Credit Card Fraud Detection

```
  #SAVE THE ORIGINAL FILE
    oldfile <- write.csv(creditcard, file = "oldfile.csv", row.names = FALSE, na = "")

    #CLEAN THE FILE. SAVE THE CLEAN. IMPORT THE CLEAN FILE. CHANGE
THE TO A DATAFRAME.
    cleandata <- creditcard[complete.cases(creditcard),]
    cleanfile <- write.csv(cleandata, file = "cleanfile.csv", row.names = FALSE, na = "")
cleanfileread <- read.csv(file = "cleanfile.csv")      cleanfiledata <-
as.data.frame(cleanfileread)

    #SUBSETTING THE DATA TO TYPES
    logicmeint <- cleanfiledata[,sapply(cleanfiledata,is.integer)]
logicmedouble <- cleanfiledata[,sapply(cleanfiledata,is.double)]
logicmefactor <- cleanfiledata[,sapply(cleanfiledata,is.factor)]
logicmenum <- cleanfiledata[,sapply(cleanfiledata,is.numeric)]
mainlogicmefactors <- cleanfiledata[,sapply(cleanfiledata,is.factor) |
sapply(cleanfiledata,is.numeric)]

 #VIEW ALL FILES
    View(cleanfiledata)
    View(logicmeint)
    View(logicmedouble)
    View(logicmefactor)
    View(logicmenum)
    View(mainlogicmefactors)


    #ANALYTICS OF THE MAIN DATAFRAME
    cleansum <- summary(cleanfiledata)
print(cleansum)
    cleandec <- describe(cleanfiledata)
print(cleandec)
    save(cleanfiledata, logicmeint, mainlogicmefactors, logicmedouble, logicmefactor,
logicmenum, numberdec, numbersum, factordec, factorsum, cleandec, oldfile, cleandata,
cleanfile, cleanfileread,   file = "cleanmework.RData")
  }
  cleanme(creditcard)
  timeamt<-cleandata[sample(1:nrow(cleandata),1000,replace = FALSE),]
timeamt library(tidyverse) library(sparklyr) library(readxl)
sc<-spark_connect(master = "local")
if(!file.exists("Data/spark-warehouse/credit-transactions"))
{
```

```
  credit_transactions <- read_csv("/home/ishita/Downloads/creditcard.csv")
head(credit_transactions)
  dim(credit_transactions)
  credit_transactions_tbl <- copy_to(sc, credit_transactions, "credit_transactions", overwrite =
TRUE)
  src_tbls(sc)
  spark_write_parquet(credit_transactions_tbl, str_c("file:", getwd(),
"/Data/sparkwarehouse/credit-transactions"),mode = "overwrite")
}
credit_transactions_tbl <- spark_read_parquet(sc, "credit_transactions", str_c("file:", getwd(),
"/Data/spark-warehouse/credit-transactions"), mode =
"overwrite") credit_transactions_tbl %>%   sample_n(10)
library(dplyr) library(arulesSequences)
credit_transactions=credit_transactions %>% select(Time, Amount, Class)
credit_transactions summary(credit_transactions)
fraud_transactions=credit_transactions$Class==
1 fraud_transactions sum(fraud_transactions)
subset<-credit_transactions[sample(1:nrow(credit_transactions),10000,replace = FALSE),]
subset
with(subset,plot(Time,Amount))
number_no_fraud=sum(credit_transactions$Class==0) number_no_fraud
number_fraud=sum(credit_transactions$Class==1) number_fraud
paste0("There are only ",number_fraud, " frauds in the original datasets, even though there
are ",number_no_fraud," no frauds in the datasets.")
paste0("The accuracy of the classifier then would be ",((284315-492)/284315)," which is the
number of good classification over the number of tuple") corr<-cor(credit_transactions)
corr
data<-
as.matrix(credit_transactions)
heatmap(credit_transactions)
heatmap(data)
data<-as.matrix(subset)
heatmap(data) heatmap(data, scale =
"column") rank<-corr["Class"] rank


#feedback mechanism

library(caTools)
shuffle<-sample(2, nrow(credit_transactions), replace=TRUE, prob=c(0.67, 0.33))
shuffle
credit_transactions.training <- credit_transactions[shuffle==1, 1:3]
head(credit_transactions.training)
```

```
credit_transactions.test <- credit_transactions[shuffle==2, 1:3]
head(credit_transactions.test)

credit_transactions.trainLabels <- credit_transactions[shuffle==1,3]
print(credit_transactions.trainLabels)

credit_transactions.testLabels <- credit_transactions[shuffle==2,
3] print(credit_transactions.testLabels) library(class)
cl = credit_transactions.trainLabels[,1]
cl=train[,3]
credit_pred <- knn(train = credit_transactions.training[,2,drop=FALSE], test =
credit_transactions.test[,2,drop=FALSE],cl , k=3)
```