DAR F21 Project Status Notebook:

DeFi

Roman Vakhrushev (vakhrr)

10/28/2021

Contents

Biweekly Work Summary

- RCS ID: vakhrr
- Project Name: Blockchain DeFi
- These two weeks I was working on analyzing deficient liquidations.
- First week I started by analyzing the deficient liquidations graphically and making hypotheses
- Second week I applied logistic regression and some other classification algorithms to see what features are important.
- Branch: dar-vakhrr, uploaded files: vakhrr_assignment05.{Rmd,html,pdf}

Personal Contribution

All contributions were completed by me.

Discussion of Primary Findings

What did you want to know?

I wanted to study deficient liquidation (liquidations with collateral<principal) in more detail. Additionally, I wanted to see what differences are there between regular and deficient liquidations. Lastly, I tried to see what factors might be important for deficient liquidations.

How did you go about finding it?

I decided to analyze the data on deficient liquidations from various perspectives: time, amount of transactions, collateral-principal ratio, etc. I created several dataframes, tables, and graphs to show these patterns. Lastly, I built several classification models to study the importance of different factors for deficient liquidations and to try to predict them.

What did you find?

```r
#data collection as always
df<-read_rds('../../Data/transactions.Rds')
# Use deplyr to drop NA reserves, add the counts and then kep only the top 20
reservecoins <- df %>%  drop_na(reserve) %>%
count(reserve) %>%
```

```
arrange(-n) %>%
head(20)
```

```
#function to mark stable and non-stable coins
coinType <- function(coin) {
    #stable_coins <- list("USDC","USDT","DAI","BUSD","SUSD","GUSD","TUSD")
    if(str_contains(coin,"USD",ignore.case = TRUE))
    {
      result = "stable"
    }
    else if(str_contains(coin,"DAI",ignore.case = TRUE))
    {
      result = "stable"
    }
    else
    {
      result = "non-stable"
    }
    return(result)
}
```

Let's start by building a dataframe for deficient liquidations and computing the percentage of deficient liquidations over all liquidations.

```
#Show transactions, where collateral<principal (exclude WETH and AmmWETH for now).
dfst <- df %>% filter(type == "liquidation") %>% filter(collateralReserve != "WETH") %>% filter(principa

dfst$collateralType <- mapply(coinType, dfst$collateralReserve)
dfst$principalType <- mapply(coinType, dfst$principalReserve)

dfs <- df %>% filter(type == "liquidation")

#Count total liquidation
count(dfst)/count(dfs)
```

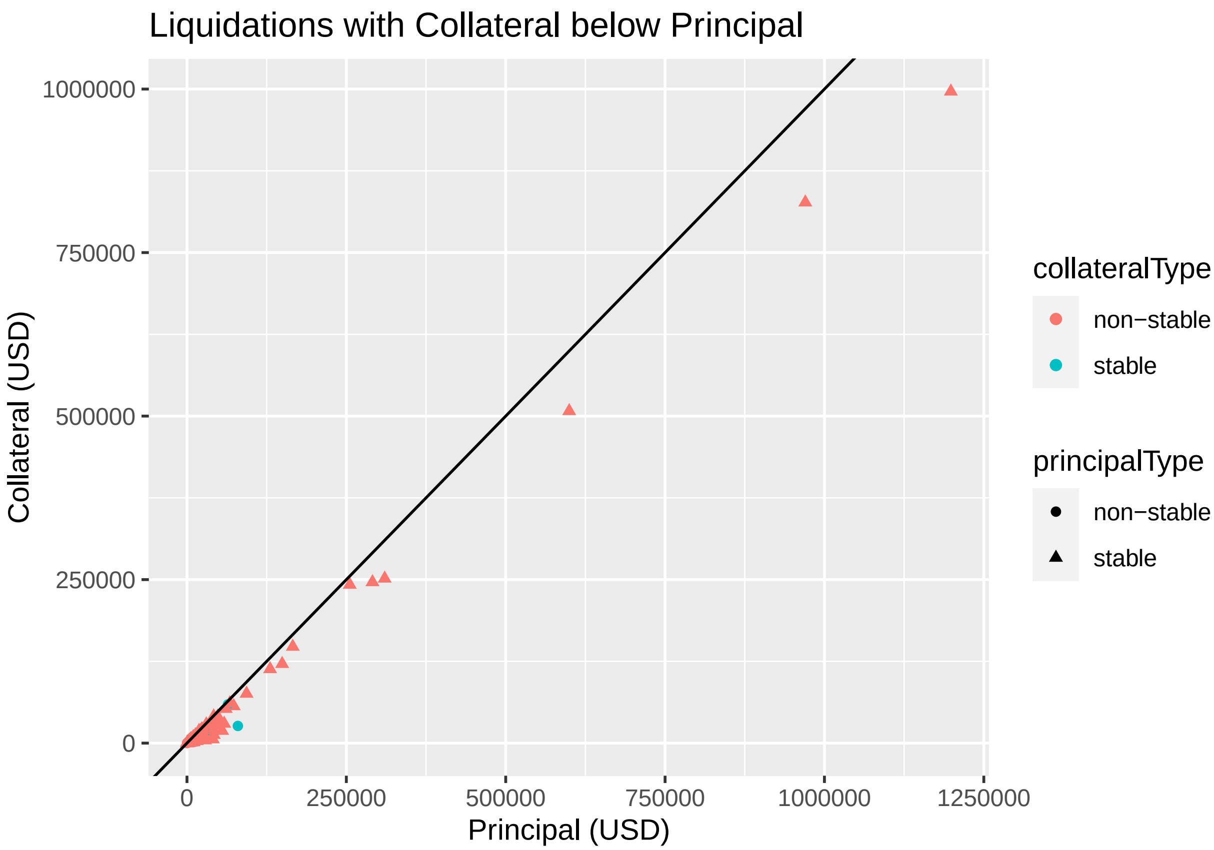
```
##            n
## 1 0.02337415
```

As we can see, there are about 2.3% of deficient liquidations over the whole data set, which is a relatively high number. This even excludes our problematic data on WETH and AmmWETH.

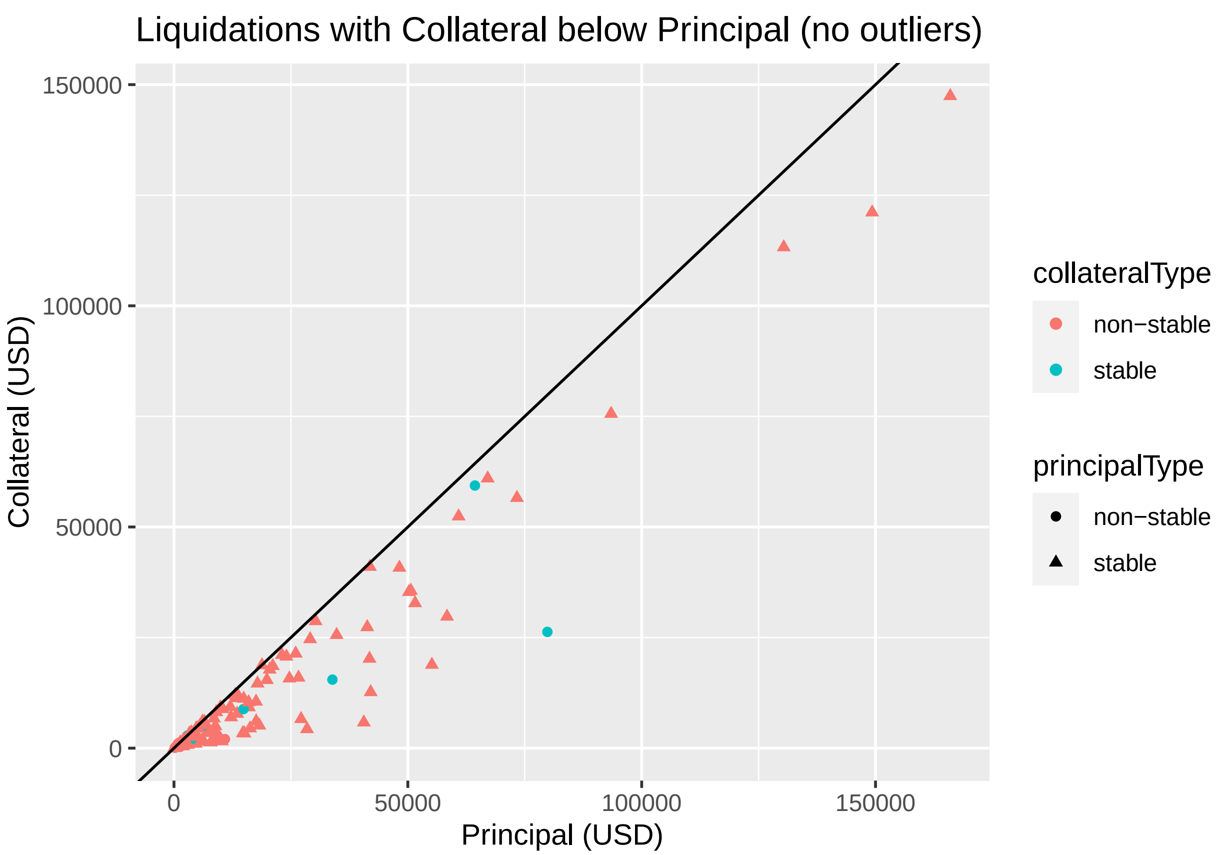
```
#Show just random 10 of those (exclude some data)
dfst %>% select(collateralReserve,principalReserve,amountUSDCollateral,amountUSDPincipal) %>% head(10)
```

```
##    collateralReserve principalReserve amountUSDCollateral amountUSDPincipal
## 1                ENJ             USDT          18862.4882        55144.5691
## 2      AmmBptBALWETH         AmmUSDC         996316.1988      1198318.9516
## 3                ZRX              DAI           3698.0659         8339.8518
## 4      AmmBptBALWETH          AmmDAI            148.1574          150.9005
## 5      AmmBptBALWETH          AmmDAI            462.7767          513.0650
## 6               WBTC             GUSD            894.7895         1350.9114
## 7                DAI              ENJ           2845.5268         2889.4629
## 8             XSUSHI              DAI          28718.2659        30281.5916
## 9                ENJ             USDC           2325.5813         9803.4118
## 10               KNC             BUSD            685.8761          984.6393
```

```r
#dfst %>% select(collateralReserve,principalReserve,amountUSDCollateral,amountUSDPincipal)[order(-dfst$

plot1 <- ggplot(dfst,aes(x = amountUSDPincipal, y = amountUSDCollateral,color = collateralType, shape =

plot1
```



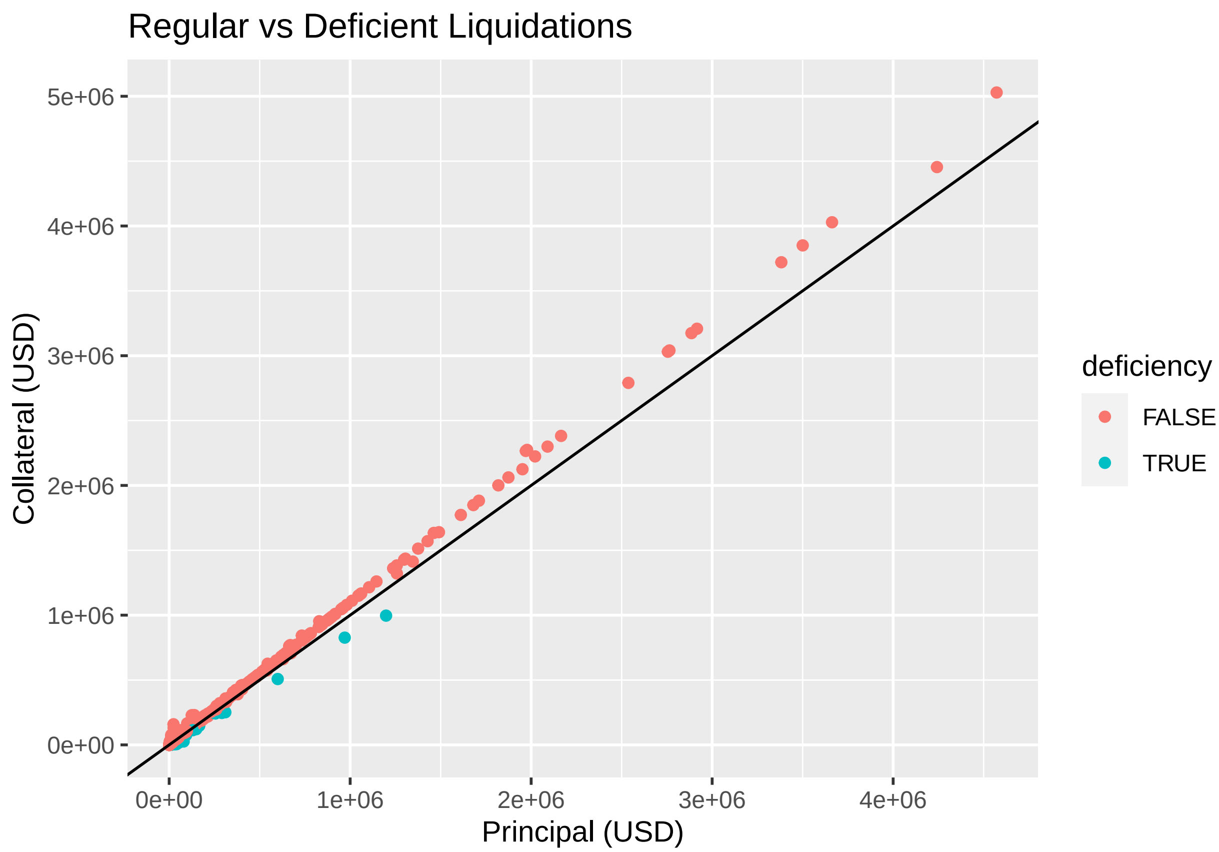

```r
dfst2 <- dfst %>% filter(amountUSDPincipal < 250000)

plot2 <- ggplot(dfst2,aes(x = amountUSDPincipal, y = amountUSDCollateral,color = collateralType, shape =

plot2
```



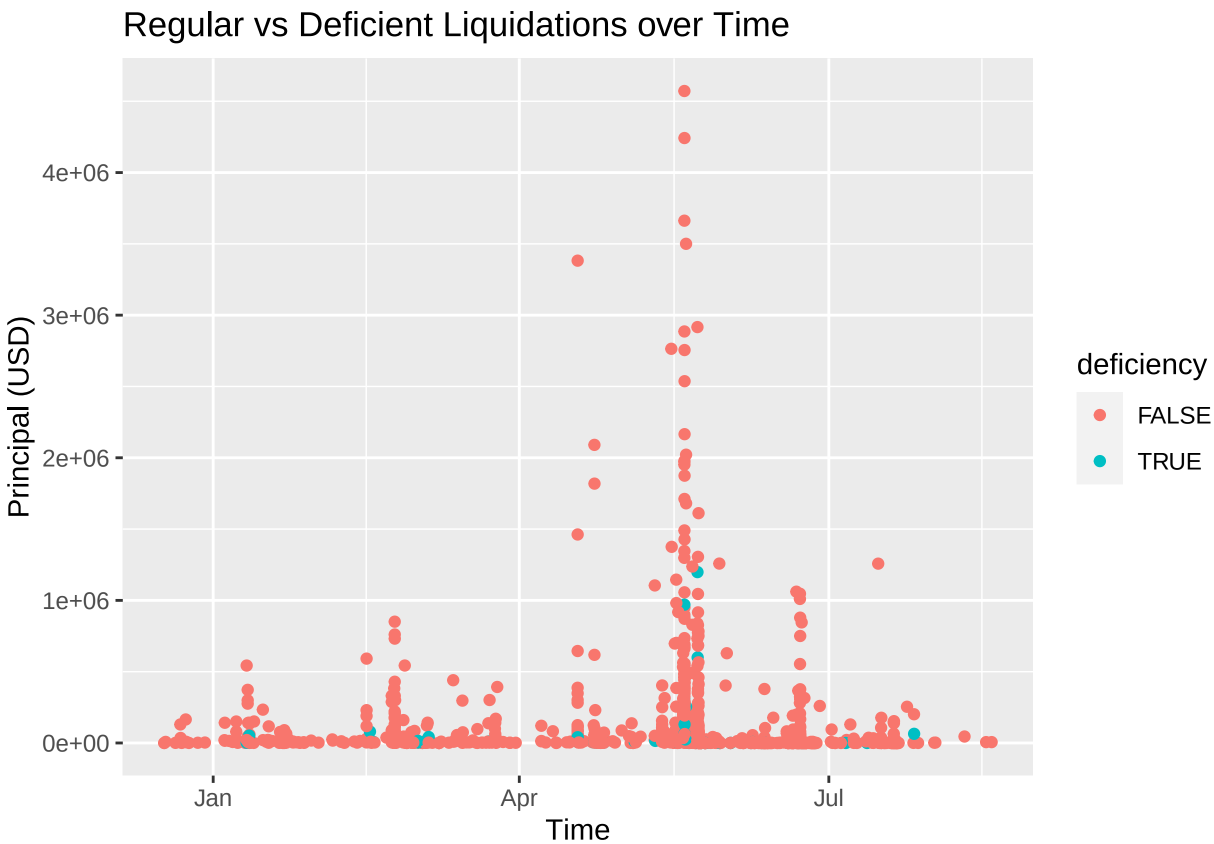
We can take a look into how deficient liquidations are distributed. First of all, there are just a few liquidations with very high collateral and principal value. Most of the deficient liquidations are below $100000 in principal. Second, we see that there are no deficient liquidations, where both collateral and principal are stable coins. This is probably just because there are very little (stable,stable) liquidations in general. Lastly, we observe that some complicated distribution in terms of distance from the identity line. There are some deficient liquidations that are extremely close to the identity line, but there are also a lot of deficient liquidations that are quite distant from it.

```
#function to mark deficient/regular liquidations
defLiquid <- function(principal, collateral) {
  if(collateral < principal)
  {
    result = TRUE
  }
  else
  {
    result = FALSE
  }
  return(result)
}

dfl <- df %>% filter(type == "liquidation") %>% filter(collateralReserve != "WETH") %>% filter(principa

dfl$deficiency <- mapply(defLiquid,dfl$amountUSDPincipal,dfl$amountUSDCollateral)

#plot of regular vs deficient liquidations

plot3 <- ggplot(dfl,aes(x = amountUSDPincipal, y = amountUSDCollateral,color = deficiency)) + geom_poin
```

```r
plot3
```

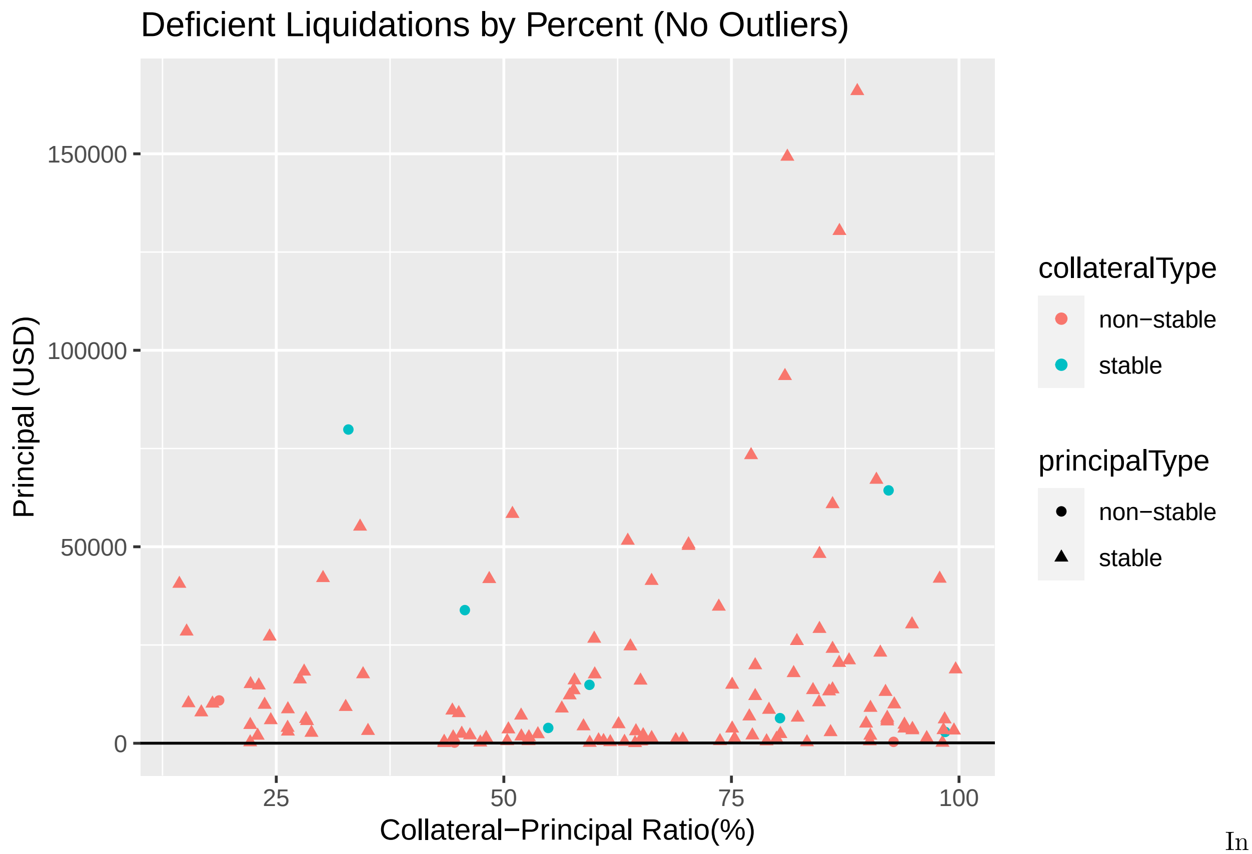



```r
#plot of liquidations over time

plot4 <- ggplot(dfl,aes(y = amountUSDPincipal,x = as_datetime(timestamp, tz = "UTC"),color = deficiency

plot4
```



We can take a look into deficient vs regular liquidations. Unsurprisingly, we see that deficient liquidations are only represented when the amount of transactions is very small compared to regular liquidations. If we look into liquidations over time, we can observe a few trends. The regular ggplot makes it harder to see (compared to ggplotly), but we can still see that all liquidations both deficient and regular are distributed non-equally (due to spikes). However, we still observe that deficient liquidations occurred in different times from January through July and August. Another interesting observation is that it seems like deficient liquidations often occur in pairs and triples (within the same day or two days), but I do not know if this is really true and how to explain it.

```
dfl <- dfl %>% filter(deficiency == TRUE) %>% mutate(percent = amountUSDCollateral*100/amountUSDPincipal

#dfl$percent <- format(dfl$percent,scientific = FALSE)


dfl$principalType <- mapply(coinType, dfl$principalReserve)
dfl$collateralType <- mapply(coinType, dfl$collateralReserve)

#plot this graph, excluding outliers, so it is easier to see
plot5 <- ggplot(dfl%>%filter(amountUSDPincipal < 250000),aes(x = percent,y = amountUSDPincipal, color =

plot5
```



In order to study the deficient liquidations in more detail, we can take a look into collateral-principal ratio. Collateral-principal ratio is defined as collateral(USD)/principal(USD). So, this ratio, expressed as percent, is always less than 100% for deficient liquidations. From the plot, we can observe that distribution (in horizontal axis) seems to be more or less uniform, at the very least there is no significant bias towards 100% as I would expect. One interesting detail to observe is that there is a gap in deficient liquidations between about 35% to 43% in collateral-principal ratio.

```
#We have to reload data for new analysis

#data collection as always
#df2<-read_rds('../../DefiResearch/transactions2.Rds')
# Use deplyr to drop NA reserves, add the counts and then kep only the top 20
reservecoins <- df %>%  drop_na(reserve) %>%
count(reserve) %>%
arrange(-n) %>%
head(20)
```

```
#Let's try logistic regression on data

#dfl <- df %>% filter(type == "liquidation")

dfll <- df %>% filter(type == "liquidation") %>% filter(collateralReserve != "WETH") %>% filter(princip

dfll$deficiency <- mapply(defLiquid,dfll$amountUSDPincipal,dfll$amountUSDCollateral)
dfll$principalType <- mapply(coinType, dfll$principalReserve)
dfll$collateralType <- mapply(coinType, dfll$collateralReserve)

dfll<-dfll %>% mutate(defNum = ifelse(deficiency == TRUE, 1, 0) )
```

```r
dfll<-dfll %>% mutate(princTypeNum = ifelse(principalType == "stable", 1, 0) )
dfll<-dfll %>% mutate(collatTypeNum = ifelse(collateralType == "stable", 1, 0) )



model <- glm(defNum ~ timestamp + collateralAmount + principalAmount + reservePriceETHPrincipal + reser
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
summary(model)
```

```
##
## Call:
## glm(formula = defNum ~ timestamp + collateralAmount + principalAmount +
##     reservePriceETHPrincipal + reservePriceETHCollateral, family = binomial,
##     data = dfll, maxit = 100)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.5486  -0.3256  -0.3038  -0.1679   3.1913
##
## Coefficients:
##                             Estimate Std. Error z value Pr(>|z|)
## (Intercept)                1.335e+02  3.640e+01   3.667 0.000246 ***
## timestamp                 -8.409e-08  2.245e-08  -3.745 0.000180 ***
## collateralAmount          -8.068e-07  2.190e-06  -0.368 0.712534
## principalAmount           -1.431e-07  6.102e-07  -0.234 0.814655
## reservePriceETHPrincipal  -3.263e-17  5.526e-17  -0.591 0.554834
## reservePriceETHCollateral -9.579e-20  2.058e-20  -4.654 3.26e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1216.1  on 3457  degrees of freedom
## Residual deviance: 1164.6  on 3452  degrees of freedom
## AIC: 1176.6
##
## Number of Fisher Scoring iterations: 16
```

```r
dfll<- dfll %>% mutate(priceRatio =  reservePriceETHCollateral/reservePriceETHPrincipal, amountRatio = 

model1 <- glm(defNum ~ priceRatio, data = dfll, family = binomial, maxit = 100)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
summary(model1)
```

```
##
## Call:
## glm(formula = defNum ~ priceRatio, family = binomial, data = dfll,
##     maxit = 100)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.3337  -0.3335  -0.3334  -0.1158   3.4013
```

```
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.860e+00  8.595e-02 -33.277  < 2e-16 ***
## priceRatio  -7.222e-05  1.625e-05  -4.444 8.84e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1216.1  on 3457  degrees of freedom
## Residual deviance: 1160.9  on 3456  degrees of freedom
## AIC: 1164.9
##
## Number of Fisher Scoring iterations: 17
```

```r
#model2 <- glm(defNum ~ priceRatio, data = dfll, family = binomial, maxit = 100)

model2 <- glm(defNum ~ collateralAmount + principalAmount + reservePriceETHPrincipal + reservePriceETHC
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
summary(model2)
```

```
##
## Call:
## glm(formula = defNum ~ collateralAmount + principalAmount + reservePriceETHPrincipal +
##     reservePriceETHCollateral + princTypeNum + collatTypeNum,
##     family = binomial, data = dfll, maxit = 100)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.6492  -0.3272  -0.3262  -0.1758   3.4606
##
## Coefficients:
##                             Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -1.430e+00  5.206e-01  -2.746  0.00603 **
## collateralAmount          -5.655e-08  1.525e-06  -0.037  0.97041
## principalAmount           -5.728e-08  5.747e-07  -0.100  0.92060
## reservePriceETHPrincipal  -6.245e-17  6.241e-17  -1.001  0.31699
## reservePriceETHCollateral -9.332e-20  2.132e-20  -4.376 1.21e-05 ***
## princTypeNum              -1.440e+00  5.106e-01  -2.821  0.00479 **
## collatTypeNum             -8.927e-01  5.354e-01  -1.667  0.09546 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1216.1  on 3457  degrees of freedom
## Residual deviance: 1170.4  on 3451  degrees of freedom
## AIC: 1184.4
##
## Number of Fisher Scoring iterations: 16
```

```r
model3 <- glm(defNum ~ timestamp + priceRatio + collateralAmount + principalAmount + reservePriceETHPri
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
summary(model3)
```

```
##
## Call:
## glm(formula = defNum ~ timestamp + priceRatio + collateralAmount +
##     principalAmount + reservePriceETHPrincipal + princTypeNum +
##     collatTypeNum, family = binomial, data = dfll, maxit = 100)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.7775  -0.3324  -0.3077  -0.1107   3.4401
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)               1.244e+02  3.818e+01   3.260  0.00112 **
## timestamp                -7.781e-08  2.361e-08  -3.296  0.00098 ***
## priceRatio               -7.655e-05  1.649e-05  -4.642 3.46e-06 ***
## collateralAmount         -5.054e-07  2.103e-06  -0.240  0.81007
## principalAmount           1.081e-07  6.481e-07   0.167  0.86748
## reservePriceETHPrincipal -8.395e-17  6.396e-17  -1.312  0.18937
## princTypeNum             -1.097e+00  5.395e-01  -2.032  0.04211 *
## collatTypeNum            -1.038e+00  5.610e-01  -1.851  0.06418 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1216.1  on 3457  degrees of freedom
## Residual deviance: 1140.3  on 3450  degrees of freedom
## AIC: 1156.3
##
## Number of Fisher Scoring iterations: 17
```

We can take a look into what factors are important for deficient liquidations. We can verify it by looking into different models using logistic regression. We additionally introduce new feature - price ratio (ratio of prices of principal and collateral coins). The results of trying different models (different features), but we probably want to look into the last model that includes most features. We obviously want to exclude percent (collateral-principal ratio) from the analysis, as it would reveal the way deficient liquidations are defined. As we can see, logistic regression proves the importance of time (which we already observed above) and price ratio (price ratio has extremely good p-value), we also see that principal and collateral type both have some statistical significance. Other factors seem to have little to no influence on the results.

```r
set.seed(2)

#data separation, done from r
ind <- sample(c(rep(TRUE,ceiling(nrow(dfll)*0.8)),rep(FALSE,floor(nrow(dfll)*0.2))))
data1 <- dfll[ind, ]
data2 <- dfll[!ind, ]

model4 <- glm(defNum ~ timestamp + priceRatio + collateralAmount + principalAmount + reservePriceETHPri
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
model4b <- rpart(defNum ~ timestamp + priceRatio + collateralAmount + principalAmount + reservePriceETHI

model4c <- randomForest(defNum ~ timestamp + priceRatio + collateralAmount + principalAmount + reservePr
```

```
## Warning in randomForest.default(m, y, ...): The response has five or fewer
## unique values. Are you sure you want to do regression?
```

```r
summary(model4)
```

```
##
## Call:
## glm(formula = defNum ~ timestamp + priceRatio + collateralAmount +
##     principalAmount + reservePriceETHPrincipal + princTypeNum +
##     collatTypeNum, family = binomial, data = data1, maxit = 100)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.7854  -0.3396  -0.3091  -0.1142   3.4216
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)               1.361e+02  4.198e+01   3.242  0.00119 **
## timestamp                -8.502e-08  2.596e-08  -3.275  0.00106 **
## priceRatio               -7.616e-05  1.793e-05  -4.247 2.17e-05 ***
## collateralAmount          3.079e-07  1.837e-06   0.168  0.86689
## principalAmount          -4.227e-07  9.510e-07  -0.444  0.65669
## reservePriceETHPrincipal -8.557e-17  7.077e-17  -1.209  0.22660
## princTypeNum             -9.936e-01  5.921e-01  -1.678  0.09334 .
## collatTypeNum            -1.193e+00  6.424e-01  -1.858  0.06323 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1000.19  on 2766  degrees of freedom
## Residual deviance:  936.06  on 2759  degrees of freedom
## AIC: 952.06
##
## Number of Fisher Scoring iterations: 16
```

```r
#summary(model4b)
summary(model4c)
```

```
##                 Length Class  Mode
## call               3   -none- call
## type               1   -none- character
## predicted       2767   -none- numeric
## mse              500   -none- numeric
## rsq              500   -none- numeric
## oob.times       2767   -none- numeric
## importance         7   -none- numeric
## importanceSD       0   -none- NULL
## localImportance    0   -none- NULL
## proximity          0   -none- NULL
## ntree              1   -none- numeric
```

```
## mtry               1   -none- numeric
## forest            11   -none- list
## coefs              0   -none- NULL
## y               2767   -none- numeric
## test               0   -none- NULL
## inbag              0   -none- NULL
## terms              3   terms  call
```

```r
#summary(model4b)

result <- predict(model4,data2,type = "response")

#resultb <- predict(model4b,data2)

print(head(result,10))
```

```
##           3           9          10          23          25          34
## 0.054326151 0.033064206 0.045063846 0.015570328 0.044631730 0.044120878
##          36          37          40          48
## 0.055758109 0.057477729 0.013927746 0.003447966
```

```r
data2<- data2 %>% mutate(predictedValue = predict(model4,data2,type = "response"))

data2<- data2 %>% mutate(predictedResult = ifelse(predictedValue>0.5,TRUE,FALSE))

data2<- data2 %>% mutate(predictedValue2 = predict(model4b,data2))

data2<- data2 %>% mutate(predictedResult2 = ifelse(predictedValue2>0.5,TRUE,FALSE))

data2<- data2 %>% mutate(predictedValue3 = predict(model4b,data2))

data2<- data2 %>% mutate(predictedResult3 = ifelse(predictedValue3>0.5,TRUE,FALSE))

head(data2 %>% select(deficiency, predictedValue, predictedResult, predictedValue2, predictedResult2, p
```

```
##    deficiency predictedValue predictedResult predictedValue2 predictedResult2
## 3       FALSE    0.054326151           FALSE     0.000000000            FALSE
## 9        TRUE    0.033064206           FALSE     0.687500000             TRUE
## 10      FALSE    0.045063846           FALSE     0.000000000            FALSE
## 23      FALSE    0.015570328           FALSE     0.000000000            FALSE
## 25      FALSE    0.044631730           FALSE     0.000000000            FALSE
## 34      FALSE    0.044120878           FALSE     0.000000000            FALSE
## 36      FALSE    0.055758109           FALSE     0.000000000            FALSE
## 37      FALSE    0.057477729           FALSE     0.000000000            FALSE
## 40      FALSE    0.013927746           FALSE     0.001477105            FALSE
## 48      FALSE    0.003447966           FALSE     0.001477105            FALSE
##    predictedValue3 predictedResult3
## 3      0.000000000            FALSE
## 9      0.687500000             TRUE
## 10     0.000000000            FALSE
## 23     0.000000000            FALSE
## 25     0.000000000            FALSE
## 34     0.000000000            FALSE
## 36     0.000000000            FALSE
## 37     0.000000000            FALSE
```

```
## 40     0.001477105             FALSE
## 48     0.001477105             FALSE
```

```r
#Logistic Regression Accuracy
count(data2 %>% filter(predictedResult == deficiency))/count(data2)
```

```
##           n
## 1 0.9638205
```

```r
#Regression Tree Accuracy
count(data2 %>% filter(predictedResult2 == deficiency))/count(data2)
```

```
##           n
## 1 0.9811867
```

```r
#Random Forest Accuracy
count(data2 %>% filter(predictedResult3 == deficiency))/count(data2)
```

```
##           n
## 1 0.9811867
```

In order to measure accuracy of the logistic regression model, we can try how well it predicts the data using those features. We introduce two more algorithms for classification: regression trees and random forest. We built all three models using the same features (as above). Additionally we separate data into testing and training sets. The resulting models and accuracy of each algorithm can be observed above (summary for regression trees is commented out as it is very lengthy). Due to very unbalanced nature of our data (98% vs 2%), logistic regression marks all liquidations as False, which still gives it high accuracy. The other two algorithms seem to both perform slightly better and they not always mark data points as False. In general, it is probably not the best idea to train classification models on such an unbalanced datasets – a good idea would be to balance data, but, unfortunately, we only have about 150 deficient observations, which is really small.