Data Analysis of ZRX (An ERC20 Token)

Nishant Gurrapadi, Huan Lei

A Brief Introduction to Ethereum

Ethereum, can be thought of as a software platform that acts as a decentralized internet as well as a decentralized app store. Ether is the native currency of the Ethereum network. Ethereum, being a decentralized system, is not controlled by anyone. Ethereum has its own blockchain, which enables users to create decentralized applications on it.

Further reading: What is ethereum? [1]

ERC20 tokens

ERC20 is a protocol standard that defines certain rules and standards for issuing tokens on Ethereum's blockchain. The same way we have the HTTP protocol for internet, we have a standard protocol for tokens to be issued on Ethereum, i.e, ERC20.

Now an ERC20 token is simply any token that follows the ERC20 guidelines. To better reinforce the idea of how these tokens work, let's take the example of a video game arcade: Let's say that you visited an arcade gaming center. Here are some of the steps you may need to take before you get to play the game of your choice on the machine:

1. You take your money in dollars and convert them into arcade coins
2. You use these coins to play your favorite game by inserting them into the slot. The slots are desigend to accept circular shaped coins.
3. After you are done, you take the remaining coins, and convert them back into dollars.

Now let's come back to the crypto world. The arcade machines in our case are the decentralized apps (Dapp) while the arcade coins are your native tokens. You need these coins in order to access the services of the Dapp.

In order to fit into the slot of the machine, the coins need to be of circular shape. Now, what if certain machines came about which didn't accept circular coins but preferred a square shaped coin? To ensure smooth running of business, the arcade owner must establish a rule that all arcade machines must only accept circular shaped coins. That, in essence is what the ERC20 guidline does.

Further reading: Beginner's guide to 0x [2]

What is ZRX?

0x is an open protocol designed to offer decentralized cryptocurrency exchange as part of the Ethereum blockchain.
Decentralized exchanges were developed to address the issue of vulnerability to theft. With this type of cryptocurrency exchange, instead of sending money right to a wallet controlled by one entity, users rely on digital signatures to directy authorize trade orders.

Further reading: ERC20 Tokens: A Comprehensive Origin Story [3]

The main use for ZRX tokens is to offer decentralized control over 0x protocol's upgrade system, meaning that ZRX owners have the authority to say how the protocol should be developed over time.

Each token of ZRX unit is made up of \(10^{18}\) subunits. There are 1 billion ZRX tokens in supply, meaning there are \(1 billion * 10^{18} = 10^{27}\) subunits in total.

About the dataset

The dataset consists of 295,393 rows, each representing a transaction (buy/sell) of some amount of the ZRX token between two users.

```
token_data <- read.table("networkzrxTX.txt")
names(token_data) <- c("From Node ID", "To Node ID", "unixTime", "token amount")
head(token_data, 5)
```

Note that the token amounts in the datasets are represented in the subunits.

Preprocessing

Detecting Outliers

Before we proceeded forward with the data analysis, we wanted to make sure there weren't any invalid transactions in our data. Since the total supply is \(10^{27}\) subunits, an easy way to rule out failed transactions are those with token amounts exceeding \(10^{27}\).

```
token_data[token_data$`token amount` > 10^27,]
```

There are 6 such transactions in the dataset. When we looked at the nodeIDs of these transactions, we saw that a majority of them came from one user.

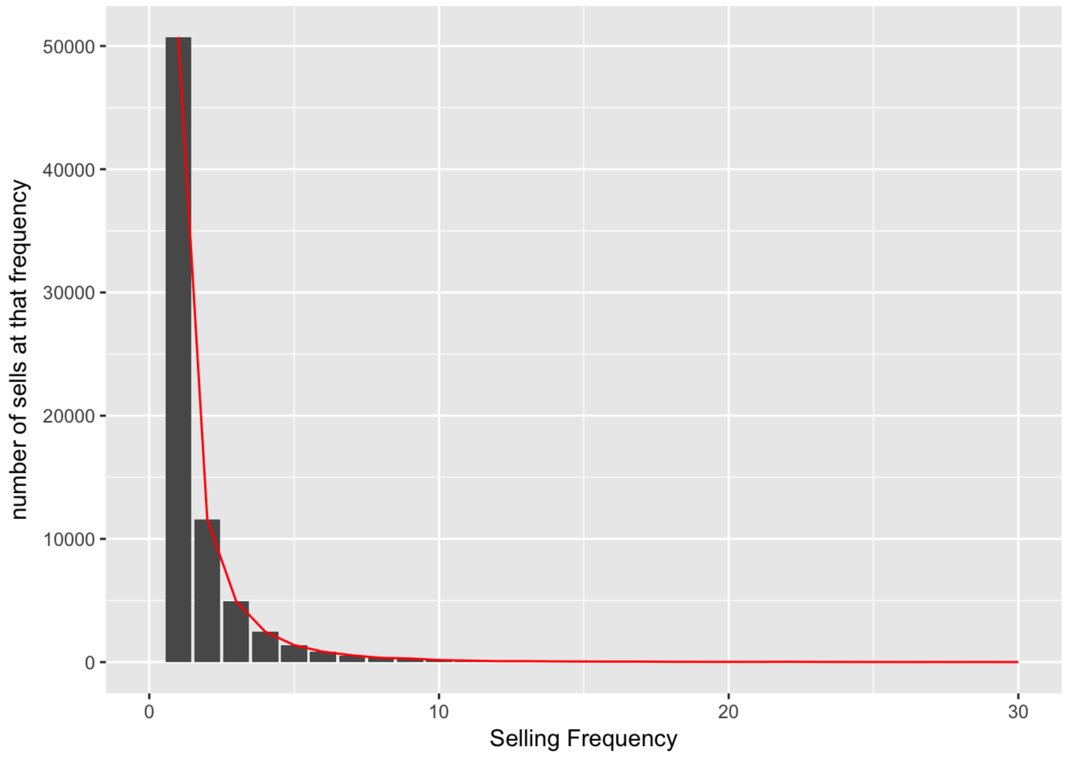
We first split the buy and sell transactions into separate datasets. For each user, we aggregated their selling frequency.

```r
token_data = token_data[!token_data$`token amount` > 10^27,]
sellers = as.data.frame(table(token_data["From Node ID"]))
names(sellers) <- c("Node ID", "SellingFrequency")
head(sellers, 5)
```

Upon observing this, we noticed how there exist a select few users who happen to make a large number of transactions. The same pattern was observed for buyers as well.

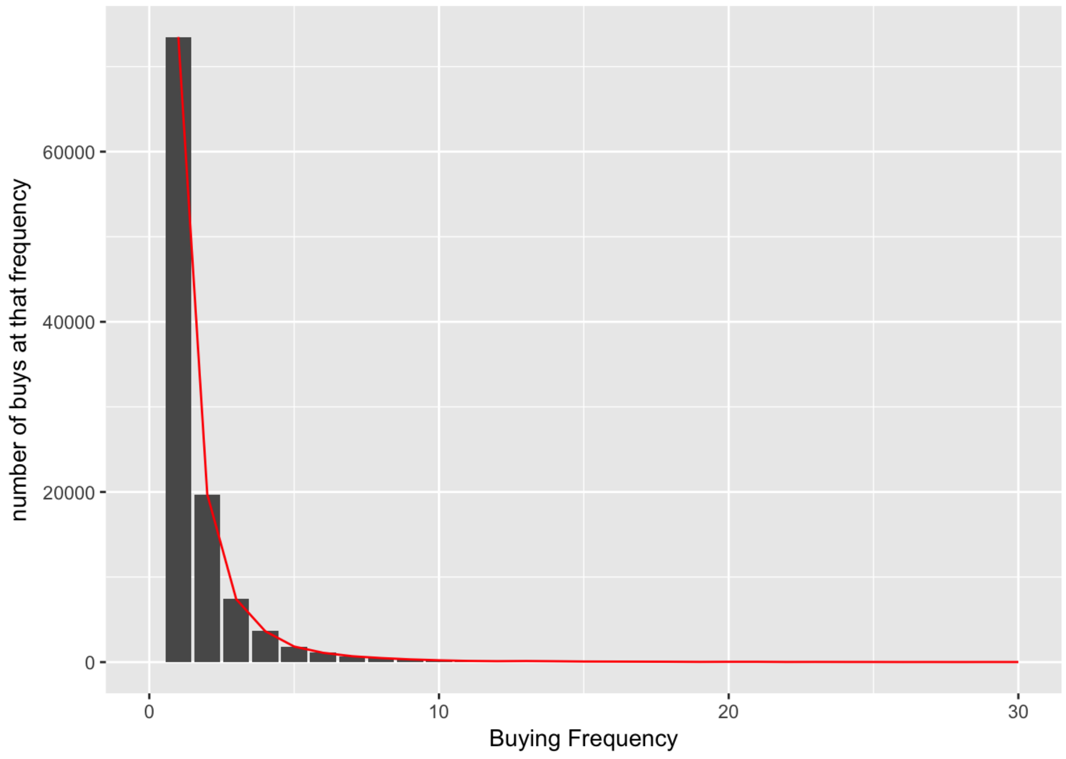
Analysis

Now that we finished preprocessing and removed the outliers, we wanted to learn more about the data. Earlier, we observed that there are a select few users who are responsible for a large chunk of the transactions, and many users who only engage in few transactions. To confirm this is true across the entire dataset, we plotted a histogram of the selling and buying frequency counts of the users.



And voila, this pattern does hold true across all the transactions. The bar chart shows us that a majority of the users usually engage in very few transactions. In fact, the users who were only involved in 1 selling transaction accounted for 73,481 transactions!

A similar pattern was observed in buys as well.



Estimating the Distribution

After plotting the bar chart, we can see a clear pattern between buying/selling frequency and the number of buyers/sellers at that frequency with most of the users engaged in fairly small transaction frequencies. Among these, a majority of users were only involved in one buying and selling transaction.

Based on the pattern and features, our initial assumption that the data resembled an exponential distribution.

We first calculated the mean number of buys and sells for each user.

```
mean(sellers$SellingFrequency)
```

```
## [1] 3.981011
```

Now to calculate \(\lambda\), we need the inverse of the mean, i.e, \(\frac{1}{\mu}\)

```
1/mean(sellers$SellingFrequency)
```

```
## [1] 0.2511925
```

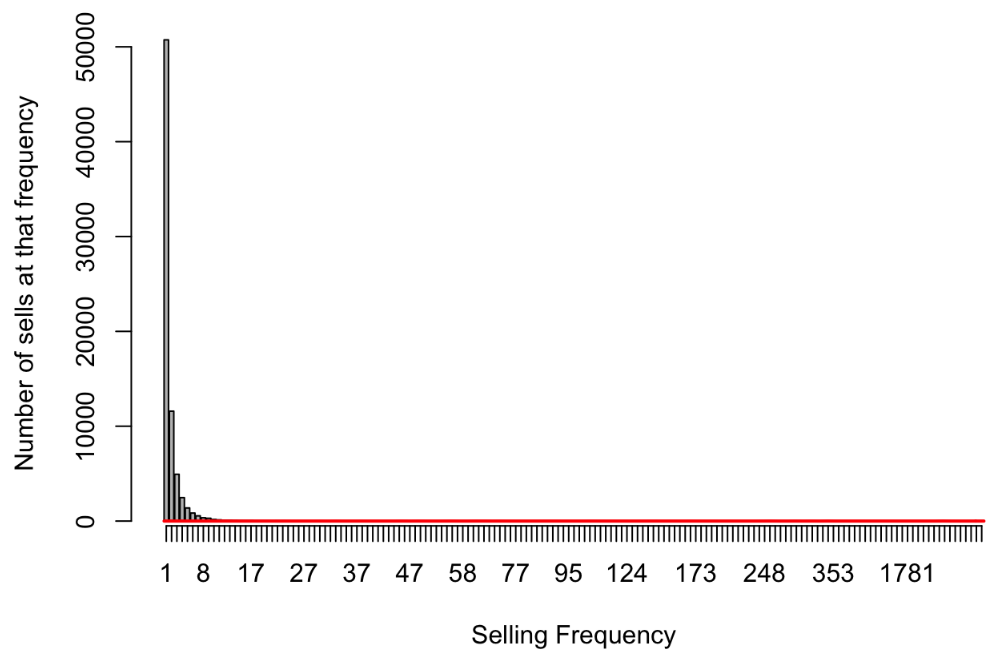
We fitted the exponential distribution to the data and compared the estimated \(\lambda\) value to the one we obtained.

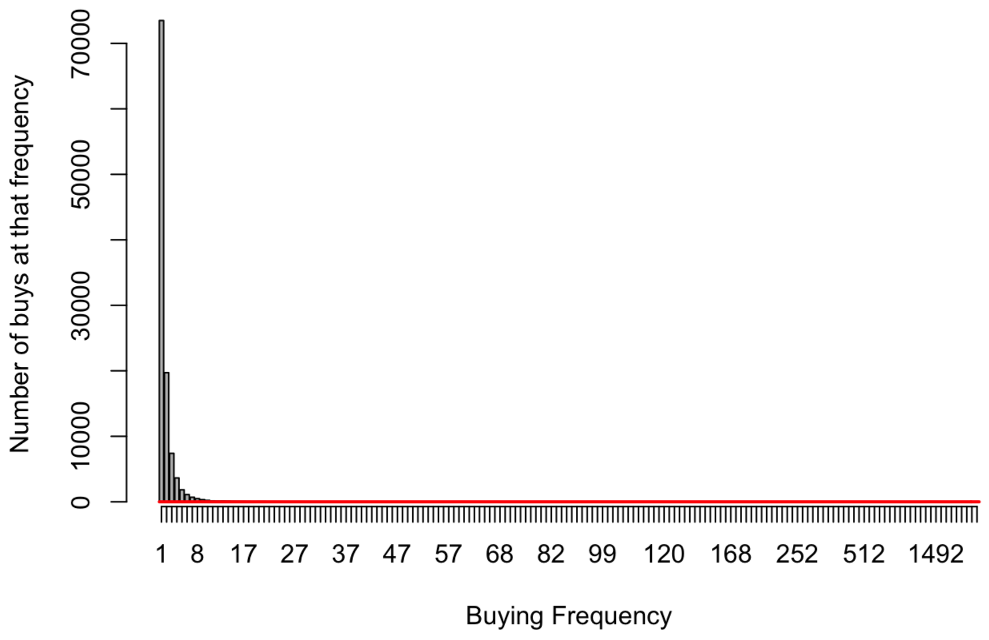
```
library(MASS)
fitdistr(sellers$SellingFrequency, "exponential")$estimate
```

```
##      rate
## 0.2511925
```

We got an exact match for \(\lambda\), which confirmed our hypothesis that the distribution of the selling frequency follows an exponential distribution. Next, we plotted a curve on the barplot with the estimated parameters. The results were identical in the case of buys as well.

```
library(MASS)
bp <- barplot(sell_counts$Freq, xlab="Selling Frequency", ylab="Number of sells at that frequency")
axis(1, at=bp, labels=sell_counts$Var1)
expfit = fitdistr(sellers$SellingFrequency, "exponential")$estimate
x <- rexp(nrow(sellers), rate= expfit)
curve(rexp, add = TRUE, col = "red", lwd=2)
```



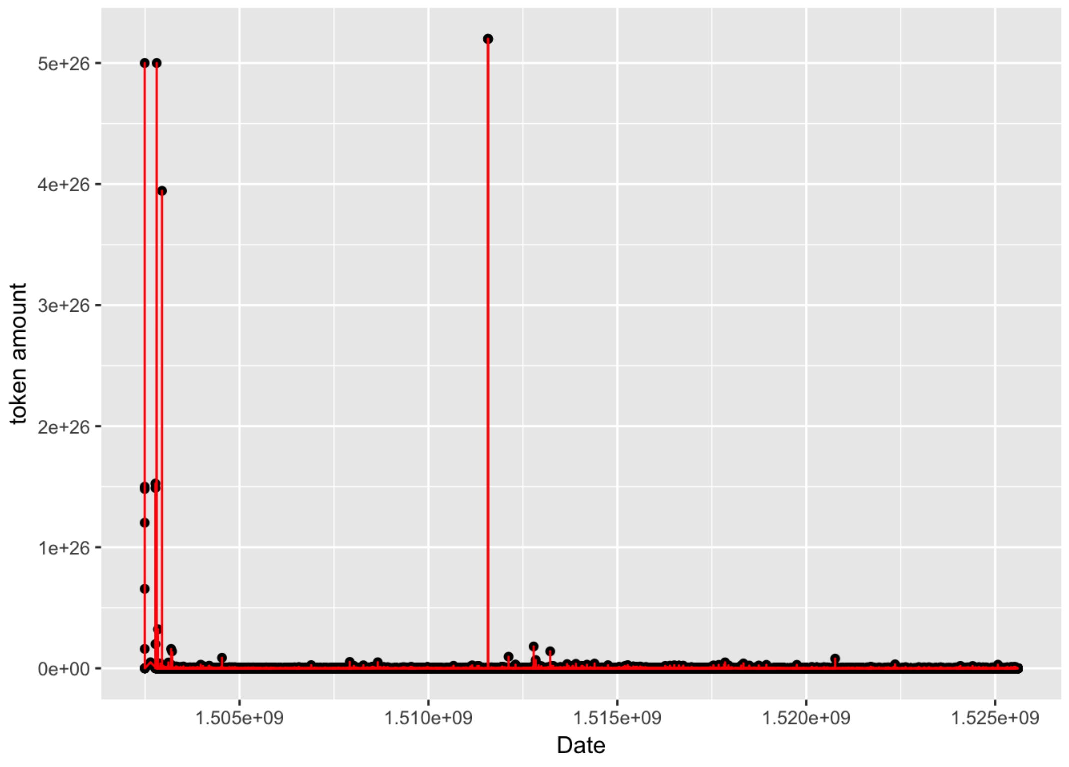


We continued analyzing the ZRX token and attempted to find correlations between the price at Open on a particular day, versus the number of buy/sell transactions that took place on that day.

Initial Findings

We first created a scatter plot of the Date vs Token amounts purchased by users to try and see if there are any patterns that can be discerned from it.

```
library(ggplot2)
ggplot(token_data, aes(x=Date, y=`token amount`)) + geom_point() + geom_line(color="red")
```



Interestingly, the spikes in the beginning indicate that there were a few users who bought tokens aggressively in large amounts when the ZRX token first went public as the prices were also at their lowest. We expected to see gradual decline in the tokens purchased through time, but that didn't happen to be the case. So we continued investigating the transactions in more detail, and used the number of transactions as a feature instead to find better correlation.

Correlating the token price with the no. of transactions

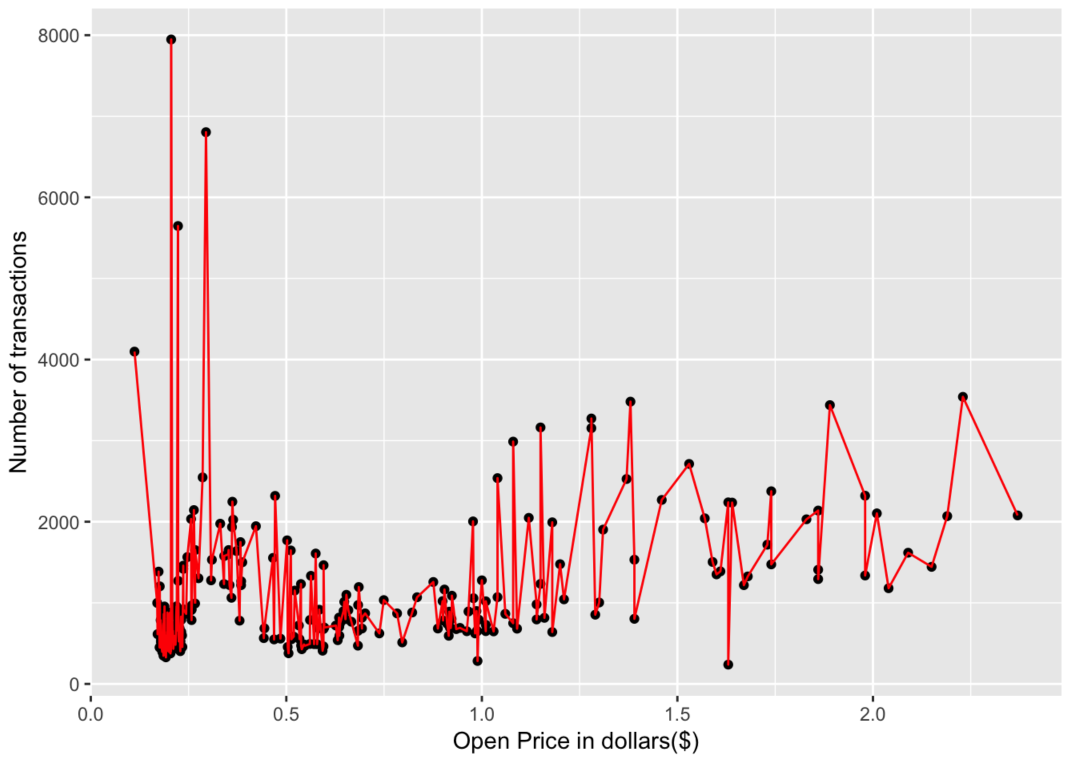
Since our goal is to correlate the prices with the token data, we converted the date column from unix time to standard time using the anytime library.

```r
library(anytime)
date_converter <- function(x) anydate(x)
token_data[3] <- lapply(token_data[3], date_converter)
head(token_data,5)
```

Then, we loaded the price data and merged the token data with the prices using the Date column of both datasets.
To our surprise, we found around ~2000 transactions that took place before the initial coin offering of the ZRX Token. We simply ignored these transactions as we did not have any data about the open and close prices of the token on those days. We then collected the dates and aggregated the number of transactions associated with each of the dates. We then merged the number of transactions for each of the dates into the dataset.

```r
library(plyr)
datecounts = count(merged$Date)
colnames(datecounts)[1] <- "Date"
#head(datecounts,5)
merged_datecount <- merge(merged, datecounts, by  = "Date")
head(merged_datecount)
```

Since our focus is the number of transactions for every unique date, we discard the From Node ID, To Node ID and Token Amount columns. Now, with the number of transactions corresponding to each date, we plotted the Number of Transactions on a particular date vs The Open Price of the token on that day.



Calculating the correlation of price with no. of transactions in different layers

We ran a for loop to try different values and find the layer with a good correlation between the prices and the number of transactions.

```r
max_t = 540141324 * 10^18
n = 0.1
for(i in 1:15){
    layer = merged[merged$`token amount` > n*max_t,]
    datecounts_layer = count(layer$Date)
    colnames(datecounts_layer)[1] <- "Date"
    layer_with_datecount <- merge(layer, datecounts_layer, by = "Date")
    dropped <- c("From Node ID","To Node ID", "token amount")
    layer_trimmed <- layer_with_datecount[ , !(names(layer_with_datecount) %in% dropped)]
    layer_trimmed <- layer_trimmed[!duplicated(layer_trimmed$Date),]
    #print("n=",n, " correlation: ", cor(layer_trimmed$Open, layer_trimmed$freq))
    cat(sprintf("n= %s correlation: %f number of transactions in the layer= %s \n", n, cor(layer_trimmed$Ope
n, layer_trimmed$freq), nrow(layer_with_datecount)))
    n <- n/5
}
```

```
## n= 0.1 correlation: -1.000000 number of transactions in the layer= 4
## n= 0.02 correlation: -0.382667 number of transactions in the layer= 8
## n= 0.004 correlation: 0.030261 number of transactions in the layer= 56
## n= 8e-04 correlation: 0.027357 number of transactions in the layer= 642
## n= 0.00016 correlation: -0.123074 number of transactions in the layer= 5688
## n= 3.2e-05 correlation: -0.131226 number of transactions in the layer= 38443
## n= 6.4e-06 correlation: -0.050099 number of transactions in the layer= 114522
## n= 1.28e-06 correlation: 0.106101 number of transactions in the layer= 191870
## n= 2.56e-07 correlation: 0.242261 number of transactions in the layer= 243990
## n= 5.12e-08 correlation: 0.299355 number of transactions in the layer= 274316
## n= 1.024e-08 correlation: 0.314407 number of transactions in the layer= 286853
## n= 2.048e-09 correlation: 0.312976 number of transactions in the layer= 289525
## n= 4.096e-10 correlation: 0.307627 number of transactions in the layer= 291800
## n= 8.192e-11 correlation: 0.307019 number of transactions in the layer= 292168
## n= 1.6384e-11 correlation: 0.306779 number of transactions in the layer= 292329
```

We ran the loop for 15 iterations, decreasing n by a factor of 5 every iteration. Upon closer observation, we saw that the correlation value reaches 0.31 around the 11th iteration and stagnates.

Contrary to what we initially thought, the positive correlation indicates there has been a steady increase in the number of transactions as the price has increased, over time. This shows that there's growing interest among the crypto enthusiasts to invest into this token.

Estimating the number of layers

From the loop, comparing all the correlation values obtained from each of the layers, we estimate the number of layers to be 11, since it gives us the highest correlation.

Building a regression model to predict price return

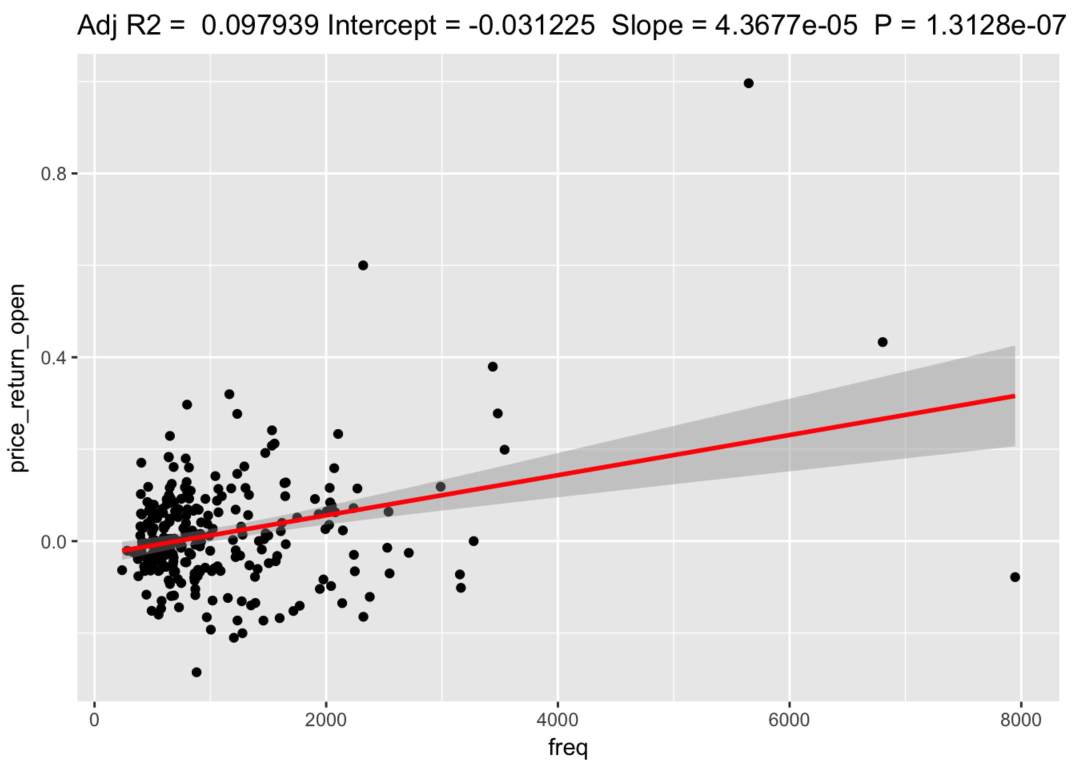
Price return for a token is calculated using the following equation: \(\frac{P_t - P_{t-1}}{P_{t-1}}\)

To compute this, we modified the dataset to include a new column which shows the price return on the ZRX token for each day.

```r
library(dplyr)
sorted_prices <- price_data[order(price_data$Date),]
newprices <- mutate(sorted_prices, prev_high= lag(High), prev_low=lag(Low),prev_open=lag(Open))
newprices <- newprices[-1,]
newprices <- transform(newprices, price_return_open= (Open-prev_open)/prev_open, price_return_low=(Low-prev_
low)/prev_low, price_return_high= (High-prev_high)/prev_high)
merged <- merge(token_data, newprices, by= "Date")
#head(merged)

merged_datecount <- merge(merged, datecounts, by  = "Date")
dropped <- c("From Node ID","To Node ID", "token amount")
merged_trimmed <- merged_datecount[ , !(names(merged_datecount) %in% dropped)]
merged_trimmed <- merged_trimmed[!duplicated(merged_trimmed$Date),]
head(merged_trimmed, 10)
```

Now that we have the price return for each date, we went ahead and implemented the linear regression model to predict the price return using the number of transactions as the sole feature.

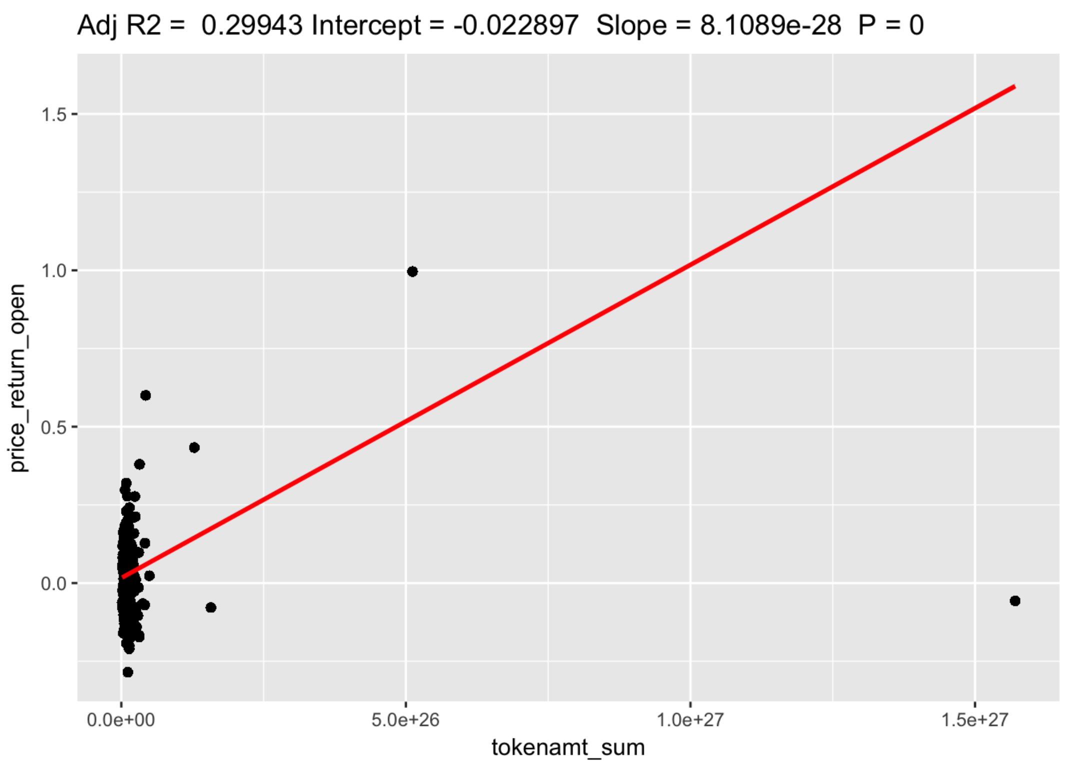


As we can see, the linear regression model got an \(R^2\) value of 0.09, which is pretty low.

Since we only had one feature so far, i.e, the number of transactions per day, we looked for other factors that could potentially impact the price return for a token. We decided to sum the number of tokens sold for each day and use it as an additional feature to see if it would make a difference.

```
## 
## Call:
## lm(formula = price_return_open ~ tokenamt_sum + freq, data = merged_prices_sum_t)
## 
## Coefficients:
##  (Intercept)  tokenamt_sum          freq  
##   -2.290e-02     8.109e-28     2.472e-05
```

```
## 
## Call:
## lm(formula = price_return_open ~ tokenamt_sum + freq, data = merged_prices_sum_t)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -1.31997 -0.07840 -0.00513  0.07416  0.53075 
## 
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  -2.290e-02  4.520e-04  -50.66   <2e-16 ***
## tokenamt_sum  8.109e-28  3.420e-30  237.07   <2e-16 ***
## freq          2.472e-05  2.056e-07  120.26   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.1601 on 289130 degrees of freedom
## Multiple R-squared:  0.2994, Adjusted R-squared:  0.2994 
## F-statistic: 6.179e+04 on 2 and 289130 DF,  p-value: < 2.2e-16
```



And voila, it did! The increase in \(R^2\) indicates that the linear combination of token amount sum and the number of transactions gives a better estimate of the price return than just the number of transactions alone.

Conclusion and Future Work

We have analyzed the ZRX Token, by selecting a few features from user transactions dataset and performing data analysis on it. Then, we extracted the number of transactions for each day,and the sum of tokens purchased for each day, and performed regression on the price return using these two features. Our results showed that using the linear combination of the total amount of tokens purchased and the number of transactions on a particular day is a better estimator than using just the number of transactions alone, to estimate the price return of the token for a given day. There are many more inferences that can be made, such as finding correlations between the number of buyers/sellers and the change in prices, analyzing the transactions in the first few months of the ICO of the token being a few examples. Blockchain technology is growing in popularity and through ERC20 tokens, we can expect to see an increased presence of decentralized applications in the future.

References

[1] ERC20 Tokens: A comprehensive Origin Story, Blockgeeks. Link: https://blockgeeks.com/guides/erc20-tokens/

[2] Oliver Dale, Beginner's Guide to 0x: An Open Protocol for Decentralized Exchanges. Link: https://blockonomi.com/0x-guide/

[3] Alyssa Hertig, What is Ethereum? Link: https://www.coindesk.com/information/what-is-ethereum