ADS-503 Team 4 EDA

Leonard Littleton Lina Nguyen Emanuel Lucban

6/25/21

```
library(readr)
library(data.table)
library(mlbench)
library(ggplot2)
library(tidyr)
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
library(e1071)
library(caret)
```

```
## Loading required package: lattice
```

```
library(naniar)
library(MLmetrics)
```

```
##
## Attaching package: 'MLmetrics'
```

```
## The following objects are masked from 'package:caret':
##
##     MAE, RMSE
```

```
## The following object is masked from 'package:base':
##
##     Recall
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:data.table':
##
##     between, first, last
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

Synthetic Financial Datasets For Fraud Detection

```
synth_df <- fread('./data/PS_20174392719_1491204439457_log.csv', header = TRUE)
```

Exploratory Data Analysis

```
head(synth_df)
```

```
##    step     type   amount    nameOrig oldbalanceOrg newbalanceOrig    nameDest
## 1:    1  PAYMENT  9839.64 C1231006815        170136      160296.36 M1979787155
## 2:    1  PAYMENT  1864.28 C1666544295         21249       19384.72 M2044282225
## 3:    1 TRANSFER   181.00 C1305486145           181           0.00  C553264065
## 4:    1 CASH_OUT   181.00  C840083671           181           0.00   C38997010
## 5:    1  PAYMENT 11668.14 C2048537720         41554       29885.86 M1230701703
## 6:    1  PAYMENT  7817.71   C90045638         53860       46042.29  M573487274
##    oldbalanceDest newbalanceDest isFraud isFlaggedFraud
## 1:              0              0       0              0
## 2:              0              0       0              0
## 3:              0              0       1              0
## 4:          21182              0       1              0
## 5:              0              0       0              0
## 6:              0              0       0              0
```

```
synth_df$isFraud <- as.factor(synth_df$isFraud)
synth_df$isFlaggedFraud <- as.factor(synth_df$isFlaggedFraud)
summary(synth_df)
```

```
##       step           type               amount           nameOrig
##  Min.   :  1.0   Length:6362620     Min.   :       0   Length:6362620
##  1st Qu.:156.0   Class :character   1st Qu.:   13390   Class :character
##  Median :239.0   Mode  :character   Median :   74872   Mode  :character
##  Mean   :243.4                      Mean   :  179862
##  3rd Qu.:335.0                      3rd Qu.:  208721
##  Max.   :743.0                      Max.   :92445517
##  oldbalanceOrg      newbalanceOrig       nameDest         oldbalanceDest
##  Min.   :       0   Min.   :       0   Length:6362620     Min.   :        0
##  1st Qu.:       0   1st Qu.:       0   Class :character   1st Qu.:        0
##  Median :   14208   Median :       0   Mode  :character   Median :   132706
##  Mean   :  833883   Mean   :  855114                      Mean   :  1100702
##  3rd Qu.:  107315   3rd Qu.:  144258                      3rd Qu.:   943037
##  Max.   :59585040   Max.   :49585040                      Max.   :356015889
##  newbalanceDest      isFraud     isFlaggedFraud
##  Min.   :        0   0:6354407   0:6362604
##  1st Qu.:        0   1:   8213   1:     16
##  Median :   214661
##  Mean   :  1224996
##  3rd Qu.:  1111909
##  Max.   :356179279
```

Null Values

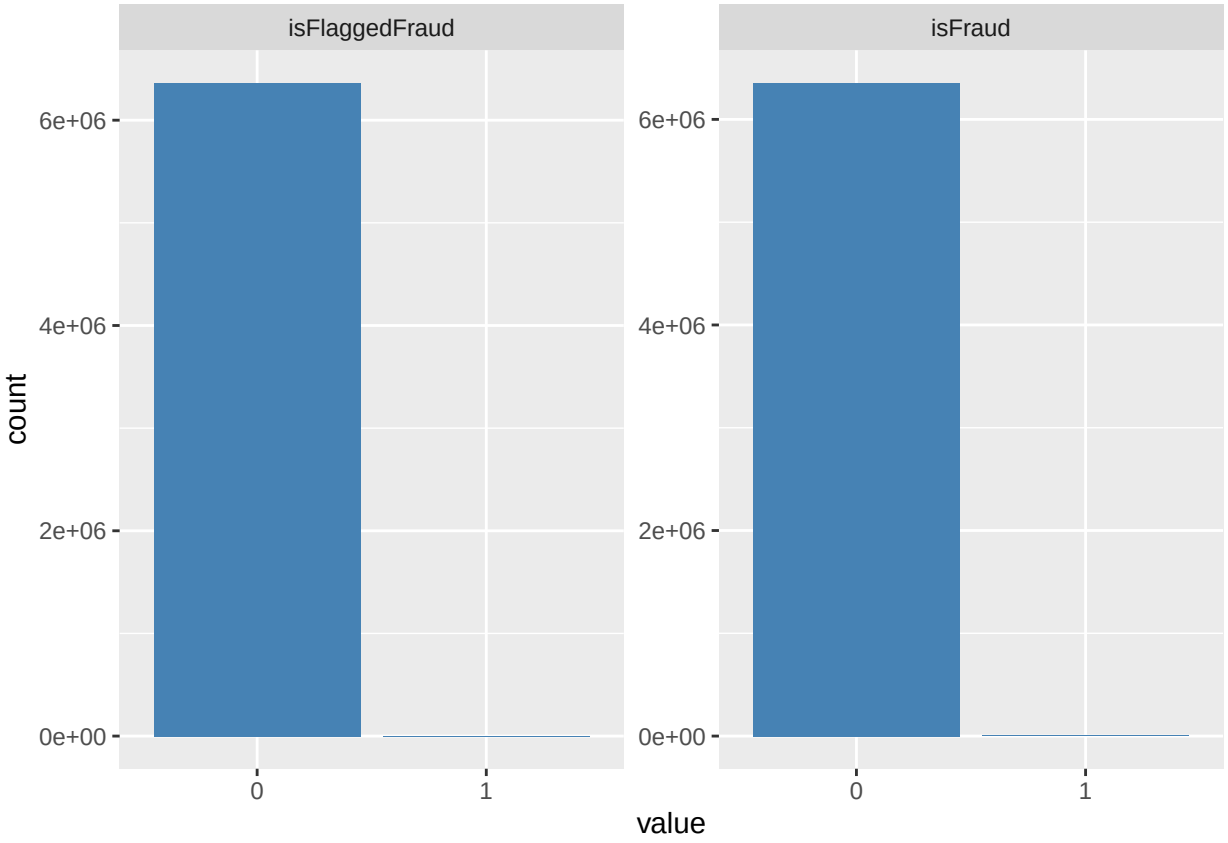
```
sapply(synth_df, function(x) sum(is.na(x)))
```

```
##            step            type          amount        nameOrig  oldbalanceOrg
##               0               0               0               0              0
## newbalanceOrig        nameDest oldbalanceDest newbalanceDest         isFraud
##               0               0               0               0              0
## isFlaggedFraud
##               0
```

There are no null values in any of the predictors.

Target Variable Distribution

```
ggplot(gather(synth_df[, 10:11]), aes(value)) + geom_bar(fill = "steelblue") + facet_wrap(~key, scales =
```

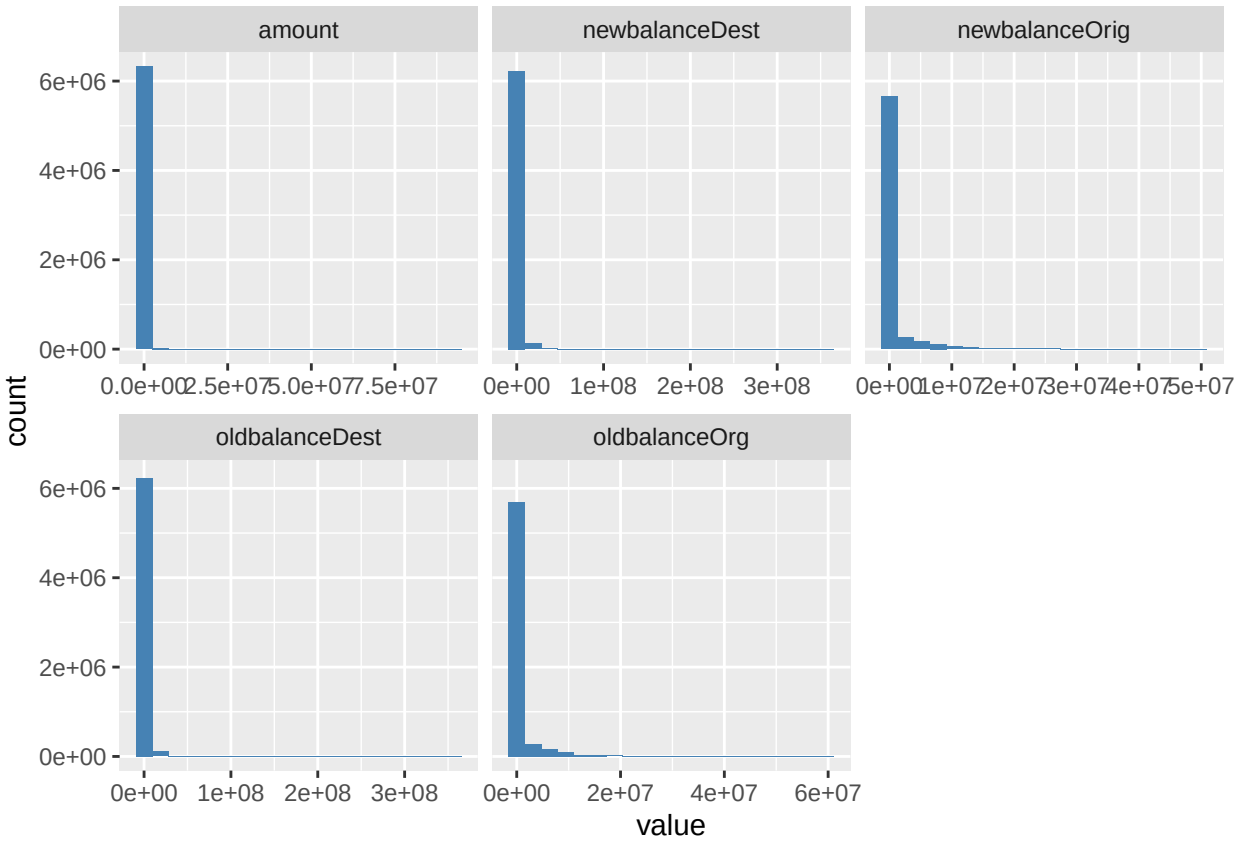


The data dictionary describes the variable isFlaggedFraud as an attempt to transfer more than 200,000 in a single transaction. From the summary, there are only 16 positive observations. We have decided to focus our attention to the isFraud variable as our target and will drop isFlaggedFraud from the dataset.

```
synth_df <- subset(synth_df, select = -c(isFlaggedFraud))
```

Continuous Predictors Distribution

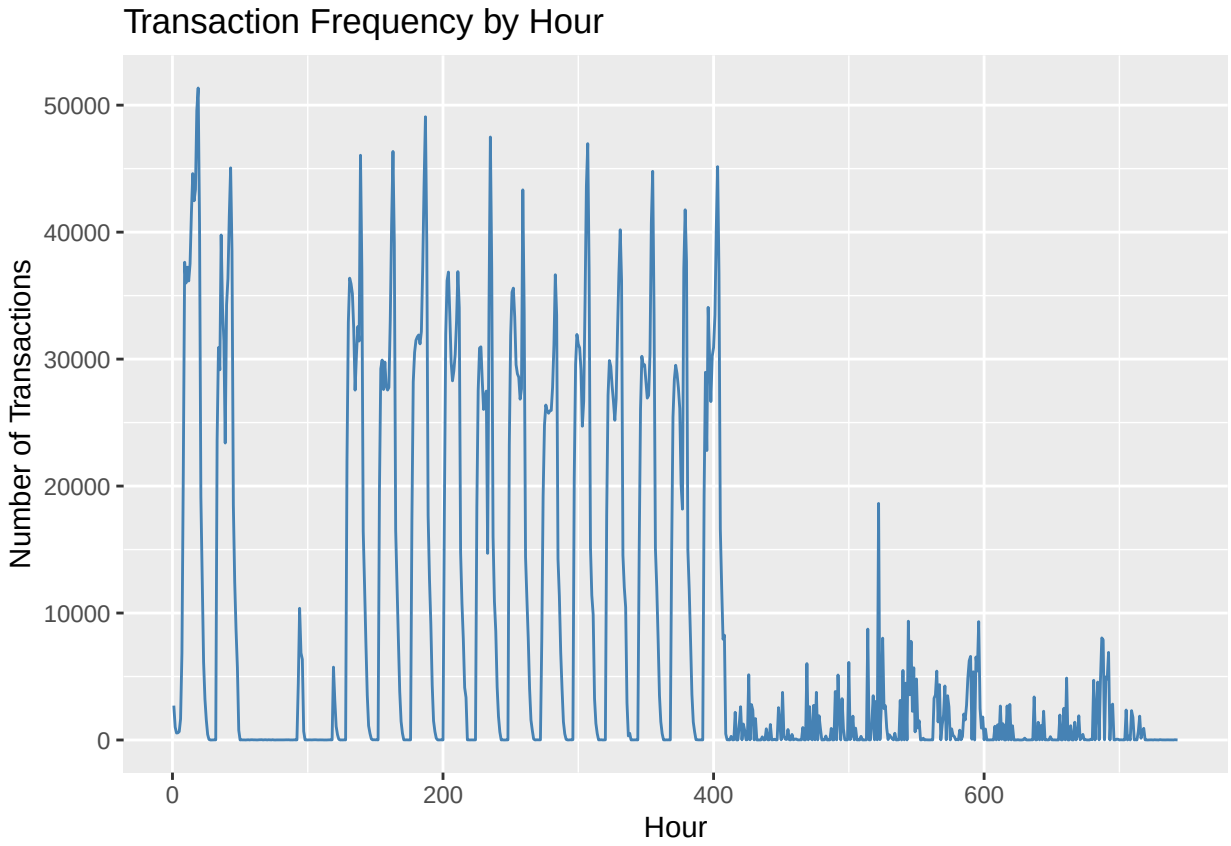
```
ggplot(gather(synth_df[, c('amount', 'oldbalanceOrg', 'newbalanceOrig', 'oldbalanceDest', 'newbalanceDes
geom_histogram(bins = 20, fill = "steelblue") +
facet_wrap(~key, scales = 'free_x')
```



From the continuous distributions above there is a high right skew due to the presence of large outliers, meaning small number of customers with very large balances. Box-Cox or Log transformations can be performed to mitigate the effect of the outliers.

Transaction Frequency by Hour

```
ggplot(data = synth_df, aes(x=step)) +
  geom_line(stat = 'count', color = "steelblue") + xlab("Hour") + ylab("Number of Transactions") +
  ggtitle("Transaction Frequency by Hour")
```



The predictor Step is an interval of time by hour, the dataset is a collection transactions for each hour for an approximate 30 day period, in this case 744 hours total. The barplot represents the number of transactions for each hour of the 30 day period. This predictor will be further analyzed by re-sampling time intervals by intraday (mod 24), intraweek (mod 168), etc.

Count of unique values from Nominal predictors

```
nominals <- c('type', 'nameOrig', 'nameDest')
data.frame(predictor = nominals, count = c(length(unique(synth_df[,'type']$type)),
                                           length(unique(synth_df[, 'nameOrig']$nameOrig)),
                                           length(unique(synth_df[, 'nameDest']$nameDest))))
```

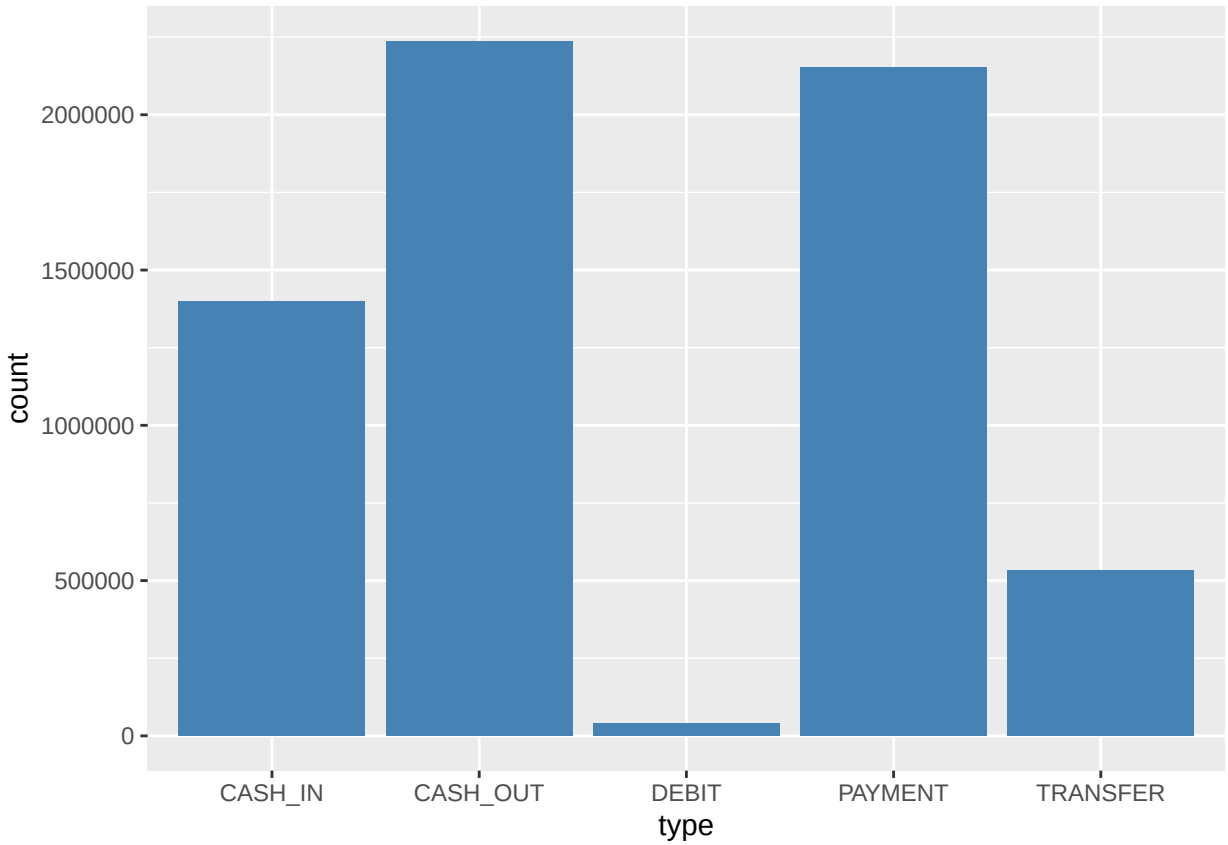
```
##   predictor   count
## 1      type       5
## 2  nameOrig 6353307
## 3  nameDest 2722362
```

Due to the large amount of unique categorical values of the predictors nameOrig and nameDest, it would be impractical to perform one-hot encoding on these values so it was decided that these predictors will be excluded.

```
reduced_df <- subset(synth_df, select = -c(nameOrig, nameDest))
```

Distribution of Nominal Predictors

```
ggplot(data = reduced_df, aes(type)) + geom_bar(fill = "steelblue")
```



Near Zero Variance

```
nearZeroVar(subset(reduced_df, select = -c(isFraud)))
```

```
## integer(0)
```

There are no degenerate distributions in the dataset.

Correlation

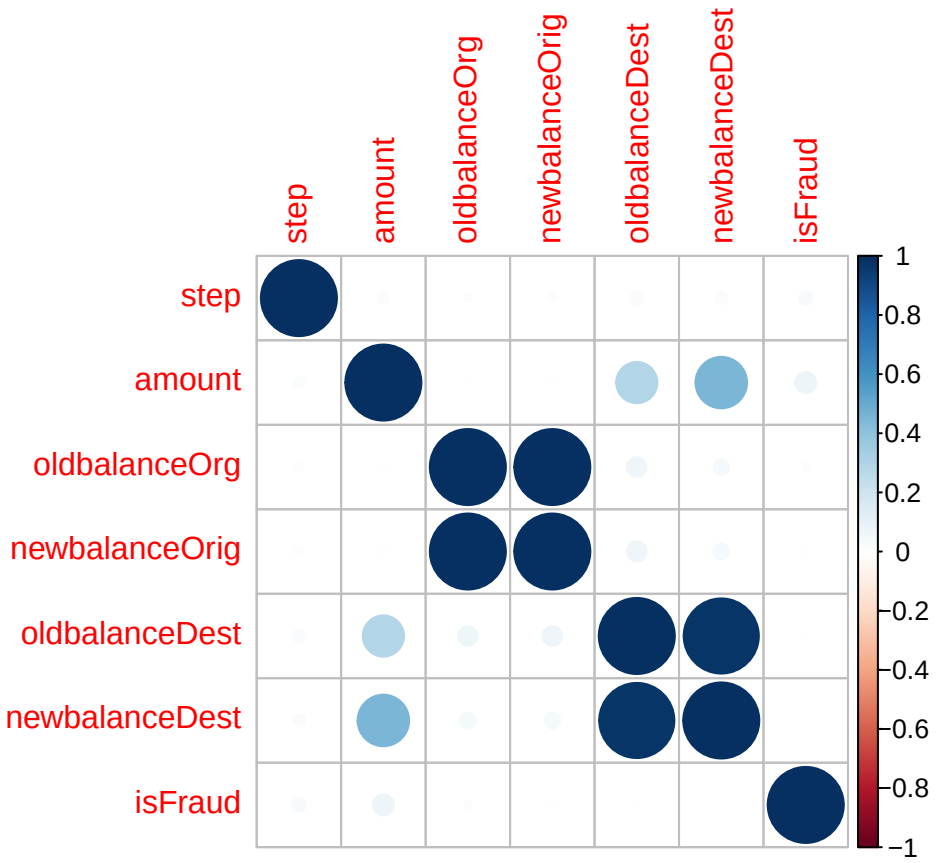
```
reduced_df$isFraud = as.numeric(as.character(reduced_df$isFraud))

# exclude type
data_corr <- cor(subset(reduced_df, select = -c(type)))
data_corr
```

```
##                      step       amount oldbalanceOrg newbalanceOrig
## step            1.00000000  0.022372995  -0.010058378   -0.010299037
## amount          0.02237299  1.000000000  -0.002762475   -0.007860925
## oldbalanceOrg  -0.01005838 -0.002762475   1.000000000    0.998802763
## newbalanceOrig -0.01029904 -0.007860925   0.998802763    1.000000000
## oldbalanceDest  0.02766536  0.294137450   0.066242501    0.067811518
## newbalanceDest  0.02588818  0.459304267   0.042028619    0.041837497
## isFraud         0.03157757  0.076688429   0.010154422   -0.008148161
##                oldbalanceDest newbalanceDest       isFraud
## step              0.027665360   0.0258881757  0.0315775686
## amount            0.294137450   0.4593042673  0.0766884288
```

```
## oldbalanceOrg     0.066242501   0.0420286188  0.0101544219
## newbalanceOrig    0.067811518   0.0418374971 -0.0081481613
## oldbalanceDest    1.000000000   0.9765685054 -0.0058852782
## newbalanceDest    0.976568505   1.0000000000  0.0005353471
## isFraud          -0.005885278   0.0005353471  1.0000000000
```

```
corrplot::corrplot(data_corr)
```



As shown by the heatplot, the predictor pairs (newbalanceOrg, oldbalanceOrg) and (newbalanceDest, oldbalanceDest) are correlated, which is expected since each datapoint represents a transaction and the old and new balances represent pre and post transaction. However, correlated predictors may cause issues during the modeling process and decorrelation may be necessary.

Analysis of fraudulent transactions

Average amount of fraudulent transactions

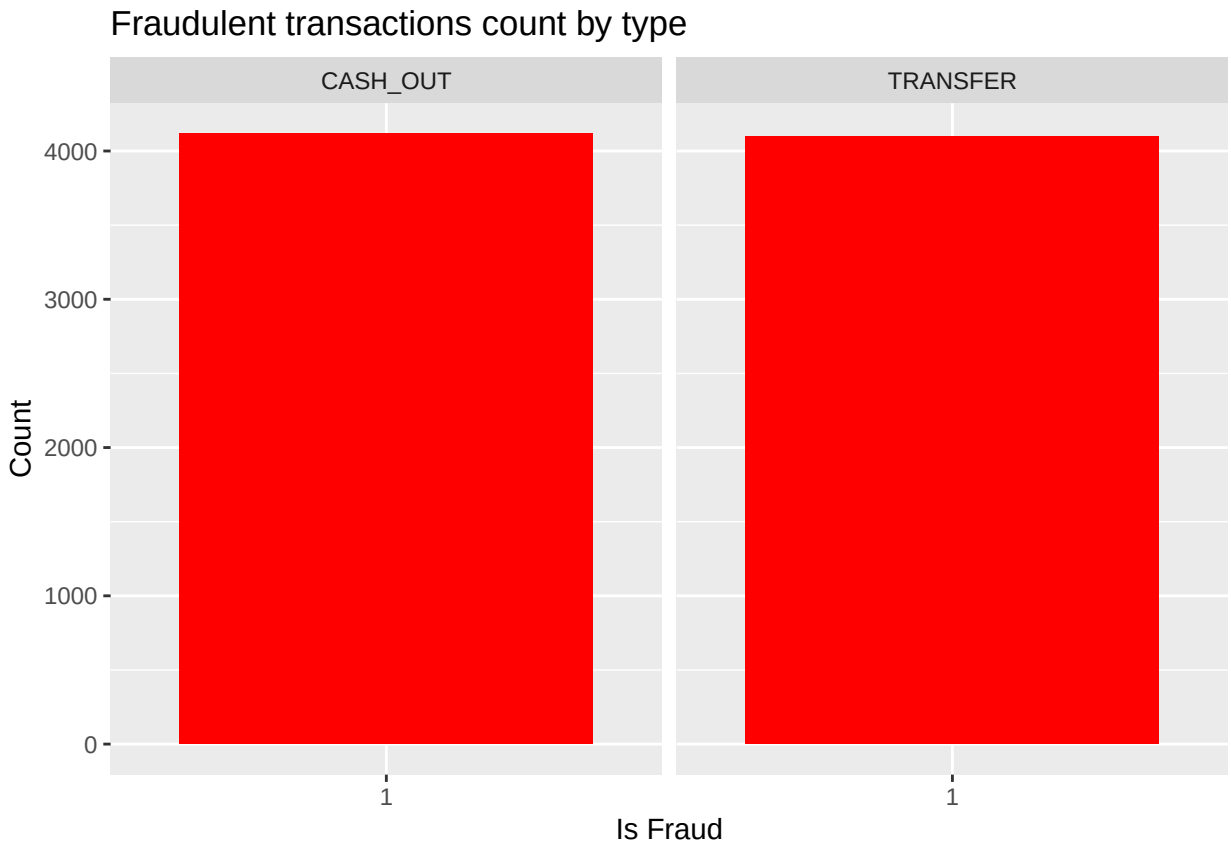
```
median(reduced_df[reduced_df$isFraud == 1, ]$amount)
```

```
## [1] 441423.4
```

Due to the extreme right skew, the median was used instead of the mean. The median amount of fraudlent transactions is 441423.4 (local currency)

Fraudulent transactions by type

```
ggplot(data = reduced_df[reduced_df$isFraud == 1, ], aes(x=as.factor(isFraud))) +
  geom_bar(stat = 'count', fill = "red") +
  ggtitle(label = "Fraudulent transactions count by type") + xlab('Is Fraud') + ylab('Count') +
  facet_wrap(~type)
```

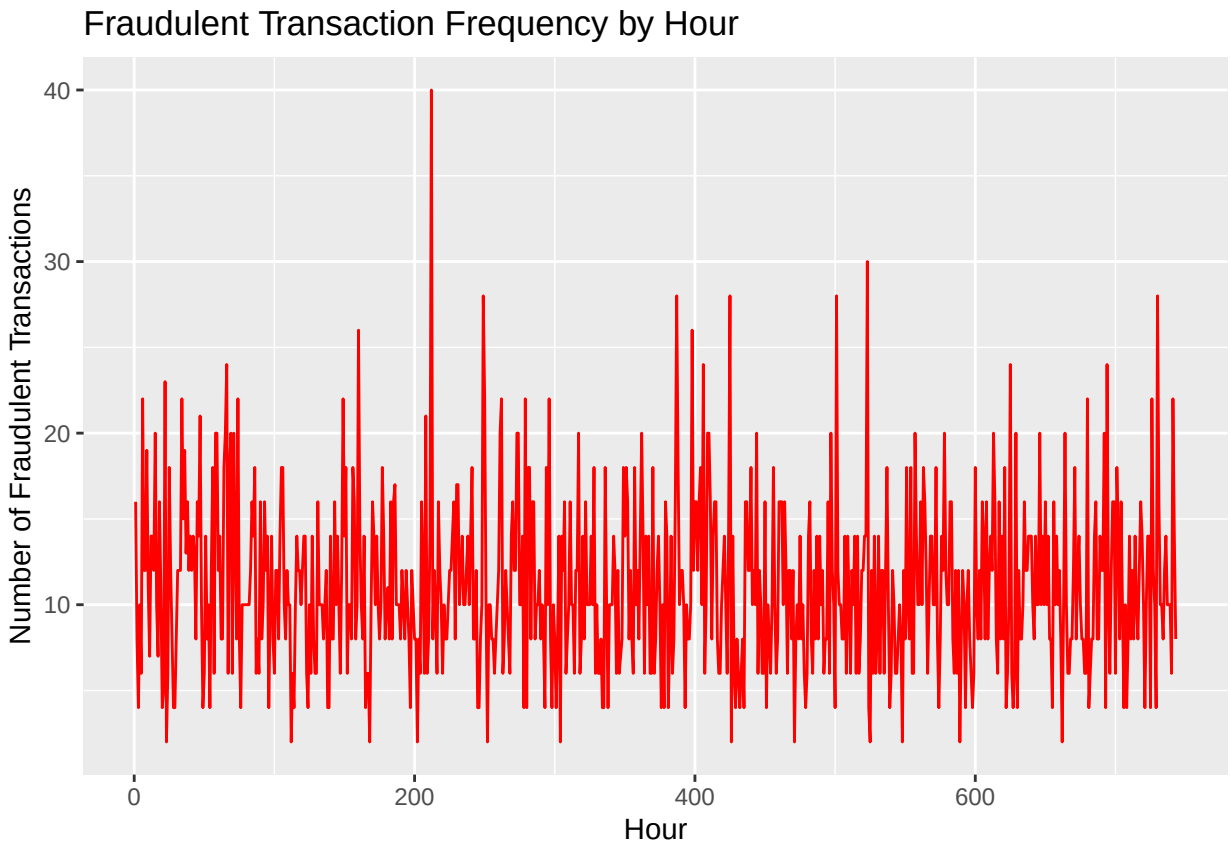


Fraudulent transactions in the dataset **only** occur when the transaction type is CASH_OUT or TRANSFER

Frequency of fraudulent transactions by each hour

```
fraud_by_hour <- reduced_df[reduced_df$isFraud == 1, ] %>%
                    group_by(step) %>%
                    count(step)

ggplot(data = fraud_by_hour, aes(x=step, y=n)) + geom_line(stat = 'identity', color="red") +
  ggtitle("Fraudulent Transaction Frequency by Hour") +
  xlab("Hour") + ylab("Number of Fraudulent Transactions")
```



Time Series Analysis - Downsampling

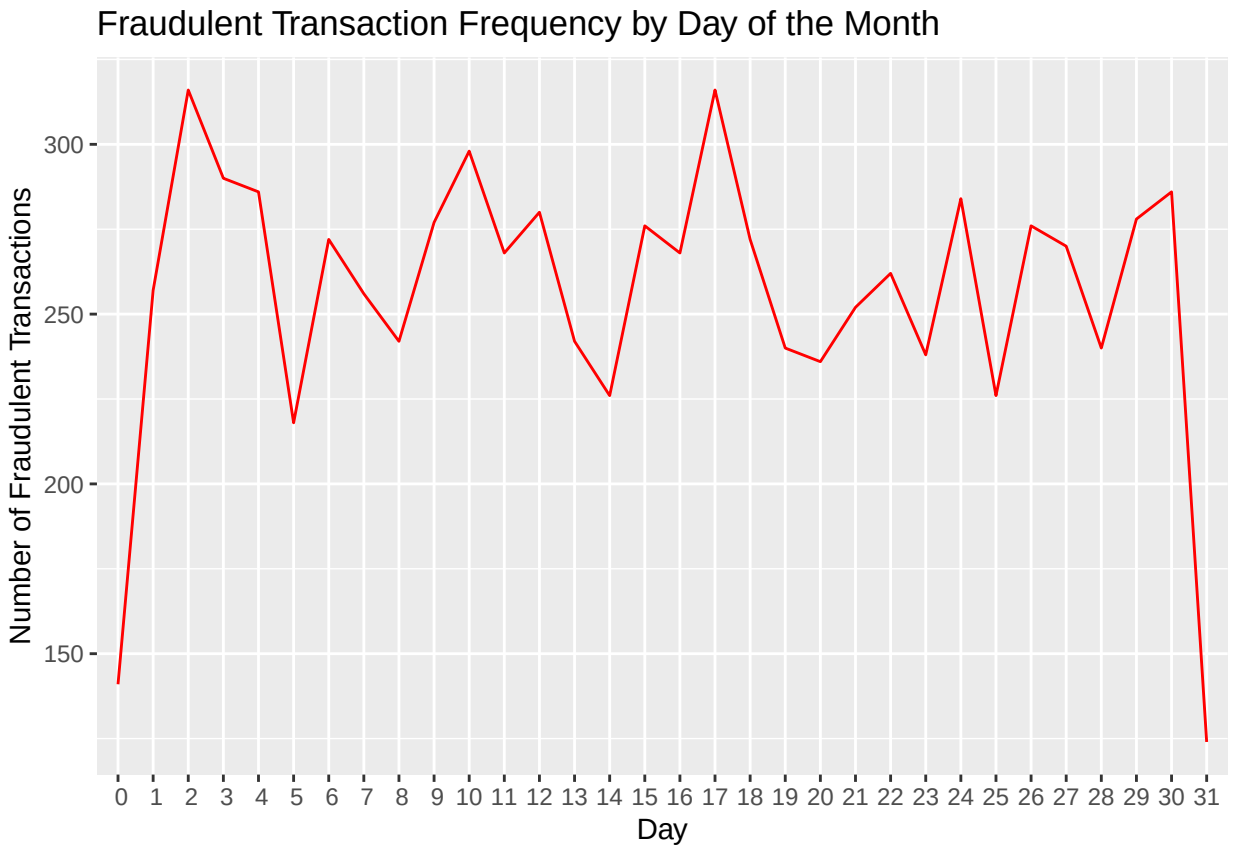
In order to further analyze patterns in fraudulent activity, we need to resample the time series frequency of step from hours to days and weeks.

```
# By Day will group data points by the day of the month
# Hours Intraday will group data points by the hour of day
# Day of Week will group data points by the day of the week
reduced_df <- cbind(reduced_df, data.frame(hours_intraday = reduced_df$step %% 24,
                                           by_day = round(reduced_df$step / 24),
                                           day_of_week = round(reduced_df$step / 24) %% 7))
```

Fraudulent Transaction Frequency by Day of the Month

```
fraud_byday <- reduced_df[reduced_df$isFraud == 1, ] %>%
                    group_by(by_day) %>%
                    count(by_day)

ggplot(data = fraud_byday, aes(x=by_day, y=n)) + geom_line(stat = 'identity', color="red") +
  ggtitle("Fraudulent Transaction Frequency by Day of the Month") +
  xlab("Day") + ylab("Number of Fraudulent Transactions") +
  scale_x_discrete(limits = 0:31)
```

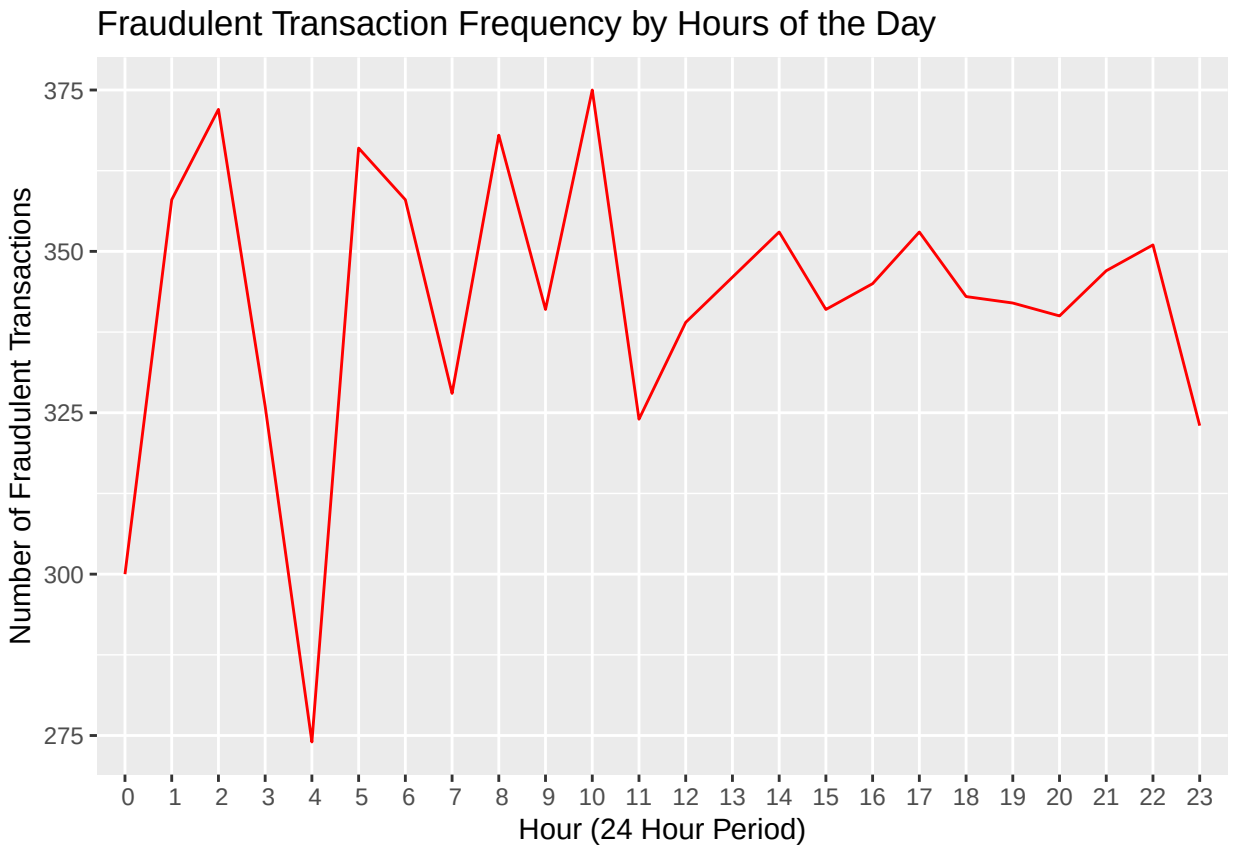


Aggregation by day of the month shows peaks of fraudulent transactions on the 2nd and 17th day. The lowest recorded fraudulent transactions occured on the 5th day.

Intraday Fraudulent Transactions

```
fraud_intraday <- reduced_df[reduced_df$isFraud == 1, ] %>%
                    group_by(hours_intraday) %>%
                    count(hours_intraday)

ggplot(data = fraud_intraday, aes(x=hours_intraday, y=n)) + geom_line(stat = 'identity', color="red") +
  ggtitle("Fraudulent Transaction Frequency by Hours of the Day") +
  xlab("Hour (24 Hour Period)") + ylab("Number of Fraudulent Transactions") +
  scale_x_discrete(limits = 0:23)
```

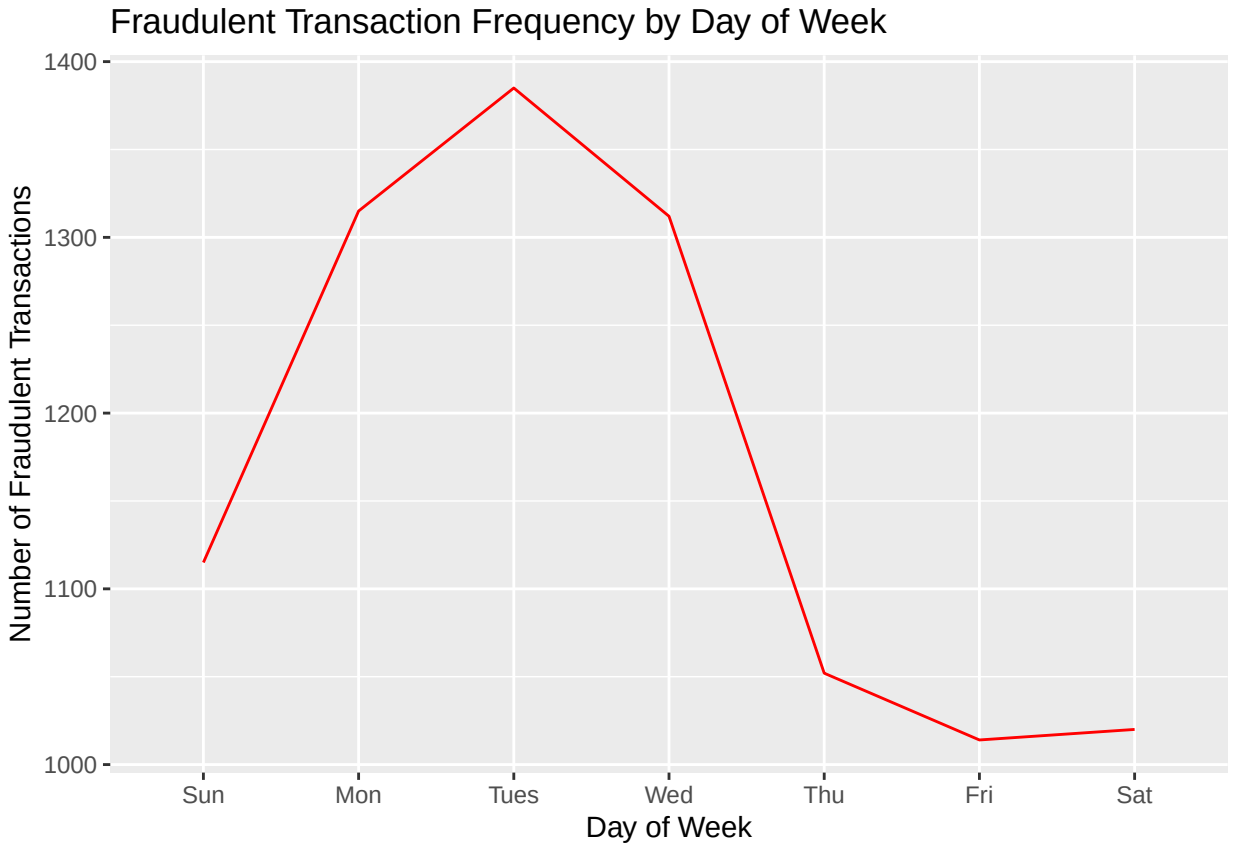


Aggregation by hours of the day reveals that the lowest number of fraudulent activity occurs at 4am and peaks at 10am.

Fraudulent Transaction Frequency by Day of Week

```
fraud_dayofweek <- reduced_df[reduced_df$isFraud == 1, ] %>%
                    group_by(day_of_week) %>%
                    count(day_of_week)

ggplot(data = fraud_dayofweek, aes(x=day_of_week, y=n)) + geom_line(stat = 'identity', color="red") +
  ggtitle("Fraudulent Transaction Frequency by Day of Week") +
  xlab("Day of Week") + ylab("Number of Fraudulent Transactions") +
  scale_x_discrete(limits = 0:6, labels = c('Sun', 'Mon', 'Tues', 'Wed', 'Thu', 'Fri', 'Sat'))
```



Aggregation by day of the week reveals that most fraudulent transactions occur from Monday - Wednesday and peaks on Tuesdays. The lowest number of fraudulent transactions occurs on Fridays.

Data Preparation

Log Transformation of Continuous Predictors

In order to deal with the outliers and extreme right skew of the continuous variables, we will perform log transformations for each continuous predictor

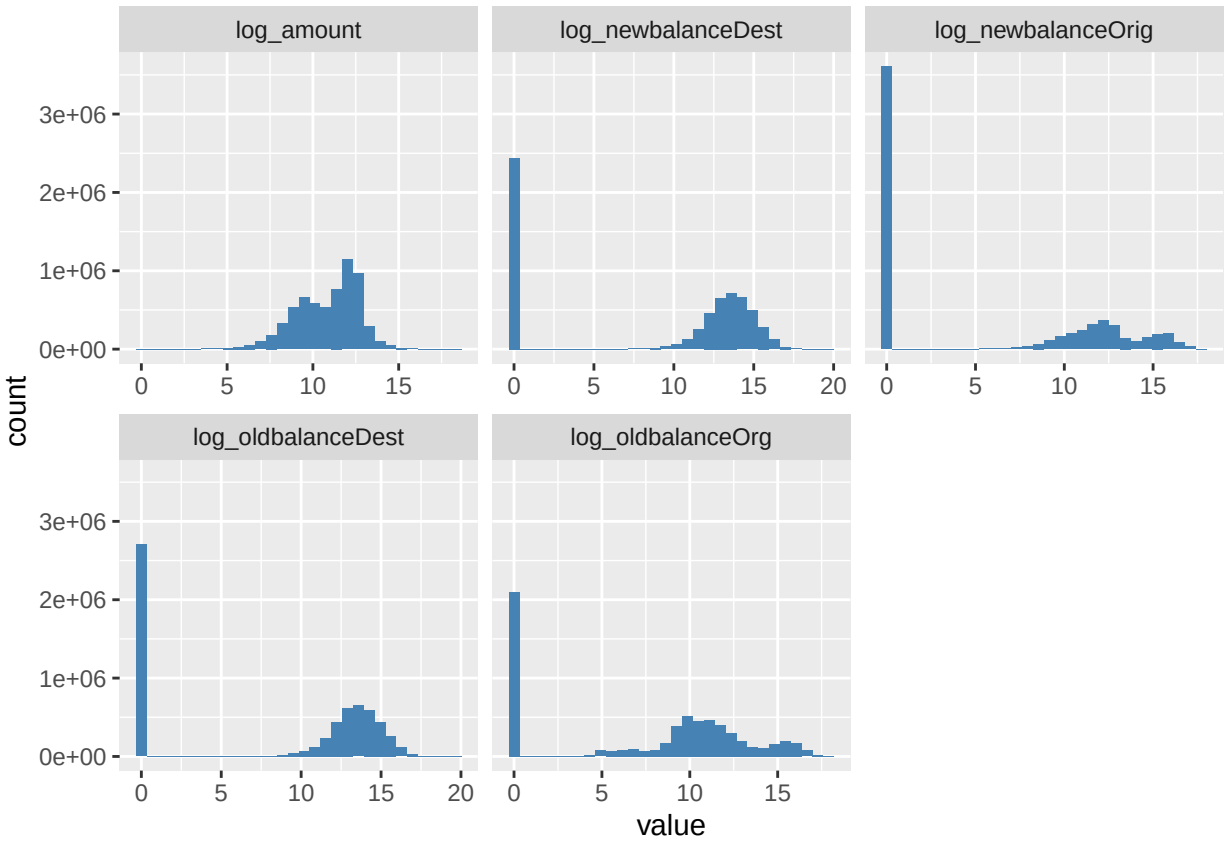
```
# Log transform  (amount, oldbalanceOrg, newbalanceOrig, oldbalanceDest, newbalanceDest)
cont_vars <- c('amount', 'oldbalanceOrg', 'newbalanceOrig', 'oldbalanceDest', 'newbalanceDest')

# add small constant to prevent inf values
log_scaled <- sapply(data.frame(reduced_df)[, cont_vars], function(x) log(x + 1))
colnames(log_scaled) <- lapply(cont_vars, function(x) paste('log_', x, sep=''))


reduced_df <- cbind(reduced_df, log_scaled)
```

Transformed Distributions

```
ggplot(gather(reduced_df[, 12:16]), aes(value)) +
  geom_histogram(bins = 30, fill = "steelblue") +
  facet_wrap(~key, scales = 'free_x')
```



One-Hot Encode Categorical Predictor

```
reduced_df$type <- as.factor(reduced_df$type)
dmy <- dummyVars(" ~ type", data = reduced_df, sep = '.', fullRank = TRUE)
reduced_df <- cbind(reduced_df, data.frame(predict(dmy, newdata = reduced_df)))
# drop type
reduced_df <- subset(reduced_df, select = -c(type))
```

Split Data into Training and Test Datasets using Stratified Random Sampling

```
set.seed(42)

# split x and y
x <- subset(reduced_df, select = -c(isFraud))
y <- reduced_df$isFraud

data_part <- createDataPartition(y = y, p = 0.75, list = FALSE)

x_train <- x[data_part, ]
y_train <- y[data_part]
x_test <- x[-data_part, ]
y_test <- y[-data_part]
```

```
summary(x_train)
```

```
##       step           amount         oldbalanceOrg      newbalanceOrig
##  Min.   :  1.0   Min.   :       0   Min.   :       0   Min.   :       0
##  1st Qu.:156.0   1st Qu.:   13386   1st Qu.:       0   1st Qu.:       0
##  Median :239.0   Median :   74779   Median :   14199   Median :       0
```

```
##  Mean   :243.4   Mean   :  179679   Mean   :  833985   Mean   :  855228
##  3rd Qu.:335.0   3rd Qu.:  208639   3rd Qu.:  107361   3rd Qu.:  144410
##  Max.   :743.0   Max.   :92445517   Max.   :59585040   Max.   :49585040
##  oldbalanceDest      newbalanceDest      hours_intraday      by_day
##  Min.   :        0   Min.   :        0   Min.   : 0.00   Min.   : 0.00
##  1st Qu.:        0   1st Qu.:        0   1st Qu.:12.00   1st Qu.: 6.00
##  Median :   132474   Median :   214502   Median :16.00   Median :10.00
##  Mean   :  1100397   Mean   :  1224666   Mean   :15.32   Mean   :10.26
##  3rd Qu.:   942400   3rd Qu.:  1111217   3rd Qu.:19.00   3rd Qu.:14.00
##  Max.   :356015889   Max.   :356179279   Max.   :23.00   Max.   :31.00
##   day_of_week      log_amount     log_oldbalanceOrg log_newbalanceOrig
##  Min.   :0.000   Min.   : 0.000   Min.   : 0.000    Min.   : 0.000
##  1st Qu.:1.000   1st Qu.: 9.502   1st Qu.: 0.000    1st Qu.: 0.000
##  Median :2.000   Median :11.222   Median : 9.561    Median : 0.000
##  Mean   :2.521   Mean   :10.840   Mean   : 7.415    Mean   : 5.367
##  3rd Qu.:4.000   3rd Qu.:12.248   3rd Qu.:11.584    3rd Qu.:11.880
##  Max.   :6.000   Max.   :18.342   Max.   :17.903    Max.   :17.719
##  log_oldbalanceDest log_newbalanceDest type.CASH_OUT      type.DEBIT
##  Min.   : 0.00      Min.   : 0.000     Min.   :0.0000   Min.   :0.000000
##  1st Qu.: 0.00      1st Qu.: 0.000     1st Qu.:0.0000   1st Qu.:0.000000
##  Median :11.79      Median :12.276     Median :0.0000   Median :0.000000
##  Mean   : 7.72      Mean   : 8.329     Mean   :0.3515   Mean   :0.006514
##  3rd Qu.:13.76      3rd Qu.:13.921     3rd Qu.:1.0000   3rd Qu.:0.000000
##  Max.   :19.69      Max.   :19.691     Max.   :1.0000   Max.   :1.000000
##   type.PAYMENT    type.TRANSFER
##  Min.   :0.0000   Min.   :0.00000
##  1st Qu.:0.0000   1st Qu.:0.00000
##  Median :0.0000   Median :0.00000
##  Mean   :0.3383   Mean   :0.08363
##  3rd Qu.:1.0000   3rd Qu.:0.00000
##  Max.   :1.0000   Max.   :1.00000
```