@authors: Vineet Vats - vxv180008, Shruti Jaiswal – sxj170027

Analyzing ZRX Token data of Ethereum Blockchain

Background on Tokens and Ethereum

Tokens. Tokens are cryptocurrencies that can be used to represent particular asset or utility and is usually found on another blockchain. A token is an asset that is mutually interchangeable and can be traded. They include commodities, cryptocurrencies, loyalty points and others. Tokens became famous because it is very easy to create them. No code has to be modified from a particular protocol nor do we have to create a blockchain from scratch. There is a standard template on blockchain that must be followed while creating a token. Ethereum is the most popular token among all. Smart contracts have made the creation and use of tokens easy.

Smart Contracts. Smart contracts are a series of instructions, written using the programming language "solidity", which work on the basis of the IFTTT logic also known as the IF-THIS-THEN-THAT logic. We will move to the next step only if the previous step is completed and this process is continued until we reach the end of the contract. The benefit of smart contracts is that every transaction will be recorded and updated on the network, so everyone involved in the contract is accountable for their actions.

Ether. A smart contract is a collection of transactions and complex computation that has a cost associated with it which is measured in "gas". The price of this gas is paid by the requester in "Ether". Ether is the currency with which is used in the Ethereum environment and is hard coded into the Ethereum blockchain. It is sold and traded as a cryptocurrency and used in Ethereum network by users to pay for smart contract transaction fees.

Ethereum. Ethereum is an environment that can be used by any user to take advantage of the blockchain technology to build their own projects and DAPPS (decentralized applications) through smart contracts.

Ethereum Token. Ethereum tokens are the digital assets that are built on top of the Ethereum blockchain, so the developers can take advantage of the existing infrastructure instead of building their own blockchain. The Ethereum ecosystem uses ether as their currency that strengthens the demand for ether, needed to power the smart contracts.

Ethereum tokens can be used to represent a physical object like gold to a native currency used to pay transaction fees and can be used for a variety of purposes such as paying to access a network or for decentralized governance over an organization. In the future, tokens may even be used to represent financial instruments like stocks and bonds. The properties and functions of each token depends how the token is to be used. Tokens can have a fixed supply, constant inflation rate, or even a supply determined by a sophisticated monetary policy. Tokens are issued to the public through ICO also known as initial coin offering. The token is offered in exchange for ether and sometimes bitcoin and other digital currencies.

ERC20 Ethereum Standard for Tokens. The Ethereum blockchain attaches an input data field to each transaction to store log data. This input data field is used by smart contract transactions to transmit messages; creating a transaction with input data to a Smart Contract is same as passing variables to a function. Earlier there were no standards mentioned for the transactions, input messages and functions contained in the Smart Contracts. So, the user must know which structure to use while transmitting a message to the smart contract. Recently, new standards have been imposed, such as ERC201 (Ethereum Request for Comments 20) and ERC223, which define a common list of rules for tokens to follow within the larger Ethereum ecosystem.

ERC-20 defines a set of six functions that other smart contracts within the Ethereum ecosystem will understand and recognize. These include, how to transfer a token (by the owner or on behalf of the owner) and how to access data (name, symbol, supply, balance) about the token. The standard also describes two events – signals that a smart contract can fire – that other smart contracts 'listen' for.

Together, these functions and events make Ethereum tokens work the same everywhere within the Ethereum ecosystem. As a result, nearly all wallets that support ether, including Jaxx, MyEtherWallet.com and Ethereum Wallet (also called Mist Wallet), now also support ERC-20 compliant tokens.

Data Description of the ZRX Token

Our data files contain two primary groups: token network edge files, and token price files. The Ethereum project is a blockchain platform, and our data comes from there. Although Ethereum started in 2015, most tokens have been created since 2016. As such, tokens have different starting dates, and their data starts from that initial date.

Token edge files have this row structure:

fromNodeID toNodeID unixTime tokenAmount

This row implies that fromNodeID sold tokenAmount of the token to toNodeID at time unixTime. fromNodeID and toNodeID are people who invest in the token in real life; each investor can also use multiple addresses. Two addresses can sell/buy tokens multiple times with multiple amounts. For this reason, the network is considered a weighted, directed multi(edge) graph. Each token has a maximum token count maxt, it is the total circulating token amount.

Sample Data from Token edge file:

FromNodeId	ToNodeId	Unixdate	TokenAmount
352161	352162	2018-04-24	9.040900e+20
3133205	75994	2018-04-24	1.072463e+21
104467	104468	2018-04-24	4.616710e+20
1747184	196367	2018-04-24	4.326388e+20
1747184	315284	2018-04-24	6.551790e+19
17	9944329	2018-04-24	5.938776e+20
312135	9944330	2018-04-24	6.438989e+19
40002	53472	2018-04-24	3.692400e+20

Token price file has the following row structure

Date Open High Low Close Volume MarketCap

The price data is taken from https://coinmarketcap.com/. Open and close are the prices of the specific token at the given date. Volume and MarketCap give total bought/sold tokens and market valuation at the particular date.

Sample Data from Token price file:

Date	Open	High	Low	Close	Volume	MarketCap
7/4/2018	0.887873	1.030000	0.859933	0.949942	18,949,000	473,463,000
7/3/2018	0.948858	0.988706	0.886003	0.891924	21,272,400	506,007,000
7/2/2018	0.793840	0.989407	0.789722	0.950831	40,158,300	423,247,000
7/1/2018	0.790663	0.835991	0.735183	0.796163	12,751,400	421,499,000
6/30/2018	0.659480	0.834762	0.659480	0.791598	25,434,300	351,506,000
6/29/2018	0.606791	0.676429	0.583630	0.660468	9,439,390	323,052,000
6/28/2018	0.642657	0.661132	0.602466	0.607078	6,446,650	342,188,000

Background about ZRX Token.

ZRX token is an Ethereum token used for paying trading fees to Relayers for their services. The main use for ZRX is to offer decentralized control over 0x protocol's upgrade system, meaning that ZRX token owners have the authority to say how the protocol should be developed over time. ZRX is inherently a governance token that plays a critical role in the process of upgrading 0x protocol. Token owners can use the currency to vote on decisions that concern the blockchain, such as protocol additions and upgrades, without disrupting the network's functionality.

0x is an open protocol that allows any ERC20 token to be traded on the Ethereum blockchain which will be used to build a decentralized exchange. 0x aims to make token trading with off-chain transactions more efficient by running smart contracts on the Ethereum blockchain .Founded in October of 2016, 0x was created with the purpose of enabling anyone to trade freely any type of asset and to let any user operate a decentralized exchange.

ZRX can be traded on some of the most popular exchanges—which includes Coinbase as of October 018. It is listed on several major exchanges such as Binance, OKEx, and Bittrex.The token can be traded for Bitcoin, Ethereum, and even for USD, and USDT. Most of the trading volume comes from Binance and OKEx. As mentioned before, ZRX is an ERC20-based token, so any Ethereum-compatible wallet will support 0x.

The total supply for the ZRX token is 1,000,000,000 ZRX ($765,413,604.10). Associated dollar value for each ZRX token is $0.7654 @ 0.003867 Eth as on 31stOctober, 2018. This dollar value is listed in online exchange websites such as etherscan.io. The value is arbitrated by token demand and supply. Most tokens have a fixed supply that is set at the time when a token contract is created. As the supply is fixed, the value of a token is mostly determined by its demand.

Project Description

- Through this project we are finding the distribution of how many times a user buys and sells a token. We will try to fit the distribution in one of the distribution types that can best describe it and then estimate the distribution parameters.
- In the second part of this project we will create different layers of transactions with increasing amounts. The number of layers is selected through hit and trial to get the number that best divides our data. Using these layers, we can compute various features in each layer like the number of transactions, number of buyers etc.

Libraries used:

The tidyverse is a set of packages that work in harmony because they share common data representations and API design. The tidyverse package is designed to make it easy to install and load core packages from the tidyverse in a single command.

```
library(tidyverse)
```

The sqldf() function is typically passed a single argument which is an SQL select statement where the table names are ordinary R data frame names. sqldf() transparently sets up a database, imports the data frames into that database, performs the SQL select or other statement and returns the result using a heuristic to determine which class to assign to each column of the returned data frame.

```
library(sqldf)
```

'ggpubr' provides some easy-to-use functions for creating and customizing 'ggplot2'- based publication ready plots. We are using this library to use a scatterplot function.

```
library("ggpubr")
```

Extends the fitdistr() function with several functions to help the fit of a parametric distribution to data. We are using this function to fit Poisson distribution to our data.

```
library(fitdistrplus)
```

Main Functions used in the project:

- **plot( ) -** Generic function for plotting of R objects
- **barplot( ) -** Creates a bar plot with vertical or horizontal bars.
- **fitdist( ) -** Fit of univariate distributions to non-censored data by maximum likelihood, moment matching , quantile matching or maximizing goodness-of-fit estimation. The latter is also known as minimizing distance estimation.
- **cor.test( ) -** Test for association between paired samples, using one of Pearson's product moment correlation coefficient, Kendall's τ or Spearman's ρ.
- **scatterplot( ) -** Create a scatter plot.
- **Sqldf( ) -** SQL select on data frames and for using SQL queries on a data frame.

Steps followed in our Project:

1. The first step is that we load both the datasets i.e. Token Graph and Token Prices into a data frame.

```
zrxtoken<-read.table(file="D:/networkzrxTX.txt",header=F,sep=" ")
pricezrx<-read.table(file="D:/zrx.txt",header=F,sep="\t")
```

```
summary(zrxtoken)
##    FromNodeId         ToNodeId          Unixdate
##  Min.   :      4   Min.   :      2   Min.   :1.502e+09
##  1st Qu.:  44146   1st Qu.: 104233   1st Qu.:1.506e+09
##  Median :1746888   Median :1759326   Median :1.515e+09
##  Mean   :2523044   Mean   :3406836   Mean   :1.513e+09
##  3rd Qu.:2777146   3rd Qu.:5919106   3rd Qu.:1.519e+09
##  Max.   :9982269   Max.   :9982274   Max.   :1.526e+09
##   TokenAmount
##  Min.   :1.000e+00
##  1st Qu.:3.060e+20
##  Median :1.994e+21
##  Mean   :2.430e+22
##  3rd Qu.:8.000e+21
##  Max.   :5.199e+26
summary(pricetenx)
##         Date          Open              High              Low
##  1/1/2018 :  1   Min.   :0.1117   Min.   :0.1764   Min.   :0.1040
##  1/10/2018:  1   1st Qu.:0.2281   1st Qu.:0.2387   1st Qu.:0.2121
##      Close              Volume            MarketCap
##  Min.   :0.1708   1,000,210:  1   1,008,110,000:  1
##  1st Qu.:0.2289   1,091,020:  1   1,017,560,000:  1
##  [ reached getOption("max.print") -- omitted 5 rows ]
```

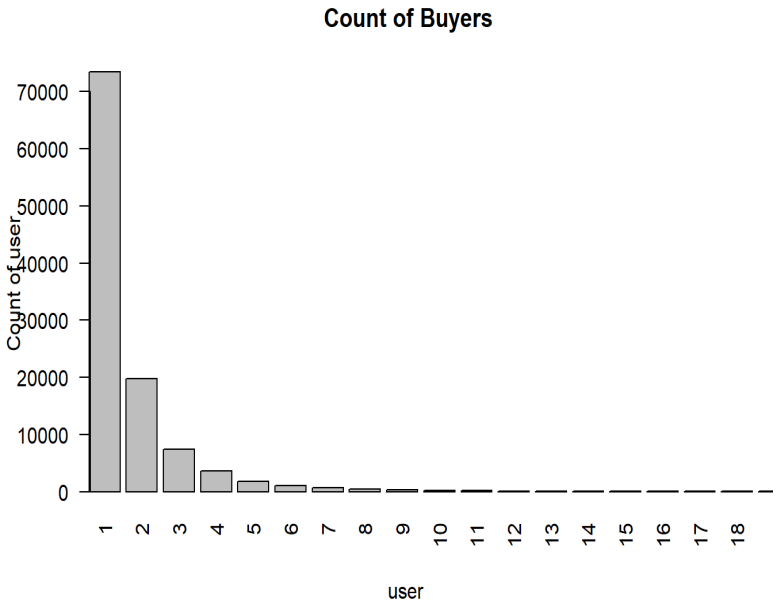
2. The second step is removing the outliers. Outliers are numbers or data points that are extremely high or extremely low relative to other data points or values in a dataset. In our ZRX token, all the values greater than 1e+27 are the Outliers as these values are impossible in our data. Given

below are the six outliers which can affect our results, so we have removed them from our dataset.

```
outliersvals <-zrxtoken[ which(zrxtoken$V4>=1e+27), ]
outliersvals
##              V1     V2         V3           V4
## 103704 8105805 198548 1516152231 1.400000e+32
## 104170 8105805 198548 1516170850 1.400000e+32
## 105821 8105805 198548 1516238722 1.400000e+32
## 105838 8105805 198548 1516239903 1.400000e+32
## 277325  121639 120427 1503087173 1.133408e+28
## 277327  121639 120427 1503087188 1.133408e+28
Removing the Outliers here in the code given below:
zrxtoken <- zrxtoken[ which(zrxtoken$V4<=1e+27), ]
```

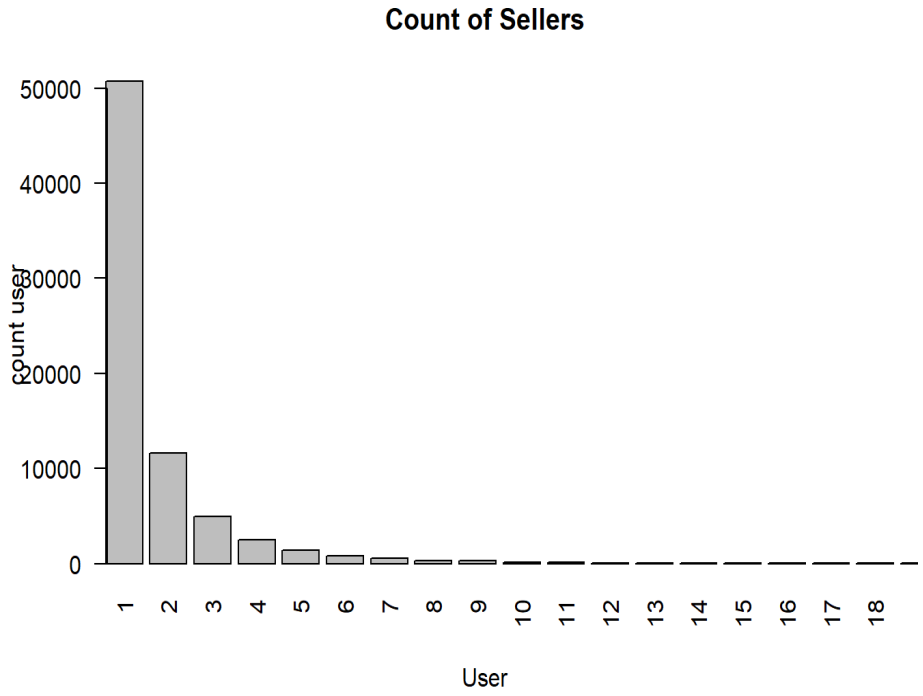
3. Processing the Data according to our needs:
   Now we are taking the count of Users who are buying the token and after that we are taking the count of the frequency of the user buying token so that we can plot that how many times a user buy a token.

```
Buytoken <-c(zrxtoken[,2])
CountOfBuy<-as.data.frame(table(Buytoken))
colnames(CountOfBuy)<-c("UserId","Frequency")
DoubleCountOfBuy<-as.data.frame(table(CountOfBuy$Frequency))
colnames(DoubleCountOfBuy)<-c("Count_buy","Count_user")
```



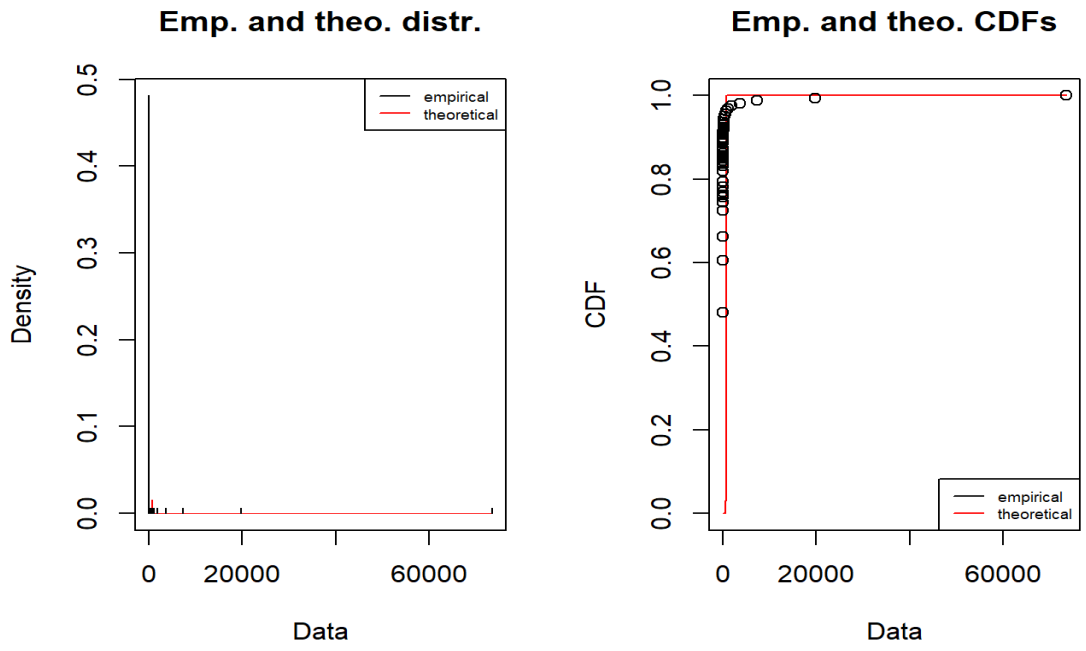
4. Now we are visualizing the data in form of bar plots.
   The above graph shows that how many times a user buys a token

Similarly, we have found that how many times a particular user sells a token.



5. Our goal is to find the type of discrete distribution which can fit the distribution best. The Poisson distribution fits the distribution best and is defined as the probability distribution of independent event occurrences in an interval. The distribution is shown below.

```
No.pois<-fitdist(DoubleCountOfBuy[,2], distr = "pois")
plot(No.pois)
```



6. Now we have reached half of the way and when we know that our data is following a Poisson distribution, so we would like to find correlation among the attributes.

7. For that firstly we will convert the Unix date into Standard date format.

```
zrxtoken$Unixdate<-   as.Date(as.POSIXct(zrxtoken$Unixdate,   origin="1970-01-01"))
zrxtoken$Unixdate
```

```
##  [1] "2018-04-24" "2018-04-24" "2018-04-24" "2018-04-24" "2018-04-24"
##  [6] "2018-04-24" "2018-04-24" "2018-04-24" "2018-04-24" "2018-04-24"
## [11] "2018-04-24" "2018-04-24" "2018-04-24" "2018-04-24" "2018-04-24"
## [16] "2018-04-24" "2018-04-24" "2018-04-24" "2018-04-24" "2018-04-24"
##  [ reached getOption("max.print") -- omitted 295367 entries ]
```

8. We have divided the data into 7 layers so that data present in each layer is comparable with each other. The bin size is selected on hit and try.

```
layer1<- zrxtoken[which(zrxtoken$TokenAmount>1e+22),]
layer2<-              zrxtoken[which(zrxtoken$TokenAmount<=1e+22              &
zrxtoken$TokenAmount>1e+21),]
layer3<-              zrxtoken[which(zrxtoken$TokenAmount<=1e+21              &
zrxtoken$TokenAmount>1e+20),]


layer4<-              zrxtoken[which(zrxtoken$TokenAmount<=1e+20              &
zrxtoken$TokenAmount>1e+19),]
layer5<-              zrxtoken[which(zrxtoken$TokenAmount<=1e+19              &
zrxtoken$TokenAmount>1e+15),]
layer6<-              zrxtoken[which(zrxtoken$TokenAmount<=1e+15              &
zrxtoken$TokenAmount>1e+5),]
layer7<- zrxtoken[which(zrxtoken$TokenAmount<=1e+5),]
```

9. Finding the frequency of the Transaction happening on a particular date according to a particular layer.

```
Flayer_1 <-c(layer4[,3])
count_layer1<-as.data.frame(table(Flayer_1))
colnames(count_layer1)<-c("Date","Frequency")
count_layer1$Date<-   as.Date(as.POSIXct(count_layer1$Date,   origin="1970-01-
01"))
```

10. Applying SQL commands join ZRX Token Data and ZRX prices data based on the Date which is present in both the Datasets.

```
df3<-sqldf("SELECT f.Date, p.Open, f.Frequency FROM count_layer1 f INNER JOIN
pricezrx p WHERE f.Date=p.Date")
summary(df3)
```

```
##       Date                  Open          Frequency
##  Min.   :2017-08-16   Min.   :0.1117   Min.   : 15.0
##  1st Qu.:2017-10-20   1st Qu.:0.2145   1st Qu.: 52.0
##  Median :2017-12-25   Median :0.5086   Median : 83.5
##  Mean   :2017-12-25   Mean   :0.6550   Mean   :117.3
##  3rd Qu.:2018-03-01   3rd Qu.:0.9687   3rd Qu.:135.0
##  Max.   :2018-05-06   Max.   :2.3700   Max.   :636.0
```

The second thing which we want to find out from the ZRX token is that is there any correlation between the attributes in the data.

11. We are getting Pearson correlation between the token price and Frequency of transaction happening on a particular date.

```
cor.test(df3$Open, df3$Frequency, method = "pearson")
```

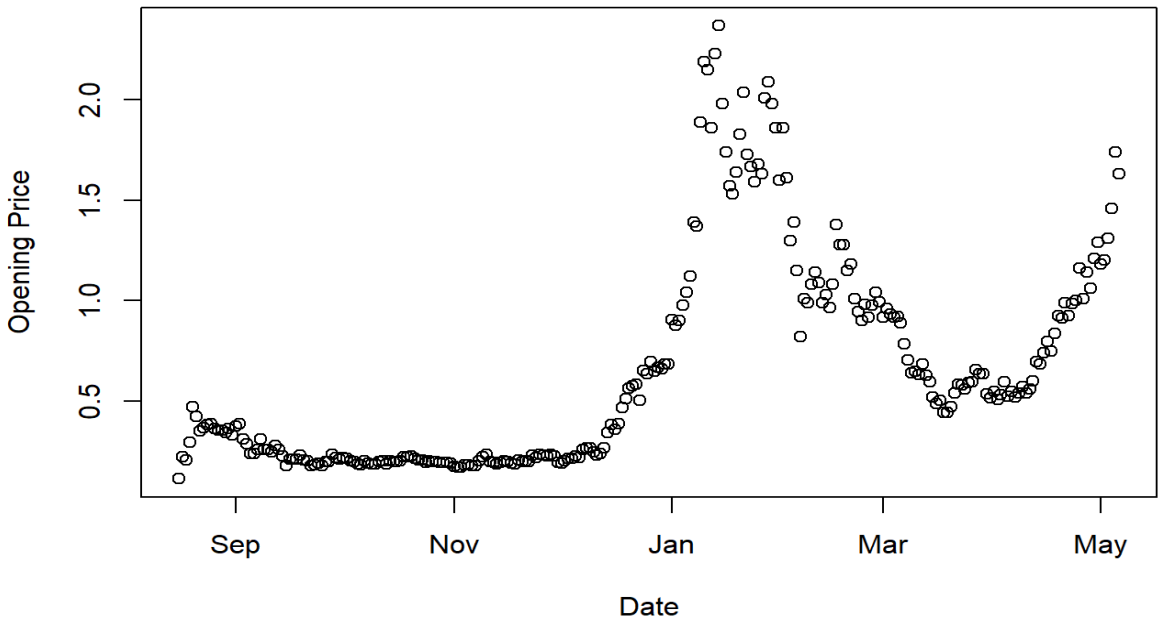
```
##
##  Pearson's product-moment correlation
##
## data:  df3$Open and df3$Frequency
## t = 15.645, df = 262, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.6268344 0.7525555
## sample estimates:
##       cor
## 0.6949691
```

12. When we plot both the data, we can see the correlation frim the graph

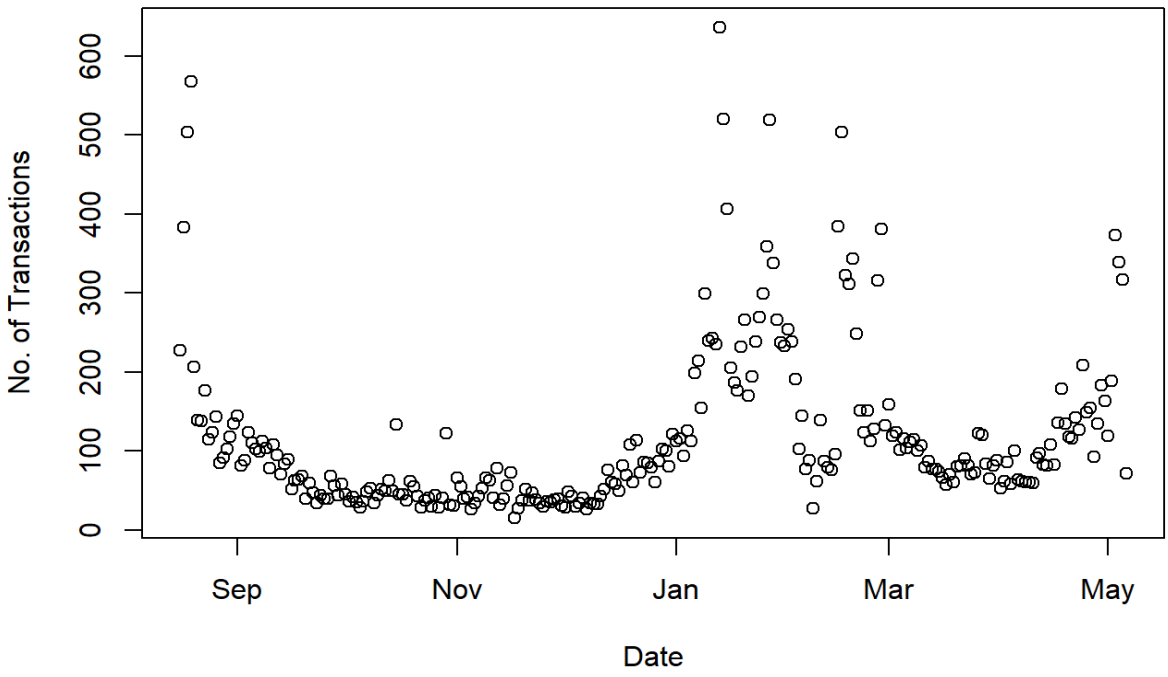
```
plot(df3$Date,df3$Open, main= "Plot between Date and Opening price of token for Layer4(Best Case)", xlab="Date", ylab="Opening Price")


plot(df3$Date,df3$Frequency, main= "Plot between Date and Frequency of TokenAmount for Layer4(Best Case) ", xlab="Date", ylab="No. of Transactions")
```

Plot between Date and Opening price of token for Layer4(Best Case)

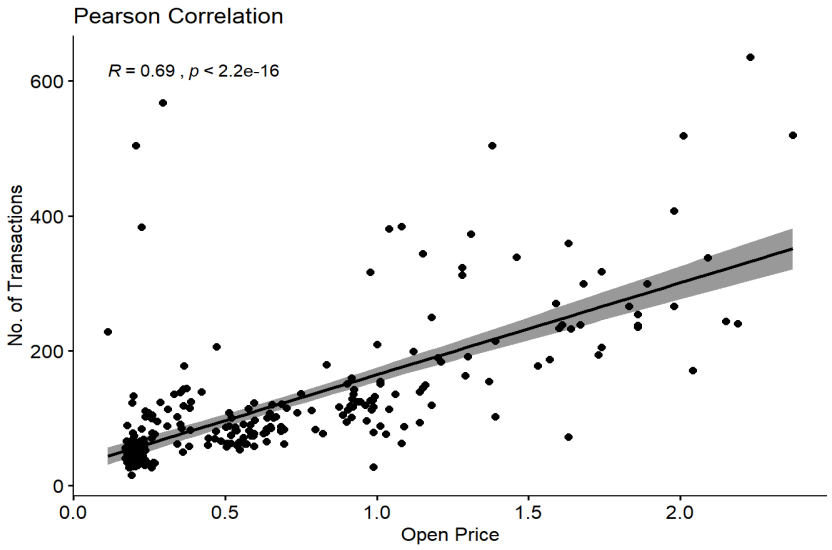


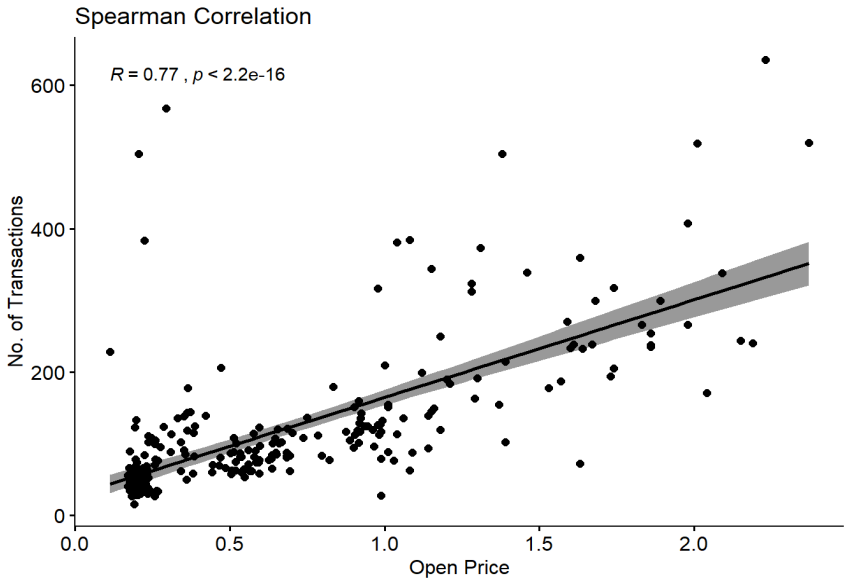
Plot between Date and Frequency of TokenAmount for Layer4(Best Case



13. Plotting a Scatterplot of Open Price Versus No. of Transactions using Pearson and Scatter plot Correlation.

```
ggscatter(df3, x = "Open", y = "Frequency",
          add = "reg.line", conf.int = TRUE,
          cor.coef = TRUE, cor.method = "pearson",
          xlab = "Open Price", ylab = "No. of Transactions", main="Pearson
Correlation")
```





Conclusion:

From our findings, we concluded that the number of times a user buys or sells a ZRX token follows a Poisson distribution. The correlation between Date vs Opening price of a token using Pearson Correlation coefficient is 0.70 and using Spearman Correlation Coefficient is 0.77 when we are taking 7 bins of comparable sizes. We can also check the same by looking at the plot between Date & Price of Token and Date and Frequency of transaction of token.

References:

[1] https://www.rdocumentation.org
[2] https://etherscan.io/
[3] https://cran.r-project.org/manuals.html
[4] https://cointelegraph.com/explained/erc-20-tokens-explained