c(...1, token\_name, description, name\_categories))  
#df <- df %>% filter(funds\_lost!=0)

# Removing dictionary values from the scam\_type column  
df$scam\_type <- gsub("[^:]\*,[^:]\*", "",df$scam\_type)  
df$scam\_type <- gsub("'id'::", "",df$scam\_type)  
df$scam\_type <- gsub("\\{|\\}", "",df$scam\_type)  
df$scam\_type <- gsub("'", "",df$scam\_type)  
df$scam\_type <- gsub("type: ", "",df$scam\_type)  
df$scam\_type <- gsub(" ", "",df$scam\_type)  
  
# Removing list brackets from the scamNetworks column  
df$scamNetworks <- gsub("\\[|\\]", "", df$scamNetworks)  
df$scamNetworks <- gsub("'", '', df$scamNetworks)  
df$scamNetworks <- gsub(", +", ",", df$scamNetworks) # remove whitespace after comma for grouping later

# pooling together scam types into respective types   
df <- df %>%   
 mutate(scam\_type\_grouped = if\_else(scam\_type=="Honeypot" | scam\_type=="Rugpull" | scam\_type=="Abandoned" | project\_name=="Kronos Dao" | project\_name=="Genesis" | project\_name=="Celsius Network" | project\_name=="Voyager" | project\_name=="BlockFi" | project\_name=="FTX Group" | project\_name=="Bitcoin Sheikh" | project\_name=="Trade Coin Club" | project\_name=="EmpiresX" | project\_name=="Vauld", "Exit Scam", "Exploit"))  
df <- subset(df, select = -c(scam\_type, day\_of\_week\_of\_attack, day\_of\_year\_of\_attack, date, project\_name))  
table(df$scam\_type\_grouped)

##   
## Exit Scam Exploit   
## 2677 486

#only month\_of\_attack has NA's (1873 of them), we can impute "unknown" for them or get rid of the column. Let's first see the performance from imputation  
df$month\_of\_attack=month.name[df$month\_of\_attack]  
df$month\_of\_attack[is.na(df$month\_of\_attack)] <- "Unknown"  
#df <- na.omit(df)

# pooling scamNetworks into 5 levels (Eth, binance, polygon, other centralized, other decentralized)  
df <- separate\_rows(df,scamNetworks,sep = ",")  
df <- df %>%   
 mutate(scam\_networks\_grouped = if\_else(scamNetworks == "Avax" | scamNetworks == "Algorand" | scamNetworks == "Arbitrum" | scamNetworks == "Cosmos" | scamNetworks == "Cronos" | scamNetworks == "Elastos" | scamNetworks == "Elrond" | scamNetworks == "EOS" | scamNetworks == "Fantom" | scamNetworks == "Fuse" | scamNetworks == "Gnosis" | scamNetworks == "Harmony" | scamNetworks == "Heco" | scamNetworks == "Klaytn" | scamNetworks == "KuCoin" | scamNetworks == "Moonriver" | scamNetworks == "Near" | scamNetworks == "OKExChain" | scamNetworks == "Optimism" | scamNetworks == "Ronin" | scamNetworks == "RSK" | scamNetworks == "Solana" | scamNetworks == "TRON" | scamNetworks == "Polkadot" | scamNetworks == "Other" | scamNetworks == "Terra Classic", "Other Decentralized", scamNetworks))   
df <- df %>% filter(scam\_networks\_grouped != "") # remove empty string level  
df <- subset(df, select = -c(scamNetworks))

# specify dtypes before train test split  
  
df$scam\_networks\_grouped <- as.factor(df$scam\_networks\_grouped)  
df$scam\_type\_grouped <-as.factor(df$scam\_type\_grouped)  
df$month\_of\_attack <-as.factor(df$month\_of\_attack)  
  
# add +1 because we have zeros in funds\_returned and helps avoid negative inf values  
  
df$log\_funds\_lost <- log(df$funds\_lost + 1)  
df$log\_funds\_returned <- log(df$funds\_returned + 1)   
df <- subset(df, select = -c(funds\_lost, funds\_returned))

library(caret)  
  
set.seed(3738)  
  
df <- df[sample(1:nrow(df)), ] # shuffle rows  
  
train.index <- createDataPartition(df$scam\_networks\_grouped,   
 p = .8, list = FALSE)  
train <- df[ train.index,]  
test <- df[-train.index,]  
  
x\_train <- train %>% select(log\_funds\_lost, log\_funds\_returned,   
 scam\_networks\_grouped)  
y\_train <- train$scam\_type\_grouped  
  
x\_test <- test %>% select(log\_funds\_lost, log\_funds\_returned,   
 scam\_networks\_grouped)  
y\_test <- test$scam\_type\_grouped  
  
classifier\_RF <- randomForest(x = x\_train,  
 y = y\_train,  
 ntree = 500)  
  
classifier\_RF

##   
## Call:  
## randomForest(x = x\_train, y = y\_train, ntree = 500)   
## Type of random forest: classification  
## Number of trees: 500  
## No. of variables tried at each split: 1  
##   
## OOB estimate of error rate: 7.53%  
## Confusion matrix:  
## Exit Scam Exploit class.error  
## Exit Scam 2052 98 0.0455814  
## Exploit 95 318 0.2300242

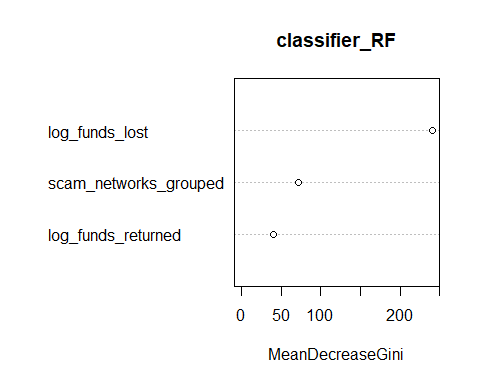
# Predicting the Test set results  
y\_pred = predict(classifier\_RF, newdata = x\_test)  
  
# Confusion Matrix  
confusion\_mtx = table(y\_test, y\_pred)  
confusion\_mtx

## y\_pred  
## y\_test Exit Scam Exploit  
## Exit Scam 504 23  
## Exploit 37 73

#Evaluate variable importance  
importance(classifier\_RF)

## MeanDecreaseGini  
## log\_funds\_lost 240.56063  
## log\_funds\_returned 39.59532  
## scam\_networks\_grouped 71.51662

varImpPlot(classifier\_RF)



library(caret)  
set.seed(2377)  
train.index <- createDataPartition(df$scam\_networks\_grouped,   
 p = .8, list = FALSE)  
train <- df[ train.index,]  
test <- df[-train.index,]  
  
train$scam\_type\_grouped = ifelse(train$scam\_type\_grouped == "Exploit", 1, 0)  
test$scam\_type\_grouped = ifelse(test$scam\_type\_grouped == "Exploit", 1, 0)  
logistic\_model <- glm(scam\_type\_grouped ~ ., data = train, family = binomial(link = "logit"))

# logistic\_model\_summary <- summary(logistic\_model)  
# logistic\_model\_summary  
stargazer::stargazer(logistic\_model,type='html',report = "vc\*stp",  
 ci = TRUE)

Dependent variable:

scam\_type\_grouped

month\_of\_attackAugust

-1.153\*\*

(-2.240, -0.066)

t = -2.078

p = 0.038

month\_of\_attackDecember

-1.396\*\*

(-2.466, -0.326)

t = -2.557

p = 0.011

month\_of\_attackFebruary

-0.707

(-1.839, 0.424)

t = -1.225

p = 0.221

month\_of\_attackJanuary

-2.012\*\*\*

(-3.149, -0.875)

t = -3.468

p = 0.001

month\_of\_attackJuly

-0.318

(-1.414, 0.777)

t = -0.569

p = 0.570

month\_of\_attackJune

-0.479

(-1.581, 0.624)

t = -0.851

p = 0.395

month\_of\_attackMarch

-0.800

(-1.907, 0.307)

t = -1.416

p = 0.157

month\_of\_attackMay

-0.554

(-1.666, 0.558)

t = -0.976

p = 0.330

month\_of\_attackNovember

-1.260\*\*

(-2.316, -0.204)

t = -2.339

p = 0.020

month\_of\_attackOctober

-0.834

(-1.889, 0.220)

t = -1.550

p = 0.122

month\_of\_attackSeptember

-1.214\*\*

(-2.303, -0.125)

t = -2.185

p = 0.029

month\_of\_attackUnknown

-2.399\*\*\*

(-3.631, -1.167)

t = -3.816

p = 0.0002

scam\_networks\_groupedCentralized

0.978\*\*\*

(0.450, 1.506)

t = 3.627

p = 0.0003

scam\_networks\_groupedEthereum

0.784\*\*\*

(0.411, 1.158)

t = 4.113

p = 0.00004

scam\_networks\_groupedOther Decentralized

1.414\*\*\*

(0.780, 2.049)

t = 4.367

p = 0.00002

scam\_networks\_groupedPolygon

0.997\*\*

(0.206, 1.787)

t = 2.471

p = 0.014

log\_funds\_lost

0.235\*\*\*

(0.192, 0.277)

t = 10.918

p = 0.000

log\_funds\_returned

0.113\*\*\*

(0.045, 0.182)

t = 3.233

p = 0.002

Constant

-2.725\*\*\*

(-3.841, -1.608)

t = -4.785

p = 0.00001

Observations

2,563

Log Likelihood

-532.387

Akaike Inf. Crit.

1,102.774

Note:

*p<0.1;* ***p<0.05;*** p<0.01

train\_prob\_pred <- predict(logistic\_model, type = 'response', newdata = train)  
test\_prob\_pred <- predict(logistic\_model, type = 'response', newdata = test)  
#y\_pred = ifelse(prob\_pred > 0.5, "Exploit", "Exit Scam")  
  
# Train Confusion Matrix  
y\_train\_pred = ifelse(train\_prob\_pred > 0.5, 1, 0)  
y\_train\_pred<- as.factor(y\_train\_pred)  
train$scam\_type\_grouped <- as.factor(train$scam\_type\_grouped)  
(cm = table(train$scam\_type\_grouped, y\_train\_pred))

## y\_train\_pred  
## 0 1  
## 0 2068 87  
## 1 128 280

# Test Confusion Matrix  
y\_test\_pred = ifelse(test\_prob\_pred > 0.5, 1, 0)  
y\_test\_pred<- as.factor(y\_test\_pred)  
test$scam\_type\_grouped <- as.factor(test$scam\_type\_grouped)  
(cm = table(test$scam\_type\_grouped, y\_test\_pred)) # NAs ignored

## y\_test\_pred  
## 0 1  
## 0 508 14  
## 1 38 77

#y\_pred <- as.factor(unname(y\_pred)) # for cfm plot

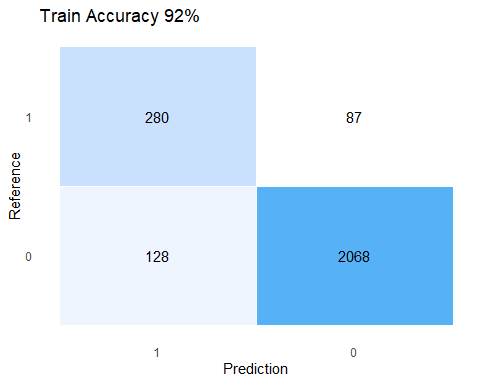
# 1. Open jpeg file  
#jpeg("Train\_CFM.jpg", width = 350, height = 350)  
  
library(scales)

##   
## Attaching package: 'scales'

## The following object is masked from 'package:purrr':  
##   
## discard

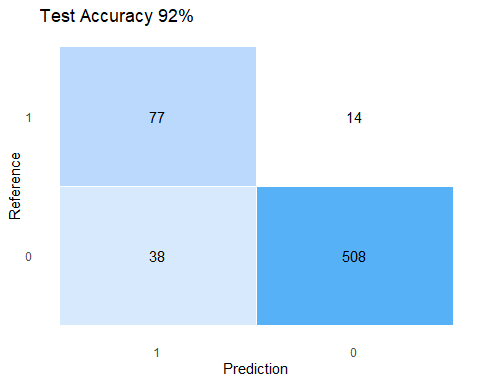
## The following object is masked from 'package:readr':  
##   
## col\_factor

ggplotConfusionMatrix <- function(m){  
 mytitle <- paste("Train Accuracy", percent\_format()(m$overall[1]))  
 p <-  
 ggplot(data = as.data.frame(m$table) ,  
 aes(x = Prediction, y = Reference)) +  
 geom\_tile(aes(fill = log(Freq)),   
 colour = "white", show.legend = FALSE) +  
 scale\_fill\_gradient(low = "white", high = "#56B1F7") +  
 geom\_text(aes(x = Prediction, y = Reference,   
 label = Freq)) +  
 ggtitle(mytitle) +   
 scale\_x\_discrete(limits = rev) +  
 theme\_minimal() +   
 theme(panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank())  
 return(p)  
}  
cfm\_train <- confusionMatrix(train$scam\_type\_grouped, y\_train\_pred)  
ggplotConfusionMatrix(cfm\_train)



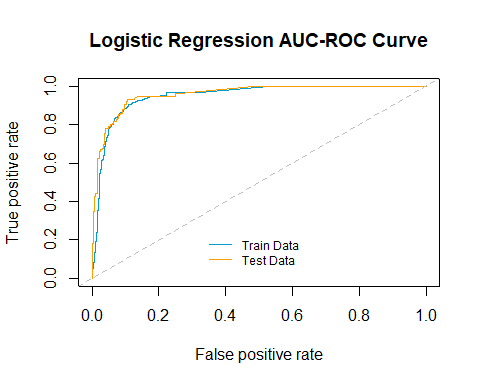
# Close the jpeg file  
#dev.off()

# Open jpeg file  
#jpeg("Test\_CFM.jpg", width = 350, height = 350)  
  
library(scales)  
ggplotConfusionMatrix <- function(m){  
 mytitle <- paste("Test Accuracy", percent\_format()(m$overall[1]))  
 p <-  
 ggplot(data = as.data.frame(m$table) ,  
 aes(x = Prediction, y = Reference)) +  
 geom\_tile(aes(fill = log(Freq)),   
 colour = "white", show.legend = FALSE) +  
 scale\_fill\_gradient(low = "white", high = "#56B1F7") +  
 geom\_text(aes(x = Prediction, y = Reference,   
 label = Freq)) +  
 ggtitle(mytitle) +   
 scale\_x\_discrete(limits = rev) +  
 theme\_minimal() +   
 theme(panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank())  
 return(p)  
}  
cfm\_test <- confusionMatrix(test$scam\_type\_grouped, y\_test\_pred)  
ggplotConfusionMatrix(cfm\_test)



# Close the jpeg file  
#dev.off()

# Train and Test Data ROC-AUC Curve  
train\_pred <- prediction(train\_prob\_pred, train$scam\_type\_grouped)  
test\_pred <- prediction(test\_prob\_pred, test$scam\_type\_grouped)  
  
# Create an ROC curve  
perf\_train <- performance(train\_pred, measure = "tpr", x.measure = "fpr")  
perf\_test <- performance(test\_pred, measure = "tpr", x.measure = "fpr")  
  
# Open a pdf and jpeg file  
#pdf("ROC.pdf", width = 6.5, height = 4.24)   
#jpeg("ROC.jpg", width = 700, height = 350)  
  
  
# Plot the ROC curve  
plot(perf\_train, main = "Logistic Regression AUC-ROC Curve",  
 col = "#009ECE")  
plot(perf\_test, add = T, col = "#FF9E00", lwd = 1.5)  
legend(0.32, 0.25, c("Train Data", "Test Data"),   
 col = c("#009ECE", "#FF9E00"),   
 bty = "n", lwd = 1.2, cex = 0.75)  
abline(0, 1, lty = 2, col = "gray") # Add y=x line



# Close the pdf/jpeg file  
#dev.off()

auc.train <- auc(train$scam\_type\_grouped, train\_prob\_pred)  
cat("Area under the curve for Logistic Regression Train Set is: ", auc.train)

## Area under the curve for Logistic Regression Train Set is: 0.9500398

auc.test <- auc(train$scam\_type\_grouped, train\_prob\_pred)  
cat("\nArea under the curve for Logistic Regressio Test Set is: ", auc.test)

##   
## Area under the curve for Logistic Regressio Test Set is: 0.9500398