MATP-4910 Final Project Notebook Draft

DAR Project (One of: DeFi, Match2, Eat4Genes)

Jaimin Vyas

Due 3 November 2021

Contents

Final Project: Github Info

- github repository: https://github.rpi.edu/DataINCITE/IDEA-Blockchain
- Your github ID: vyasj2
- Final notebook: dar_final_vyasj2_3nov2021.Rmd
- Summary of github contributions including github issues addressed.
  - https://github.rpi.edu/DataINCITE/IDEA-Blockchain/blob/master/DefiResearch/StudentNotebooks/Assignment03/vyasj2_assignment03.pdf
  - https://github.rpi.edu/DataINCITE/IDEA-Blockchain/blob/master/DefiResearch/StudentNotebooks/Assignment04/vyasj2_assignment04.pdf'
  - https://github.rpi.edu/DataINCITE/IDEA-Blockchain/blob/master/DefiResearch/StudentNotebooks/Assignment05/vyasj2_assignment05.pdf
  - https://github.rpi.edu/DataINCITE/IDEA-Blockchain/blob/master/DefiResearch/StudentNotebooks/Assignment06/vyasj2_assignment06.pdf
  - https://github.rpi.edu/DataINCITE/IDEA-Blockchain/blob/master/DefiResearch/StudentNotebooks/analysis.Rmd
  - https://github.rpi.edu/DataINCITE/IDEA-Blockchain/blob/master/DefiResearch/StudentNotebooks/analysisv2.Rmd
  - Issues #82 and #98

Overview & Problems Tackled

In this notebook, I will walk through the borrow rate analysis, as well as survival rate analysis that I performed on AAVE transaction data. Borrow rate analysis was done using rolling average calculations for

borrow rates in AAVE, and visualizing how the rates shifted over time. Survival rate analysis was performed using Kaplan-Meier curves, along with some median survival rate calculations.

Data Description

The original transaction data that was given had more than 745,000 observations, with 34 features. It was queried from TheGraph API, a website that hosts crypto protocol data and regularly updates their databases. The data had date ranging almost 11 months, with the first observation being on November 30th, 2020, and the last observation being on October 17th, 2021.

To prepare the data for survival analysis, I had to split the original dataset into all of the "borrow" transactions, and all of the "liquidation" transactions. Then, I merged the data on user id, so that we would get a dataset that contained all of the borrow data and all of the liquidation data for each user. Then, I just took the time difference between the borrow date and the liquidation date, and that was the "age" of the loan. After performing survival analysis on this, and then using different strata, there were some interesting results.

Results

From the survival analysis, I found that for all borrows through the entire dataset, the probability of a loan surviving for more than 200 days is about 35% For specific cryptocurrencies, the behavior is a bit different. For the borrow rate analysis, I found that after China's "crypto ban" in April of 2020, stable and variable borrow rates went from extremely unpredictable and variable to a screeching halt, becoming more or less constant, or fluctuating around a solid average as opposed to oscillating between very high and very low.

Problem 1

We will start with the borrow rate analysis. Borrow rate refers to the percent interest that a person taking out a loan has to pay on the principal amount to the lender of the loan. These rates fluctuate based mainly on how much of a specific type of currency is available in the lending pool, which is a central pool of money made up of different cryptocurrencies that users can borrow from/deposit to. The problem that I was aiming to solve here was to understand how the borrow rates fluctuated aside from amount available in the lending pool, and what kinds of factors could be used as predictors.

Methods

I did some fairly basic analysis, initially just plotting the borrow rates on the y-axis with the timestamp that borrow rate was observed on the x-axis, and just visualizing the trends. I noticed that the initial fluctuation of borrow rate seemed to slim past a certain date, so I plotted a 21-day rolling average of the borrow rates, and split based on the two modes, stable and variable. I picked stablecoins for this analysis (stablecoins are cryptocurrencies whose value is pegged to a real world item, in this case the coins' values are pegged to the US Dollar) because they were the most borrowed cryptocurrency in the dataset by far, accounting for more than 80% of all borrows.

Results

After plotting, the result was very easy to see; after a certain date in late May of 2021, both stable and variable borrow rates screeched to a halt and stopped fluctuating, and it took them a bit to start moving again, but they did not fluctuate as nearly as heavily as they did pre-April. After doing some digging, we found that China, arguably the biggest player in the cryptocurrency field, announced they would be outlawing all cryptocurrency transactions sometime during that date of May 2021. We don't know why it had this much of an impact on borrow rates, but it goes to show how volatile the cryptocurrency market is with respect to world events. The graphs can be found below:

```r
if (!require("readr")) {
  install.packages("readr")
  library(readr)
}
```

```
## Loading required package: readr
```

```r
if (!require("dplyr")) {
  install.packages("dplyr")
  library(dplyr)
}
```

```
## Loading required package: dplyr
```

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

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

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

```r
if (!require("ggplot2")) {
  install.packages("ggplot2")
  library(ggplot2)
}
```

```
## Loading required package: ggplot2
```

```r
if (!require("zoo")) {
  install.packages("zoo")
  library(zoo)
}
```

```
## Loading required package: zoo
```

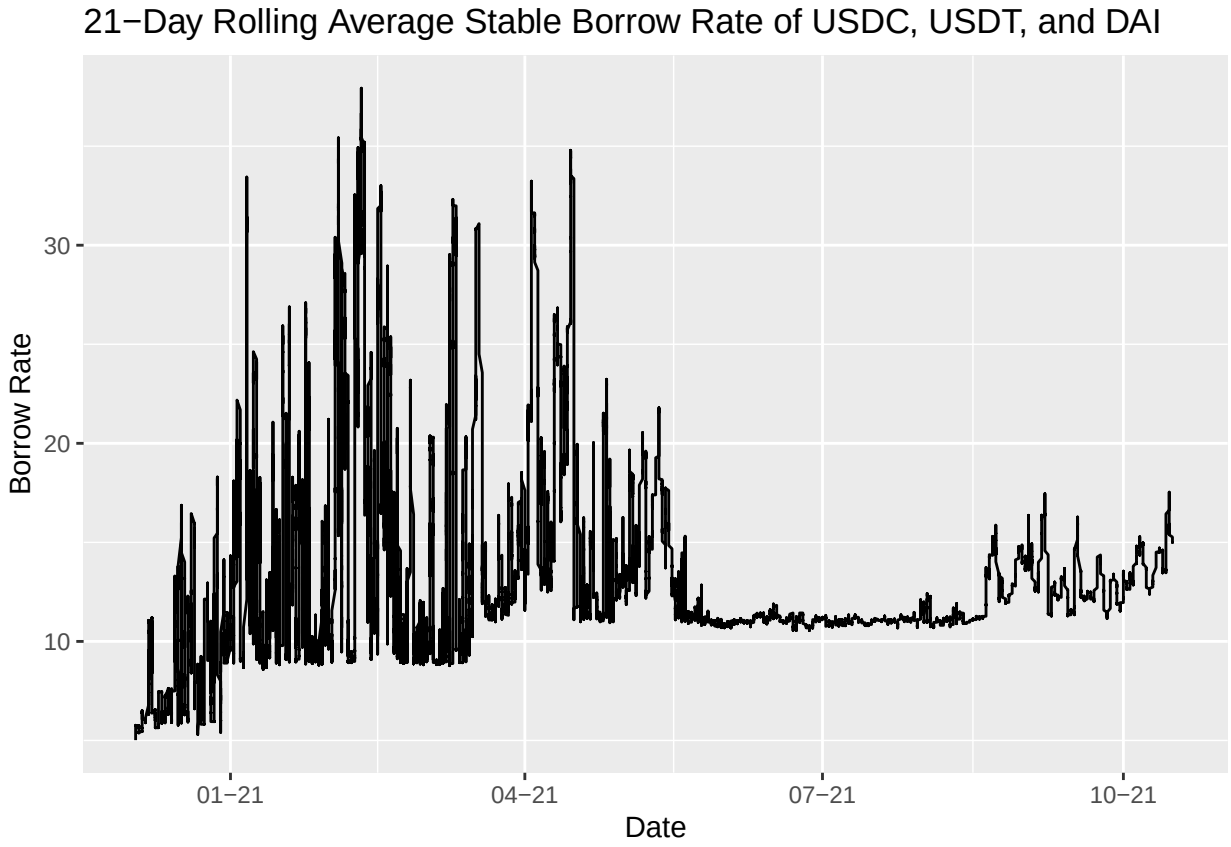
```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
raw_df <- read_rds("../../Data/transactionsv2.rds")

borrows <- raw_df %>%
  dplyr::filter(type=="borrow")

stable.borrows <- borrows %>%
  dplyr::filter(borrowRateMode=="Stable") %>%
  dplyr::filter(reserve=="USDC" | reserve=="USDT" | reserve=="DAI") %>%
  dplyr::arrange(timestamp) %>%
  dplyr::mutate(rollingAvg21=zoo::rollmean(borrowRate,k=21,fill=NA))

variable.borrows <- borrows %>%
  dplyr::filter(borrowRateMode=="Variable") %>%
```

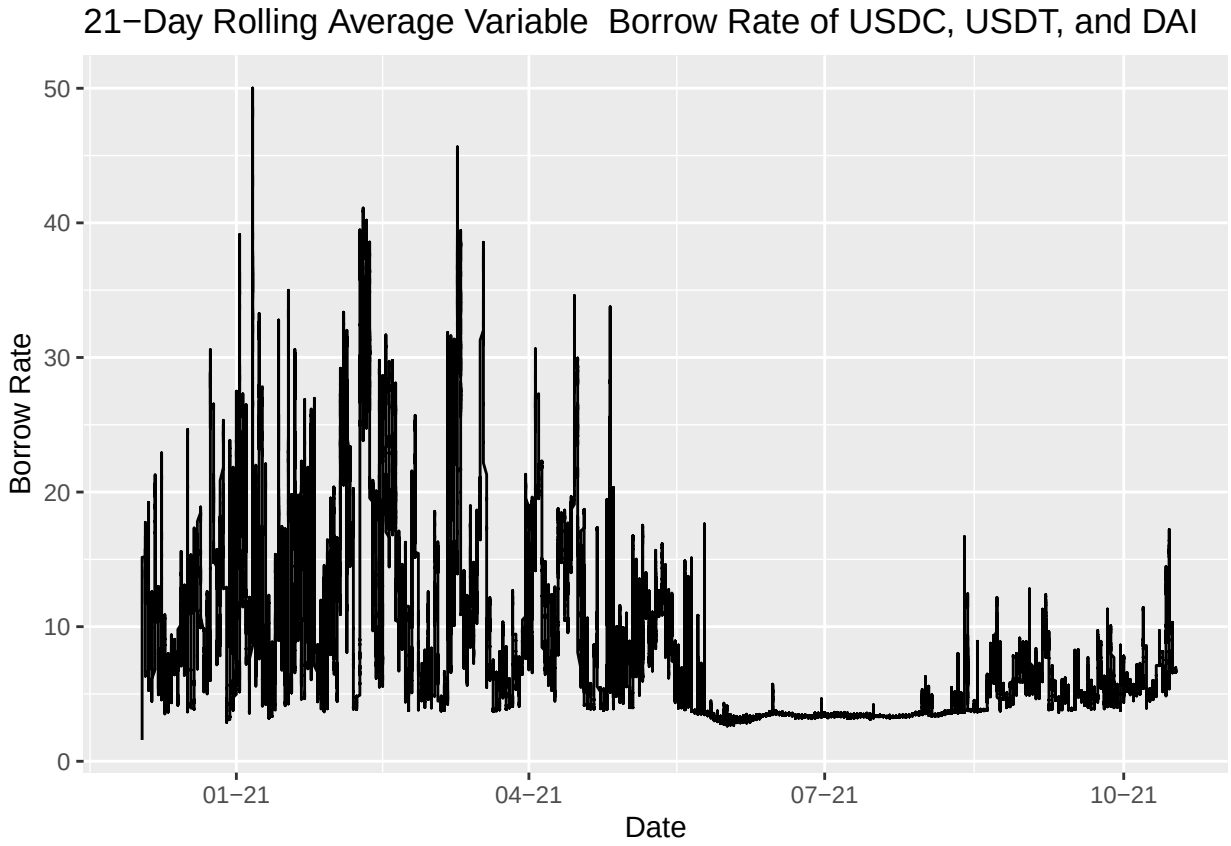
```
  dplyr::filter(reserve=="USDC" | reserve=="USDT" | reserve=="DAI") %>%
  dplyr::arrange(timestamp) %>%
  dplyr::mutate(rollingAvg21=zoo::rollmean(borrowRate,k=21,fill=NA))

ggplot(stable.borrows, aes(x=as.Date(datetime),y=rollingAvg21)) +
  geom_line() +
  scale_x_date(date_labels="%m-%y") +
  xlab("Date") +
  ylab("Borrow Rate") +
  ggtitle("21-Day Rolling Average Stable Borrow Rate of USDC, USDT, and DAI")
```

```
## Warning: Removed 20 row(s) containing missing values (geom_path).
```




```
ggplot(variable.borrows, aes(x=as.Date(datetime),y=rollingAvg21)) +
  geom_line() +
  scale_x_date(date_labels="%m-%y") +
  xlab("Date") +
  ylab("Borrow Rate") +
  ggtitle("21-Day Rolling Average Variable  Borrow Rate of USDC, USDT, and DAI")
```

```
## Warning: Removed 20 row(s) containing missing values (geom_path).
```



Discussion

Interpret results. What were your findings? What do they say about in terms of the original domain (e.g. DeFi or health eating)? What are the strengths and limitations of these results? Is there support for your findings from other sources? Include references as appropriate.

The findings of this analysis were fairly interesting, depicting how the borrow rate for stablecoins is very heavily impacted by world events. This shouldn't be the case, since borrow rate is a calculated value inside the AAVE protocol as a function of how much money is in the lending pool. These results go to show that cryptocurrency markets, though almost entirely defined by mathematical functions and the interactions of computers on the blockchain network, is still heavily dependent on sentiment.

However, this still only applies for stablecoins. For cryptocurrencies whose value is not pegged to another real world fiat currency, the impact of China's crypto ban was not as noticeable. Perhaps there is some special attribute of fiat currency-pegged cryptocurrencies that makes them more sensitive to real world market events.

Problem 2

Moving on to survival modeling and analysis, this idea came to life when as a group we were starting to tackle liquidation analysis. Liquidations are the cryptocurrency version of defaulting on a loan. Essentially, when someone lends you money, you would provide the lender some amount of collateral. However, because cryptocurrency prices are so volatile, you need to overcollateralize your loan, meaning you have to put in more USD worth of collateral than you actually initially borrowed. When the collateral's total value drops below a certain fraction of the principal amount borrowed, your loan gets liquidated, and you have to pay a very hefty fine as well as re-depositing collateral to overcollateralize your loan again.

Survival modeling is a statistical method of analyzing the time it takes for an event to occur from a given starting event. A probability curve is then fitted on the data to represent the model's prediction on what

the chances of the second event occurring are at a given time. For the transaction data that was given, I performed these kinds of analyses using the Kaplan-Meier model on the time it takes for a user to borrow, and then liquidate on that borrow.

Methods

For this problem, sorting the data and obtaining the right variables was a large task. The main two variables that Kaplan-Meier curves use are the "age", which is the time difference between the first event and the target event we are predicting, and whether or not a specific observation actually "died" (in this case, "liquidated"), or not. For this, I initially sorted the data on borrows and liquidations, and then performed a left join from borrows to liquidations, so that the final dataset had all borrows that both did and did not liquidate, and I then classified whether the user liquidated or not based on whether or not there was a liquidation timestamp given after the merge. The "age" of the loan was simply the difference in liquidation timestamp and borrow timestamp, and if there was no liquidation timestamp, it would just use the maximum borrow timestamp in the entire dataset, basically saying that the loan is still active today, or the user repaid their loan in full and did not liquidate.

Results

The results were interesting enough, because sorting by borrow rate and collateral type yielded some different graphs that provided more insight on how the conditions of the borrow (borrow rate mode, principal reserve, collateral reserve) can impact the survival of a loan. The results can be seen below:

```
if (!require("survival")) {
  install.packages("survival")
  library(survival)
}
```

```
## Loading required package: survival
```

```
if (!require("survminer")) {
  install.packages("survminer")
  library(survminer)
}
```

```
## Loading required package: survminer
```

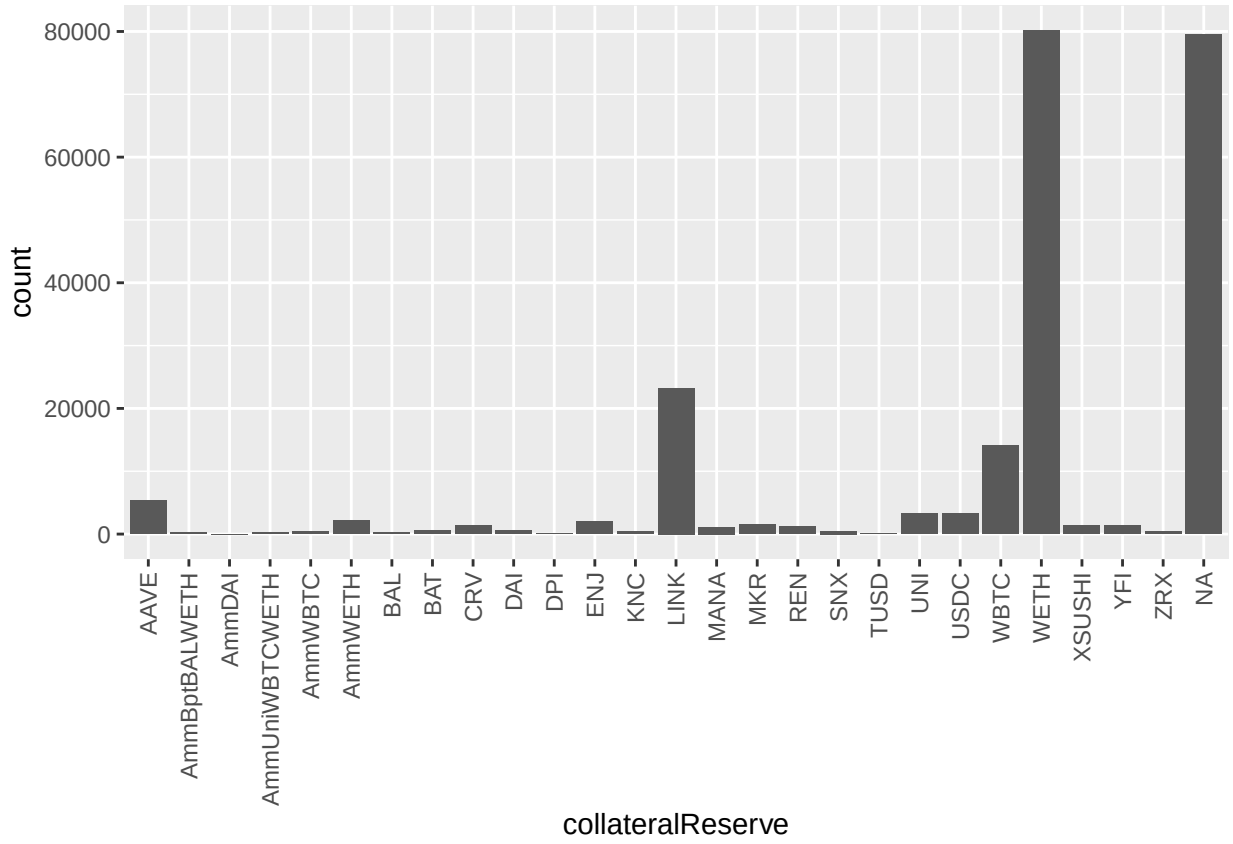
```
## Loading required package: ggpubr
```

```
##
## Attaching package: 'survminer'
```

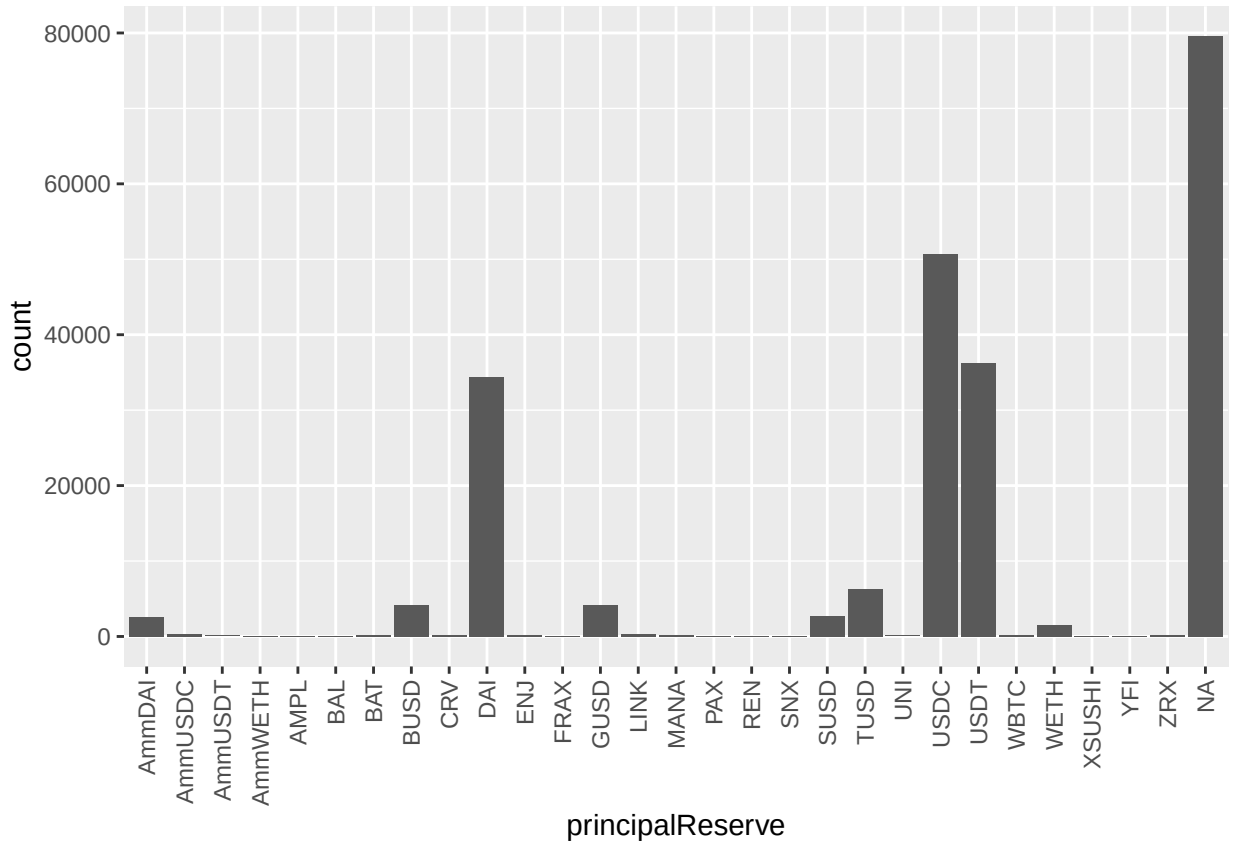
```
## The following object is masked from 'package:survival':
##
##     myeloma
```

```
borrows <- raw_df %>%
  dplyr::filter(type=="borrow") %>%
  dplyr::rename(name=onBehalfOf_alias)

liquidations <- raw_df %>%
  dplyr::filter(type=="liquidation") %>%
  dplyr::rename(name=user_alias)

allSurvivalData <- left_join(borrows,liquidations,by="name") %>%
  dplyr::mutate(age=case_when(is.na(timestamp.y) ~ max(borrows$timestamp)/86400,
                              TRUE ~ (timestamp.y-timestamp.x)/86400)) %>%
  dplyr::mutate(status=case_when(age==max(borrows$timestamp)/86400 ~ 0,
```

```
                                  TRUE ~ 1)) %>%
  dplyr::filter(age>=0) %>%
  dplyr::rename(principalReserve=principalReserve.y) %>%
  dplyr::rename(collateralReserve=collateralReserve.y) %>%
  dplyr::rename(borrowRateMode=borrowRateMode.x) %>%
  dplyr::select(name,age,status,principalReserve,collateralReserve,borrowRateMode)

ggplot(allSurvivalData, aes(x=collateralReserve)) +
  geom_bar(stat="count") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))
```



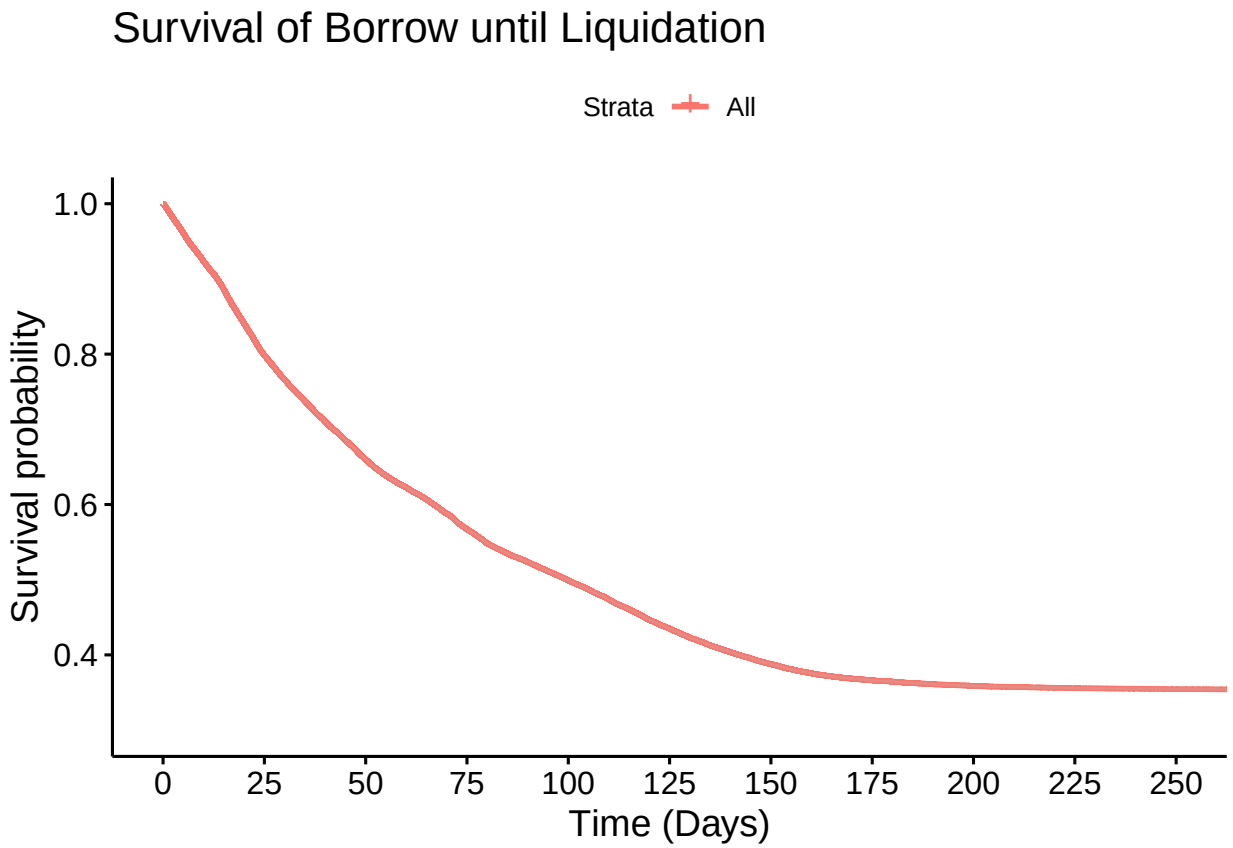

```
ggplot(allSurvivalData, aes(x=principalReserve)) +
  geom_bar(stat="count") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))
```




```
km_fit_all <- survfit(Surv(age, status) ~ 1, data=allSurvivalData)
summary(km_fit_all, times = c(1,30,60,90*(1:10)))
```

```
## Call: survfit(formula = Surv(age, status) ~ 1, data = allSurvivalData)
##
##  time n.risk n.event survival  std.err lower 95% CI upper 95% CI
##     1 223097    1594    0.993 0.000177        0.993        0.993
##    30 172140   50957    0.766 0.000893        0.764        0.768
##    60 139892   32248    0.623 0.001023        0.621        0.625
##    90 117620   22272    0.523 0.001054        0.521        0.526
##   180  81888   35732    0.364 0.001015        0.362        0.366
##   270  79573    2315    0.354 0.001009        0.352        0.356
##   360  79561      12    0.354 0.001009        0.352        0.356
##   450  79561       0    0.354 0.001009        0.352        0.356
##   540  79561       0    0.354 0.001009        0.352        0.356
##   630  79561       0    0.354 0.001009        0.352        0.356
##   720  79561       0    0.354 0.001009        0.352        0.356
##   810  79561       0    0.354 0.001009        0.352        0.356
##   900  79561       0    0.354 0.001009        0.352        0.356
```

```
ggsurvplot(km_fit_all,conf.int=TRUE,xlim=c(0,250),ylim=c(0.3,1),break.time.by=25) +
  xlab("Time (Days)") +
  ggtitle("Survival of Borrow until Liquidation")
```

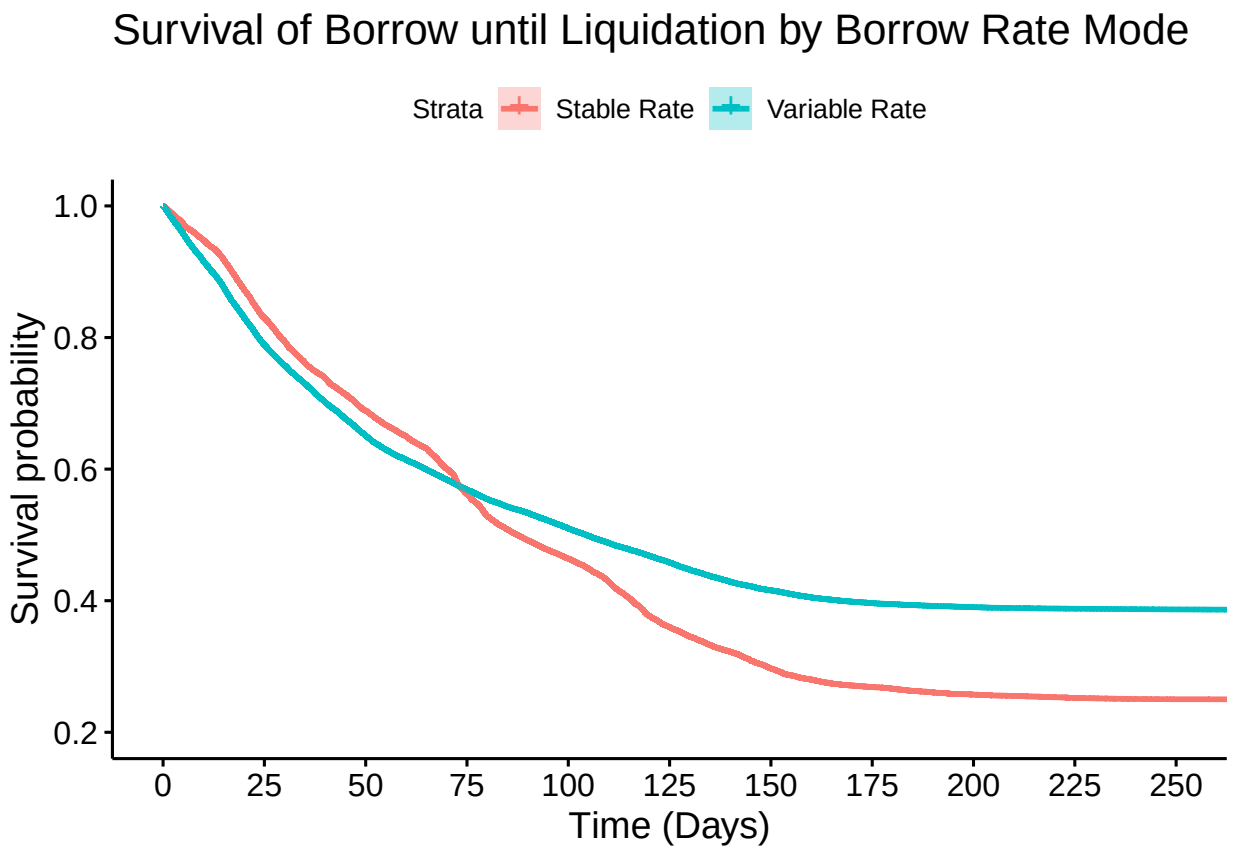



```
km_fit_brm_all <- survfit(Surv(age, status) ~ borrowRateMode, data=allSurvivalData)
summary(km_fit_brm_all, times = c(1,30,60,90*(1:10)))
```

```
## Call: survfit(formula = Surv(age, status) ~ borrowRateMode, data = allSurvivalData)
##
##                 borrowRateMode=Stable
##  time n.risk n.event survival  std.err lower 95% CI upper 95% CI
##     1  52783     266    0.995 0.000307        0.994        0.996
##    30  42124   10659    0.794 0.001756        0.791        0.798
##    60  34468    7656    0.650 0.002071        0.646        0.654
##    90  26086    8382    0.492 0.002171        0.487        0.496
##   180  14125   11961    0.266 0.001919        0.263        0.270
##   270  13268     857    0.250 0.001880        0.246        0.254
##   360  13266       2    0.250 0.001880        0.246        0.254
##   450  13266       0    0.250 0.001880        0.246        0.254
##   540  13266       0    0.250 0.001880        0.246        0.254
##   630  13266       0    0.250 0.001880        0.246        0.254
##   720  13266       0    0.250 0.001880        0.246        0.254
##   810  13266       0    0.250 0.001880        0.246        0.254
##   900  13266       0    0.250 0.001880        0.246        0.254
##
##                 borrowRateMode=Variable
##  time n.risk n.event survival  std.err lower 95% CI upper 95% CI
##     1 170314    1328    0.992 0.000211        0.992        0.993
##    30 130016   40298    0.757 0.001035        0.755        0.760
##    60 105424   24592    0.614 0.001175        0.612        0.617
##    90  91534   13890    0.533 0.001204        0.531        0.536
```

```
##   180  67763   23771     0.395 0.001180        0.392        0.397
##   270  66305    1458     0.386 0.001175        0.384        0.389
##   360  66295      10     0.386 0.001175        0.384        0.389
##   450  66295       0     0.386 0.001175        0.384        0.389
##   540  66295       0     0.386 0.001175        0.384        0.389
##   630  66295       0     0.386 0.001175        0.384        0.389
##   720  66295       0     0.386 0.001175        0.384        0.389
##   810  66295       0     0.386 0.001175        0.384        0.389
##   900  66295       0     0.386 0.001175        0.384        0.389
```

```
ggsurvplot(km_fit_brm_all,conf.int=TRUE,xlim=c(0,250),ylim=c(0.2,1),break.time.by=25,legend.labs=c("Stal
  xlab("Time (Days)") +
  ggtitle("Survival of Borrow until Liquidation by Borrow Rate Mode")
```

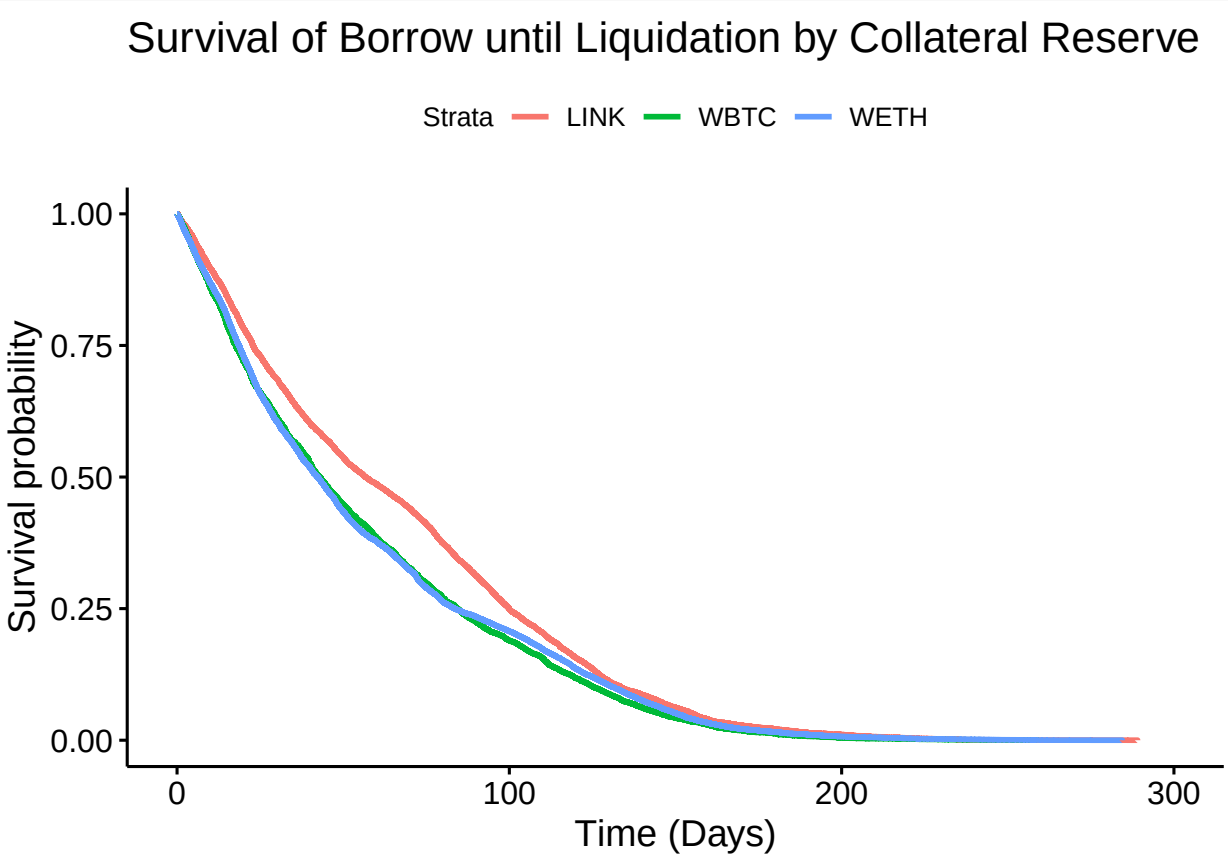



```
km_fit_cr_all <- survfit(Surv(age, status) ~ collateralReserve, data=allSurvivalData %>%
                           dplyr::filter(collateralReserve %in% c("LINK","WBTC","WETH")))
summary(km_fit_cr_all, times = c(1,30,60,90*(1:10)))
```

```
## Call: survfit(formula = Surv(age, status) ~ collateralReserve, data = allSurvivalData %>%
##     dplyr::filter(collateralReserve %in% c("LINK", "WBTC", "WETH")))
##
##                 collateralReserve=LINK
##  time n.risk n.event survival  std.err lower 95% CI upper 95% CI
##     1  23027     225 0.990323 0.000642     9.89e-01     0.991582
##    30  15971    7056 0.686866 0.003041     6.81e-01     0.692853
##    60  11314    4657 0.486582 0.003278     4.80e-01     0.493049
##    90   7272    4042 0.312747 0.003040     3.07e-01     0.318763
##   180    492    6780 0.021159 0.000944     1.94e-02     0.023093
```

```
##   270      4     488 0.000172 0.000086     6.46e-05     0.000458
##
##                collateralReserve=WBTC
##  time n.risk n.event survival  std.err lower 95% CI upper 95% CI
##     1  13879     159   0.9887 0.000893       0.9869       0.9904
##    30   8616    5263   0.6138 0.004109       0.6058       0.6219
##    60   5409    3207   0.3853 0.004108       0.3773       0.3934
##    90   3146    2263   0.2241 0.003519       0.2173       0.2311
##   180    176    2970   0.0125 0.000939       0.0108       0.0145
##
##                collateralReserve=WETH
##  time n.risk n.event survival  std.err lower 95% CI upper 95% CI
##     1  79090    1005 9.87e-01 3.93e-04     0.986682      0.98822
##    30  48516   30574 6.06e-01 1.73e-03     0.602356      0.60912
##    60  30243   18273 3.78e-01 1.71e-03     0.374247      0.38096
##    90  18797   11446 2.35e-01 1.50e-03     0.231767      0.23764
##   180   1257   17540 1.57e-02 4.39e-04     0.014856      0.01658
##   270      5    1252 6.24e-05 2.79e-05     0.000026      0.00015
```

```
ggsurvplot(km_fit_cr_all,legend.labs=c("LINK","WBTC","WETH")) +
  xlab("Time (Days)") +
  ggtitle("Survival of Borrow until Liquidation by Collateral Reserve")
```

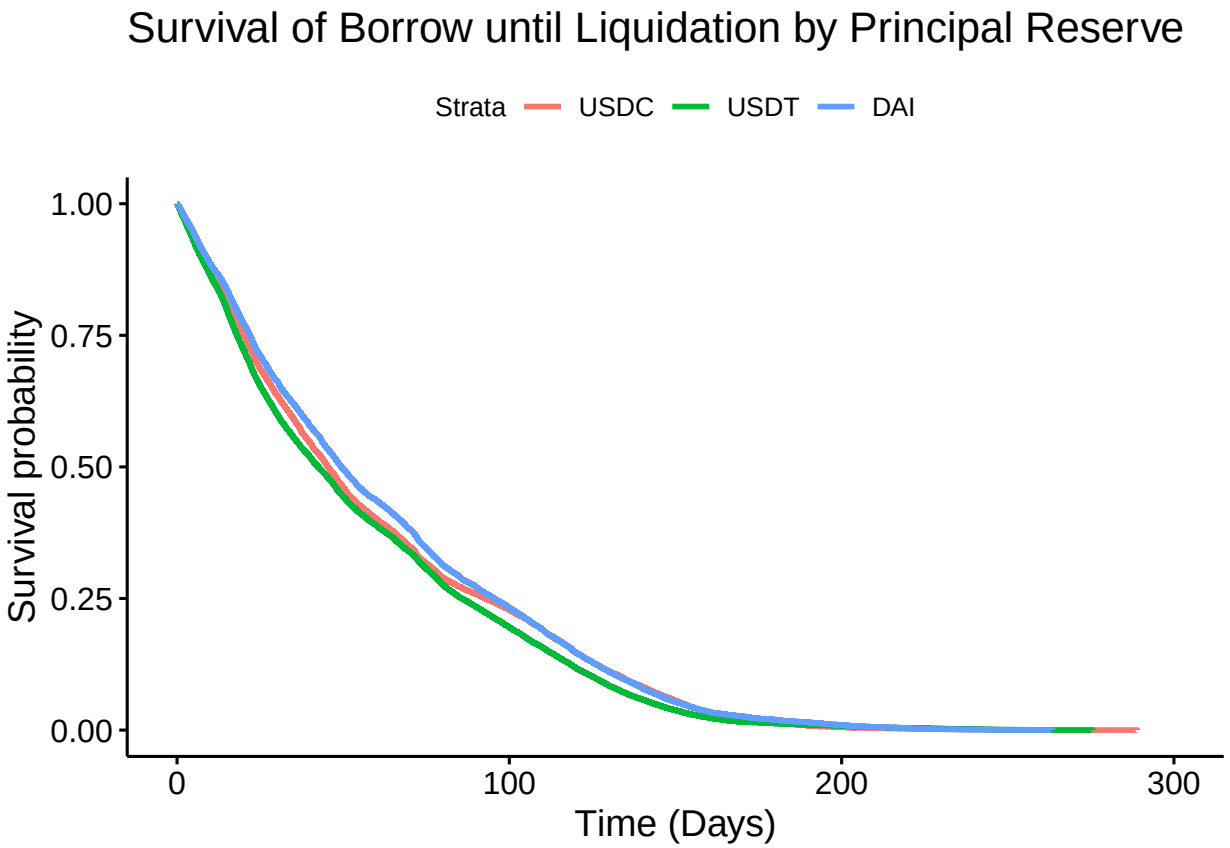



```
km_fit_pr_all <- survfit(Surv(age, status) ~ principalReserve, data=allSurvivalData %>%
                           dplyr::filter(principalReserve %in% c("USDC","USDT","DAI")))
summary(km_fit_pr_all, times = c(1,30,60,90*(1:10)))
```

```
## Call: survfit(formula = Surv(age, status) ~ principalReserve, data = allSurvivalData %>%
```

```
##     dplyr::filter(principalReserve %in% c("USDC", "USDT", "DAI")))
##
##                 principalReserve=DAI
##  time n.risk n.event survival  std.err lower 95% CI upper 95% CI
##     1  34044     390 0.988674 5.70e-04     9.88e-01     0.989792
##    30  21948   12096 0.637393 2.59e-03     6.32e-01     0.642491
##    60  13810    8138 0.401057 2.64e-03     3.96e-01     0.406267
##    90   8907    4903 0.258669 2.36e-03     2.54e-01     0.263336
##   180    528    8379 0.015334 6.62e-04     1.41e-02     0.016688
##   270      5     523 0.000145 6.49e-05     6.04e-05     0.000349
##
##                 principalReserve=USDC
##  time n.risk n.event survival  std.err lower 95% CI upper 95% CI
##     1  50052     643 9.87e-01 4.97e-04     9.86e-01     0.988291
##    30  30471   19581 6.01e-01 2.17e-03     5.97e-01     0.605343
##    60  19694   10777 3.88e-01 2.16e-03     3.84e-01     0.392746
##    90  11874    7820 2.34e-01 1.88e-03     2.31e-01     0.237940
##   180    642   11232 1.27e-02 4.97e-04     1.17e-02     0.013676
##   270      2     640 3.95e-05 2.79e-05     9.87e-06     0.000158
##
##                 principalReserve=USDT
##  time n.risk n.event survival  std.err lower 95% CI upper 95% CI
##     1  35909     338   0.9907 0.000505       0.9897       0.9917
##    30  24053   11856   0.6636 0.002482       0.6587       0.6685
##    60  15822    8231   0.4365 0.002605       0.4314       0.4416
##    90   9834    5988   0.2713 0.002335       0.2668       0.2759
##   180    717    9117   0.0198 0.000731       0.0184       0.0213
```

```
ggsurvplot(km_fit_pr_all,legend.labs=c("USDC","USDT","DAI")) +
  xlab("Time (Days)") +
  ggtitle("Survival of Borrow until Liquidation by Principal Reserve")
```



Discussion

I chose to do survival analysis on the coins that I did because as can be seen from the bar graphs, they are the most used coins for collateral reserve and principal reserve. From the graphs, we can see that less than 40% of loans survived after 200 days. However, when we split this on borrow rate modes, we see that variable borrow rate loans tend to survive more in the long run than stable borrow rate loans. In the short run however, we see the opposite behavior. This leads to the conclusion that after about 2-3 months, variable borrow rate loans have a higher chance of not liquidating than stable borrow rate loans. This is interesting, it could be because of user behavior, or the fact that the coins that users borrow using stable rates are more volatile than those they borrow using variable rates.

Summary and Recommendations

Decentralized finance is a fascinating up and coming field, and there are a plethora of research opportunities, many of which could have huge impacts on user behavior and could create a convention for how users perform transactions.

For future studies, I think analysis on borrow rates and different factors impacting liquidation would prove very useful. Looking more into other aspects of cryptocurrency markets, such as mining, and the new concept of NFTs, and how those fluctuate and might impact other factors would also be interesting.

I think a more in-depth version of my analysis could be included in a paper or an expanded app, because I think I would need to make the analysis more specific, since it's quite general right now and might not be able to help anyone specifically. For example, in an app, for users wanting to borrow a certain coin, they could input the coin they're borrowing and the amount, as well as the coin they deposited as collateral along with that amount, and the app could tell them their loan's predicted survival rate for the next week based on previous data, just as I've done here, as well as using price data for the collateral coin to predict what the

price would be over the next week.