

DAR Defi Team Assignment 2 (Fall 2021)

DeFi Reserve Coins

your name here

08/03/2021

GITHUB UPDATE & SUBMISSION INSTRUCTIONS (Delete before submission!)

For this and subsequent assignments you'll follow the same basic github steps you used for Assignment 1, ie:

- **Option 1:** Update the copy of your team's repository currently in your home directory:
 - *This is the preferred option if you have already cloned the repository*
 - In the Linux terminal: `git pull origin master`
 - If there are errors, it's simplest to proceed to "Option 2"
 - If no errors, check your branch: `git branch` (should be something like `dar-rcsid`)
- **Option 2:** Get a fresh copy of your team's repository:
 - *Do this only if Option 1 fails!*
 - In Linux: `cd ~` to get to your login directory
 - In Linux: `rm -Rf IDEA-Blockchain` (deletes all your previous, uncommitted work)
 - In Linux: `git clone https://github.rpi.edu/DataINCITE/IDEA-Blockchain.git`
 - In Linux: `cd IDEA-Blockchain`
 - In Linux: Recreate a personal branch to save your changes to: `git checkout -b dar-rcsid` (replace "rcsid" with your rcsid!)
- Locate and save a personal copy of the assignment notebook:
 - In RStudio "Files" tab, navigate to: Home > IDEA-Blockchain > DefiResearch > StudentNotebooks > Assignment02
 - Double-click on `blockchain-assignment2-f21.Rmd` to open
 - Under the "File" menu, "Save as" something like `rcsid-assignment2-f21.Rmd` (replace "rcsid" with your rcsid!)
 - You should see your file appear in the "Files" tab, usually at the bottom
 - Under "More" in the "Files" tab, select "Set as working directory"
- Edit your notebook, saving as you make changes.
 - `ctrl-S` works!
- "Knit" your notebook, generating a PDF file:
 - You should see your new PDF appear in the "Files" tab, usually at the bottom
 - "Export" your file (under the "More" menu in the "Files" tab) and save to your local file system
- Add your changed file(s) to your personal branch, and commit:
 - In Linux: `cd ~/IDEA-Blockchain/DefiResearch/StudentNotebooks/Assignment02`
 - In Linux: `git add rcsid-assignment2.*` (replace "rcsid" with your rcsid!)
 - In Linux: `git commit -m "rcsid assignment 2"`
- Push your changes to the remote repository:
 - In Linux: `git push origin dar-rcsid` (replace "rcsid" with your rcsid)
- Issue a pull request:
 - Navigate to <https://github.rpi.edu/DataINCITE/IDEA-Blockchain> in your browser
 - It should notify you of your recent push and prompt you to make a pull request.

- If not, find your branch under the popup (defaults to **master**), select and click to issue a pull request
- One of the instructors
- Upload your downloaded rcsid-assignment2-f21.pdf to LMS (replace “rcsid” with your rcsid)
- **Be prepared to share your findings in the DeFi Meeting Thursday.**

Prepare Transaction Data and Explore

We begin by loading our prepared AAVE transaction data into a dataframe. The dataset has over 400,000 rows, and 27 columns.

We are directly loading the dataframe from an Rds archive instead of a CSV file to conserve space.

```
#load Rds (binary version of csv file) into dataframe
# Assumes this notebook is in: ~/IDEA-Blockchain/DefiResearch/StudentNotebooks/Assignment02
df<-read_rds('../Data/transactions.Rds')

# Let's take a quick look at the first few observation
head(df)
```

```
##      amount borrowRate borrowRateMode  onBehalfOf      pool reserve
## 1  41501.63   6.274937      Variable 8.502518e+47 1.034668e+48    DAI
## 2 7000000.00   2.589628      Variable 4.635974e+47 1.034668e+48    USDT
## 3   15000.00   8.802541      Variable 3.735263e+47 1.034668e+48    USDC
## 4    8193.19  48.747052      Stable 6.896232e+47 1.034668e+48    USDC
## 5   11000.00   3.225055      Variable 1.089455e+48 1.034668e+48    USDT
## 6   40000.00   5.739208      Variable 2.178337e+47 1.034668e+48    USDT
##      timestamp      user   type reservePriceETH reservePriceUSD amountUSD
## 1 1621340435 8.502518e+47 borrow   2.852900e+14      0.9948044   41286.00
## 2 1622477822 4.635974e+47 borrow   3.812835e+14      1.0000000 7000000.00
## 3 1619775984 3.735263e+47 borrow   3.611000e+14      1.0043389   15065.08
## 4 1615481632 6.896232e+47 borrow   5.562201e+14      0.9993909    8188.20
## 5 1626914745 1.089455e+48 borrow   4.971100e+14      1.0000000   11000.00
## 6 1620936688 2.178337e+47 borrow   2.725248e+14      1.0000000   40000.00
##      collateralAmount collateralReserve principalAmount principalReserve
## 1                  NA                  NA
## 2                  NA                  NA
## 3                  NA                  NA
## 4                  NA                  NA
## 5                  NA                  NA
## 6                  NA                  NA
##      reservePriceETHPrincipal reservePriceUSDPrincipal reservePriceETHCollateral
## 1                  NA                  NA                  NA
## 2                  NA                  NA                  NA
## 3                  NA                  NA                  NA
## 4                  NA                  NA                  NA
## 5                  NA                  NA                  NA
## 6                  NA                  NA                  NA
##      reservePriceUSDCollateral amountUSDPincipal amountUSDCollateral
## 1                  NA                  NA                  NA
## 2                  NA                  NA                  NA
## 3                  NA                  NA                  NA
## 4                  NA                  NA                  NA
## 5                  NA                  NA                  NA
## 6                  NA                  NA                  NA
```

```
## borrowRateModeFrom borrowRateModeTo stableBorrowRate variableBorrowRate
## 1 NA NA
## 2 NA NA
## 3 NA NA
## 4 NA NA
## 5 NA NA
## 6 NA NA
```

Now look at the summaries to see the types, values, and missingness (NA's) of the data.

```
summary(df)
```

```
## amount borrowRate borrowRateMode onBehalfOf
## Min. : 0 Min. : 0.0 :386542 Min. :2.578e+33
## 1st Qu.: 24 1st Qu.: 3.3 Stable : 18408 1st Qu.:4.174e+47
## Median : 1427 Median : 3.9 Variable: 76569 Median :7.522e+47
## Mean : 191103 Mean : 9.5 Mean :7.592e+47
## 3rd Qu.: 24382 3rd Qu.: 10.8 3rd Qu.:1.168e+48
## Max. :600000000 Max. :10002.0 Max. :1.461e+48
## NA's :7289 NA's :386542 NA's :7289
## pool reserve timestamp user
## Min. :9.862e+47 USDC :105937 Min. :1.607e+09 Min. :2.578e+33
## 1st Qu.:1.035e+48 WETH :105279 1st Qu.:1.615e+09 1st Qu.:4.199e+47
## Median :1.035e+48 USDT : 58266 Median :1.621e+09 Median :8.697e+47
## Mean :1.034e+48 DAI : 55211 Mean :1.620e+09 Mean :8.082e+47
## 3rd Qu.:1.035e+48 LINK : 26404 3rd Qu.:1.624e+09 3rd Qu.:1.173e+48
## Max. :1.035e+48 WBTC : 26344 Max. :1.629e+09 Max. :1.461e+48
## (Other):104078
## type reservePriceETH reservePriceUSD
## borrow : 94977 Min. :1.000e+00 Min. :0.000e+00
## deposit :192006 1st Qu.:2.865e+14 1st Qu.:1.000e+00
## liquidation: 6289 Median :4.652e+14 Median :1.000e+00
## redeem :126705 Mean :3.458e+23 Mean :6.774e+08
## repay : 60542 3rd Qu.:9.411e+14 3rd Qu.:1.000e+00
## swap : 1000 Max. :1.647e+28 Max. :4.252e+13
## NA's :7289 NA's :7289
## amountUSD collateralAmount collateralReserve principalAmount
## Min. : 0 Min. : 0 :475230 Min. : 0
## 1st Qu.: 70 1st Qu.: 1 WETH : 2665 1st Qu.: 962
## Median : 5836 Median : 14 LINK : 1312 Median : 4362
## Mean : 245851 Mean : 5451 WBTC : 686 Mean : 66005
## 3rd Qu.: 49871 3rd Qu.: 250 AAVE : 333 3rd Qu.: 21533
## Max. :754379487 Max. :4638724 UNI : 230 Max. :4475668
## NA's :7289 NA's :475230 (Other): 1063 NA's :475230
## principalReserve reservePriceETHPrincipal reservePriceUSDPrincipal
## :475230 Min. :1.000e+00 Min. : 0.0
## USDC : 2142 1st Qu.:4.062e+14 1st Qu.: 1.0
## USDT : 1549 Median :4.682e+14 Median : 1.0
## DAI : 1459 Mean :1.556e+17 Mean : 295.6
## GUSD : 242 3rd Qu.:5.363e+14 3rd Qu.: 1.0
## TUSD : 175 Max. :4.203e+19 Max. :83819.1
## (Other): 722 NA's :475230 NA's :475230
## reservePriceETHCollateral reservePriceUSDCollateral amountUSDPrincipal
## Min. :1.000e+00 Min. :0.000e+00 Min. : 0
## 1st Qu.:1.000e+00 1st Qu.:0.000e+00 1st Qu.: 1022
```

```
## Median :5.110e+14      Median :1.000e+00      Median : 4481
## Mean   :2.177e+21      Mean   :4.543e+06      Mean   : 67361
## 3rd Qu.:1.110e+16      3rd Qu.:2.600e+01      3rd Qu.: 22066
## Max.   :9.116e+23      Max.   :2.509e+09      Max.   :4571839
## NA's   :475230        NA's   :475230        NA's   :475230
## amountUSDCollateral borrowRateModeFrom borrowRateModeTo stableBorrowRate
## Min.    : 0           :480519           :480519 Min.    : 0.0
## 1st Qu.: 0           Stable : 471      Stable : 529 1st Qu.: 9.0
## Median : 476        Variable: 529      Variable: 471 Median : 10.9
## Mean    : 37060                                     Mean    : 11.7
## 3rd Qu.: 7457                                           3rd Qu.: 12.0
## Max.    :5029023                                         Max.    :154.7
## NA's    :475230                                         NA's    :480519
## variableBorrowRate
## Min.    : 0.0
## 1st Qu.: 3.8
## Median : 3.9
## Mean    : 5.7
## 3rd Qu.: 5.1
## Max.    :148.7
## NA's    :480519
```

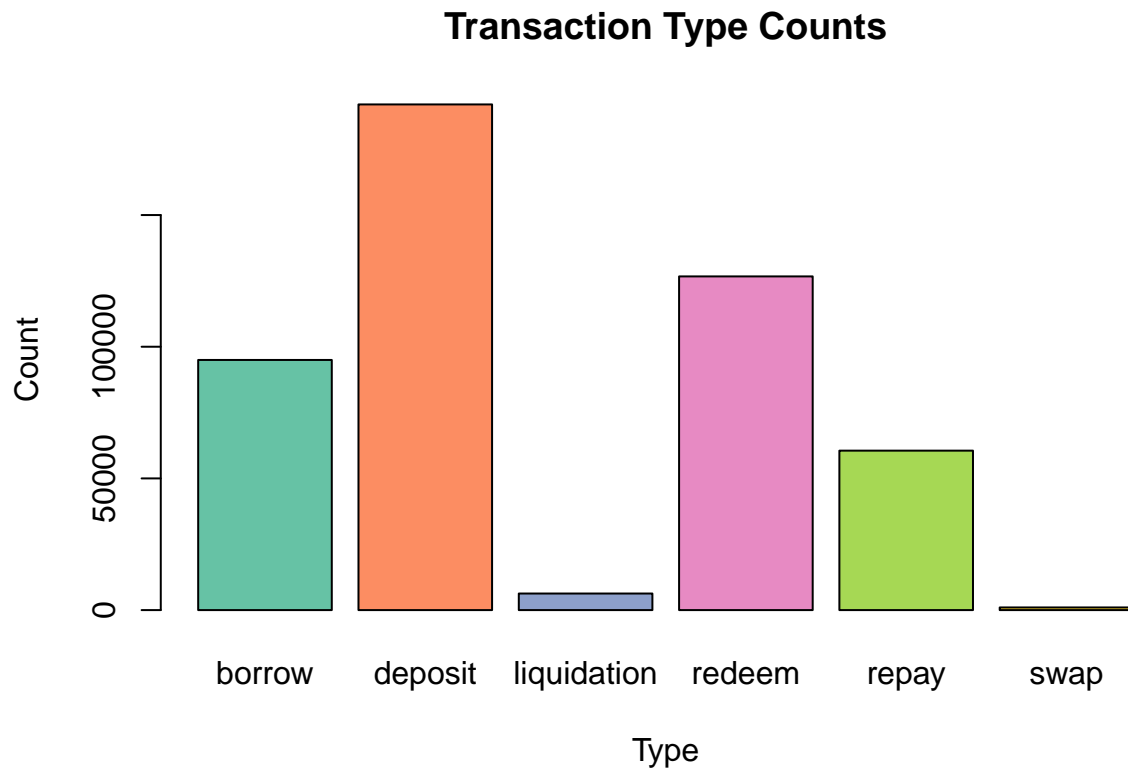
First we'll do some preliminary analysis before we ask detailed questions.

Analyze Transaction Types

Let's examine the different types of *transactions* present in the data. We'll make a simple bar plot to visualize the number of each transaction types. "Deposit" is the most common type of transaction, whereas "swaps" are the most rare.

```
#set color palette
colors = brewer.pal(6,"Set2")

#create barplot
barplot(table(df$type), main='Transaction Type Counts', xlab='Type',ylab='Count',col=colors)
```



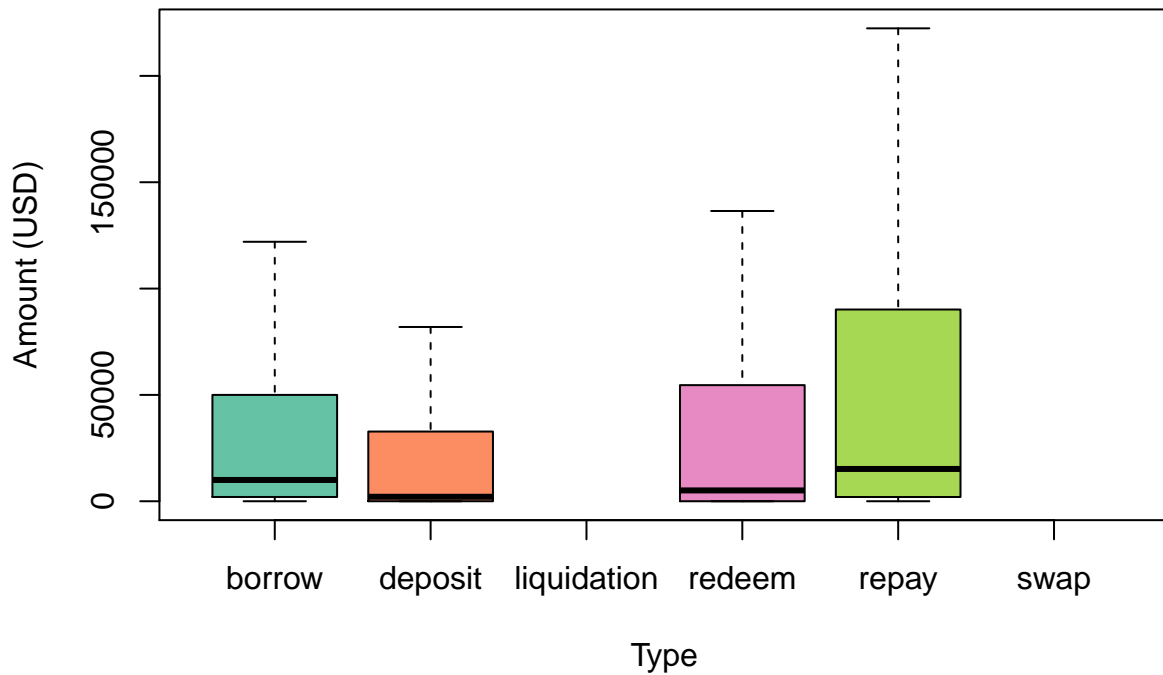
There are more “deposits” than “borrows,” because users often need to overcollateralize for loans.

Now we’ll examine the amount of US dollars being used in the different types of transactions. We create box plots for the four types of transactions that have the “amount” feature associated with them, and we visualize the distribution of that column for the different transactions.

We can see that most transactions are completed with very little money.

```
#create boxplot  
boxplot(amountUSD~type,data=df,outline=FALSE,col=colors,  
        main="Transaction Amounts",xlab="Type",ylab="Amount (USD)")
```

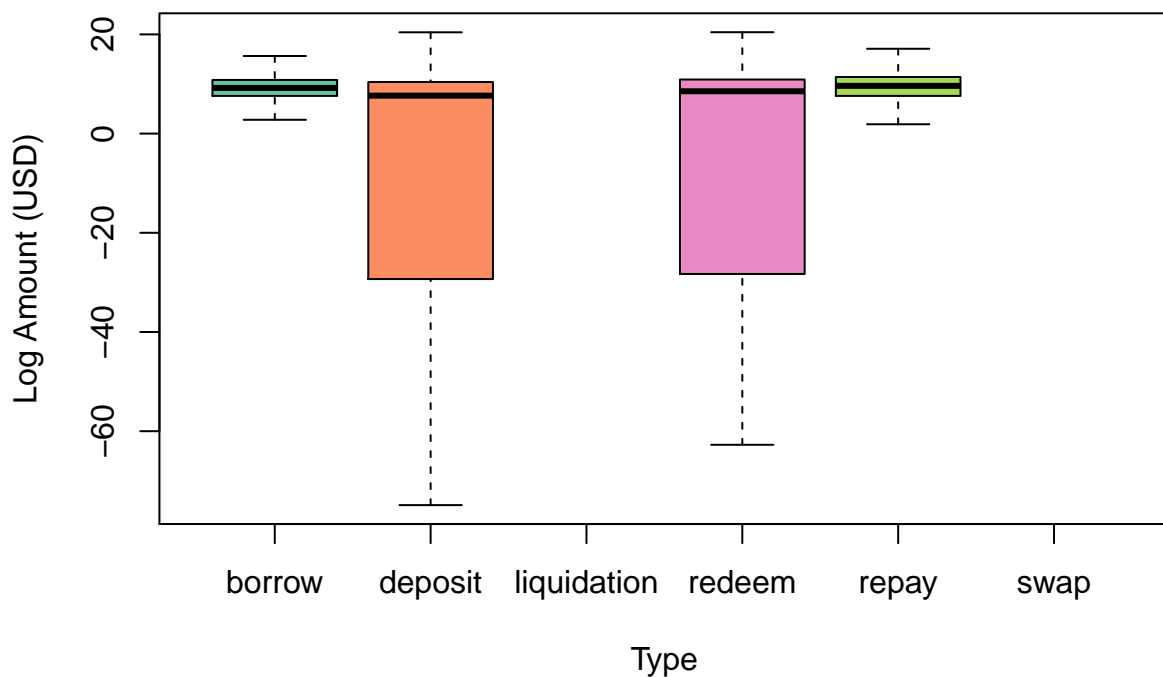
Transaction Amounts



We do find some very large amounts, so it's helpful to look at this on a log scale.

```
boxplot(log(amountUSD)~type,data=df,outline=FALSE,col=colors,  
        main="Log Transaction Amounts",xlab="Type",ylab="Log Amount (USD)")
```

Log Transaction Amounts



Observation: *There are many borrows and repays with high transactions amounts, but deposits and redeems*

have much lower transactions amounts.

Examine Reserve Coins

There are 50 different “Reserve” coins used in transactions in AAVE. Let’s create a table of those reserve coins with at least 500 transactions and rank order them by their volume.

```
# Use dplyr to drop NA reserves, add the counts and then keep only the top 20
reservecoins <- df %>% drop_na(reserve) %>%
  count(reserve) %>%
  arrange(-n) %>%
  head(20)

# Add the rank to help keep track of the reserve coins
reservecoins <- reservecoins %>%
  mutate(rank=1:nrow(reservecoins), .before=reserve)

# List the results nicely with kable()
kable(reservecoins)
```

rank	reserve	n
1	USDC	105937
2	WETH	105279
3	USDT	58266
4	DAI	55211
5	LINK	26404
6	WBTC	26344
7	AAVE	12174
8	CRV	10593
9	UNI	7547
10	XSUSHI	7337
11	SNX	6938
12	SUSD	6542
13		6289
14	GUSD	6009
15	YFI	5919
16	BUSD	4863
17	TUSD	3317
18	BAL	3152
19	MKR	3101
20	REN	2638

Let’s look at the number of transactions types for each currency.

```
TopcoinSummary <- df %>% filter(reserve %in% reservecoins$reserve) %>%
  group_by(reserve) %>%
  count(type) %>%
  mutate(percent = n/sum(n)*100)

kable(TopcoinSummary)
```

reserve	type	n	percent
	liquidation	6289	100.0000000

reserve	type	n	percent
AAVE	borrow	2	0.0164285
AAVE	deposit	7028	57.7295876
AAVE	redeem	5141	42.2293412
AAVE	repay	3	0.0246427
BAL	borrow	215	6.8210660
BAL	deposit	2171	68.8769036
BAL	redeem	612	19.4162437
BAL	repay	154	4.8857868
BUSD	borrow	1685	34.6493934
BUSD	deposit	1135	23.3395024
BUSD	redeem	836	17.1910343
BUSD	repay	1207	24.8200699
CRV	borrow	1054	9.9499670
CRV	deposit	5780	54.5643349
CRV	redeem	2607	24.6105919
CRV	repay	1152	10.8751062
DAI	borrow	14133	25.5981598
DAI	deposit	18552	33.6019996
DAI	redeem	13381	24.2361124
DAI	repay	8895	16.1109199
DAI	swap	250	0.4528083
GUSD	borrow	2282	37.9763688
GUSD	deposit	1493	24.8460642
GUSD	redeem	967	16.0925279
GUSD	repay	1267	21.0850391
LINK	borrow	1321	5.0030298
LINK	deposit	15270	57.8321466
LINK	redeem	8713	32.9987881
LINK	repay	1097	4.1546735
LINK	swap	3	0.0113619
MKR	borrow	188	6.0625605
MKR	deposit	1766	56.9493712
MKR	redeem	986	31.7961948
MKR	repay	159	5.1273783
MKR	swap	2	0.0644953
REN	borrow	196	7.4298711
REN	deposit	1417	53.7149356
REN	redeem	840	31.8423048
REN	repay	183	6.9370735
REN	swap	2	0.0758150
SNX	borrow	433	6.2409916
SNX	deposit	4002	57.6823292
SNX	redeem	2052	29.5762468
SNX	repay	451	6.5004324
SUSD	borrow	1277	19.5200245
SUSD	deposit	2403	36.7318863
SUSD	redeem	1781	27.2240905
SUSD	repay	1081	16.5239988
TUSD	borrow	991	29.8763943
TUSD	deposit	853	25.7160084
TUSD	redeem	661	19.9276455
TUSD	repay	796	23.9975882

reserve	type	n	percent
TUSD	swap	16	0.4823636
UNI	borrow	567	7.5129190
UNI	deposit	3912	51.8351663
UNI	redeem	2540	33.6557573
UNI	repay	527	6.9829071
UNI	swap	1	0.0132503
USDC	borrow	35469	33.4812200
USDC	deposit	27586	26.0400049
USDC	redeem	22131	20.8907181
USDC	repay	20326	19.1868752
USDC	swap	425	0.4011818
USDT	borrow	22332	38.3276697
USDT	deposit	12593	21.6129475
USDT	redeem	10349	17.7616449
USDT	repay	12719	21.8291971
USDT	swap	273	0.4685408
WBTC	borrow	2082	7.9031278
WBTC	deposit	13994	53.1202551
WBTC	redeem	8442	32.0452475
WBTC	repay	1816	6.8934103
WBTC	swap	10	0.0379593
WETH	borrow	7234	6.8712659
WETH	deposit	56373	53.5462913
WETH	redeem	35505	33.7246744
WETH	repay	6155	5.8463701
WETH	swap	12	0.0113983
XSUSHI	borrow	242	3.2983508
XSUSHI	deposit	4382	59.7246831
XSUSHI	redeem	2454	33.4469129
XSUSHI	repay	259	3.5300532
YFI	borrow	403	6.8085825
YFI	deposit	2976	50.2787633
YFI	redeem	2146	36.2561243
YFI	repay	394	6.6565298

Look at Sample User Transaction Histories

Finally, we will examine the transaction history of different users. To do this, we will select 3 random users from the data who have completed between 100 and 300 transactions. Then, we create swarmplots displaying the different types of transactions those users made over time.

```
#set seed
set.seed(1)

# Select three random users that have between 100 and 300 transactions
users<-vector(length=3)
count<-0
while(count<=3){
  success<-FALSE
  while(!success){
    #get random user
    ruser<-sample(df$user,1)
```

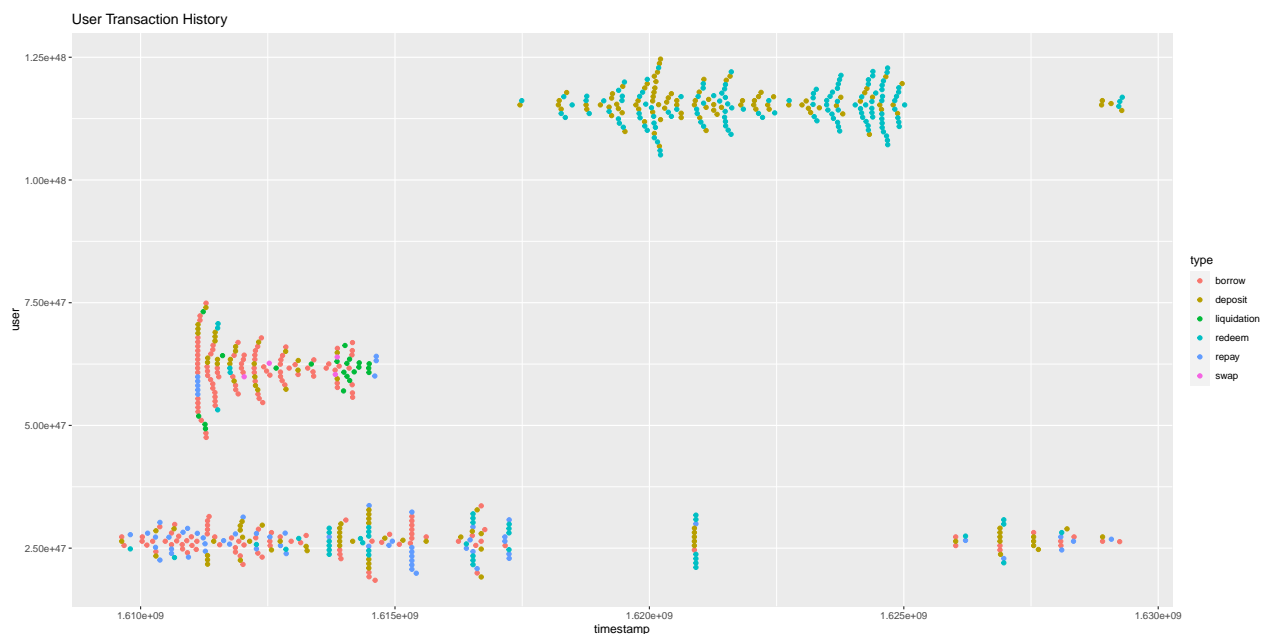
```

#check for valid number of transactions
length<-nrow(filter(df,user==ruser))
if (length>100 && length<300){
  users[count]=ruser
  success<-TRUE
  count<-count+1
}
}
}
df.rusers<-filter(df, user %in%users)

# Create a "swarmplot"

ggplot(df.rusers,aes(user, timestamp,color=type)) +
  geom_beeswarm(cex=1)+
  coord_flip()+
  ggtitle("User Transaction History")

```



Observation: *Users have very different transactions patterns, which we will try to better understand.*

Activities

##Exercise 1(10pts)

1. Divide the top 20 reserve coins between your team members so each team member has the same amount of coins; feel free to add or subtract coins if necessary.
2. Look up your coins on the internet to find out what they are. ProTip: Look them up on <http://defipulse.com> to see their *Total Value Locked (TVL)*.
3. Examine the percentage of transaction types for your coins. Hypothesize why a given coin might have more of one type of transaction than another.
4. Prepare one slide with the findings for each of your coins. (one slide summarizing each coin)

- Coordinate with your team, to combine each of your member's coin descriptions into a single presentation. Please develop a common format or template for presenting your coin summaries so that the common information for a coin (e.g. TVL) is shown in the same format. (Your slide summaries should look the same!)
- Be prepared to present your team presentation to your client in class on Thursday 9/16.

##Exercise 2(10pts)

- Perform a “creative exploration” of some aspect of the Defi data that you find interesting. Add your work in this notebook. *Your work should include at least one data visualization.*
- Put your work in this notebook.
- Write a paragraph describing your findings in the context of DeFi.
- Be prepared to share (2 minutes) in team meeting.
- You'll receive extra credit for work that goes “above and beyond” this assignment!*

#Data for Stable Coins

```
StableCoinSummary <- df %>% filter(reserve %in% reservecoins$reserve) %>%
  filter(reserve == "BUSD" | reserve == "DAI" | reserve == "GUSD" | reserve == "SUSD" | reserve == "TUSD")
  group_by(reserve) %>%
  count(type) %>%
  mutate(percent = n/sum(n)*100) %>%
  group_by(type) %>%
  summarize(percentage_mean = mean(percent), percentage_sd = sd(percent))

kable(StableCoinSummary)
```

type	percentage_mean	percentage_sd
borrow	31.3470329	6.870567
deposit	27.4126305	5.578730
redeem	20.4748248	4.031571
repay	20.5076698	3.410808
swap	0.4512236	0.035479

#Data for Unstable Coins

```
UnstableCoinSummary <- df %>% filter(reserve %in% reservecoins$reserve) %>%
  filter(reserve != "BUSD" & reserve != "DAI" & reserve != "GUSD" & reserve != "SUSD" & reserve != "TUSD")
  group_by(reserve) %>%
  count(type) %>%
  mutate(percent = n/sum(n)*100) %>%
  group_by(type) %>%
  summarize(percentage_mean = mean(percent), percentage_sd = sd(percent))

kable(UnstableCoinSummary)
```

type	percentage_mean	percentage_sd
borrow	6.1598467	2.5135273
deposit	56.3212306	4.8636319
redeem	31.7998690	5.6405824

type	percentage_mean	percentage_sd
repay	5.7011970	2.5806346
swap	0.0357134	0.0287416

Stable Coins = {busd, dai, gusd, susd, tusd, usdc, usdt}

The idea for this little exploration was to see if there is any sort of pattern in Stable vs Unstable types of coins. Although there are many different kinds of unstable coins, it looks like we can see that there are some notable differences between stable and unstable coins. The absolute majority of unstable coins is used for deposits and redeem, while only about 12% combined are used for borrowing and repayments. At the same time, stable coins are used much more evenly in terms of operations being performed with a bit more use of borrow and deposit operations.

It is important to note, however, that although there looks to be some trend in the data, the standard deviation is relatively high (even in stable coins which was a bit of surprise), which means that all coins are still used rather differently no matter what ‘group’ they belong to.