Veros

PART 1

```
rm(list = ls())
```

```
raw_data <- read.csv(file = 'networkverosTX.txt', header = F, sep = ' ')
colnames(raw_data) <- c("Buyer", "Seller", "Timestamp", "TokenAmount")
message ('The number of rows are: ', nrow(raw_data))
```

```
## The number of rows are: 392742
```

```
summary(raw_data)
```

```
##      Buyer             Seller          Timestamp
##  Min.   :   2553   Min.   :    620   Min.   :1.480e+09
##  1st Qu.:9822214   1st Qu.:9822525   1st Qu.:1.488e+09
##  Median :9822214   Median :9823000   Median :1.493e+09
##  Mean   :7917960   Mean   :9731424   Mean   :1.493e+09
##  3rd Qu.:9822214   3rd Qu.:9823887   3rd Qu.:1.498e+09
##  Max.   :9827454   Max.   :9827454   Max.   :1.526e+09
##   TokenAmount
##  Min.   :1.000e+00
##  1st Qu.:1.000e+06
##  Median :6.400e+06
##  Mean   :3.243e+72
##  3rd Qu.:4.800e+07
##  Max.   :1.158e+77
```

```
Total_circulation_amount = 828954240 * 10^18
outliers <- subset(raw_data, TokenAmount > Total_circulation_amount)
message ('The number of outliers in the dataset are: ', nrow(outliers))
```

```
## The number of outliers in the dataset are: 17
```

```
preprocessed_data <- subset(raw_data, TokenAmount <= Total_circulation_amount)
summary(preprocessed_data)
```

```
##      Buyer             Seller          Timestamp
##  Min.   :   2553   Min.   :    620   Min.   :1.480e+09
##  1st Qu.:9822214   1st Qu.:9822525   1st Qu.:1.488e+09
##  Median :9822214   Median :9823000   Median :1.493e+09
##  Mean   :7917877   Mean   :9731530   Mean   :1.493e+09
##  3rd Qu.:9822214   3rd Qu.:9823887   3rd Qu.:1.498e+09
##  Max.   :9827454   Max.   :9827454   Max.   :1.526e+09
##   TokenAmount
##  Min.   :1.000e+00
##  1st Qu.:1.000e+06
##  Median :6.400e+06
##  Mean   :2.546e+19
##  3rd Qu.:4.800e+07
##  Max.   :1.000e+25
```

```
library(plyr)
Buyer_Seller_Pair_Frequencies <- ddply(preprocessed_data, .(preprocessed_data$Buyer, preprocesse
d_data$Seller), nrow)
names(Buyer_Seller_Pair_Frequencies) <- c("Buyer", "Seller", "Frequency")
Buyer_Seller_Pair_Frequencies
```

Buyer <int>	Seller <int>	Frequency <int>
2553	9821422	1
8089	9824766	2
13760	14600	2
14514	9821227	1
14600	13760	1
14600	9820745	1
16916	16917	1
16917	5502097	1
17806	17807	1
28635	9821236	1

1-10 of 10,000 rows Previous **1** 2 3 4 5 6 … 1000 Next

```
summary(Buyer_Seller_Pair_Frequencies)
```

```
##      Buyer             Seller          Frequency
##  Min.   :   2553   Min.   :    620   Min.   :   1.00
##  1st Qu.:9821354   1st Qu.:9821354   1st Qu.:   1.00
##  Median :9822214   Median :9823000   Median :   2.00
##  Mean   :8286398   Mean   :8838975   Mean   :  33.86
##  3rd Qu.:9823634   3rd Qu.:9825268   3rd Qu.:  26.00
##  Max.   :9827454   Max.   :9827454   Max.   :5119.00
```

```
message ('Variance: ', var(Buyer_Seller_Pair_Frequencies$Frequency))
```

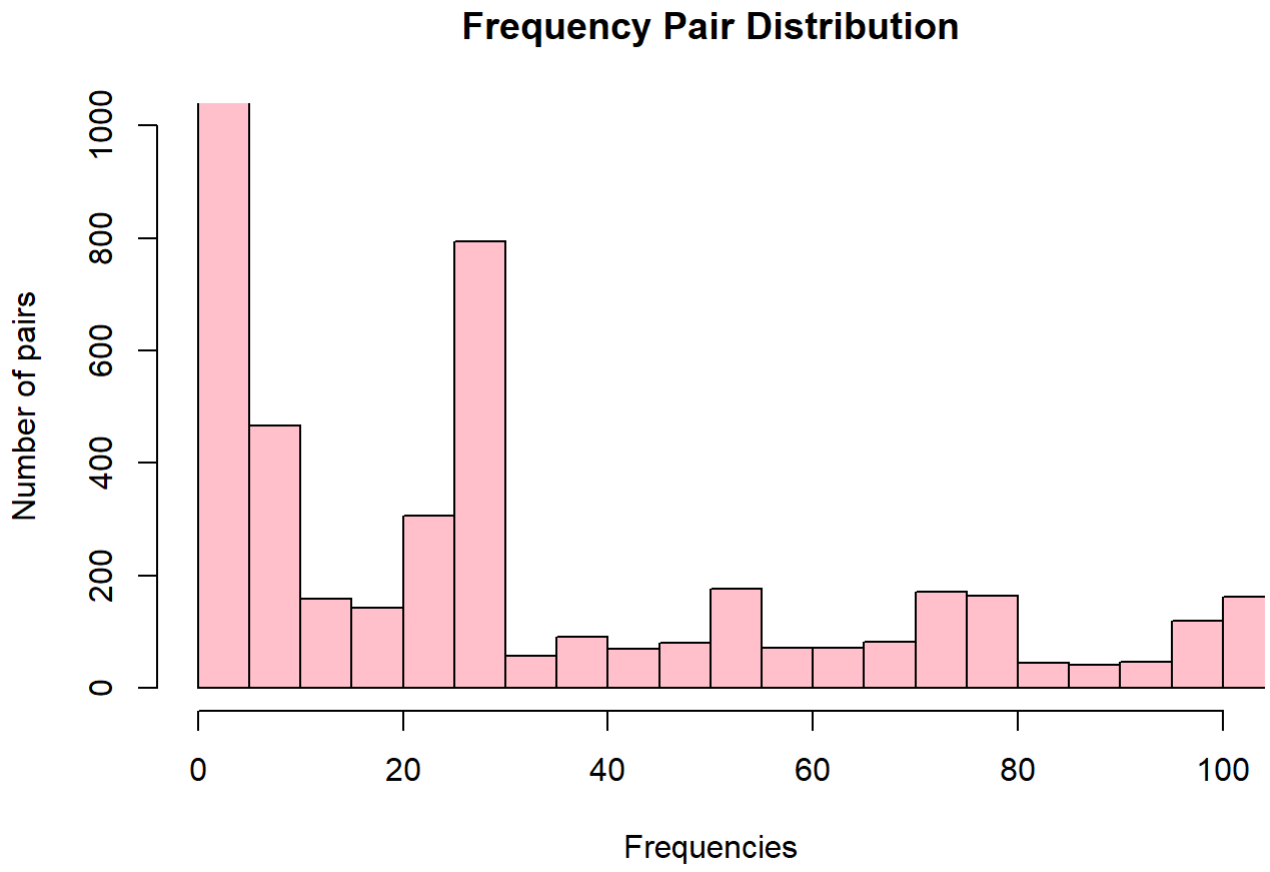
```
## Variance: 18496.9237982391
```

```
message ('Standard Deviation: ', sd(Buyer_Seller_Pair_Frequencies$Frequency))
```

```
## Standard Deviation: 136.003396274649
```

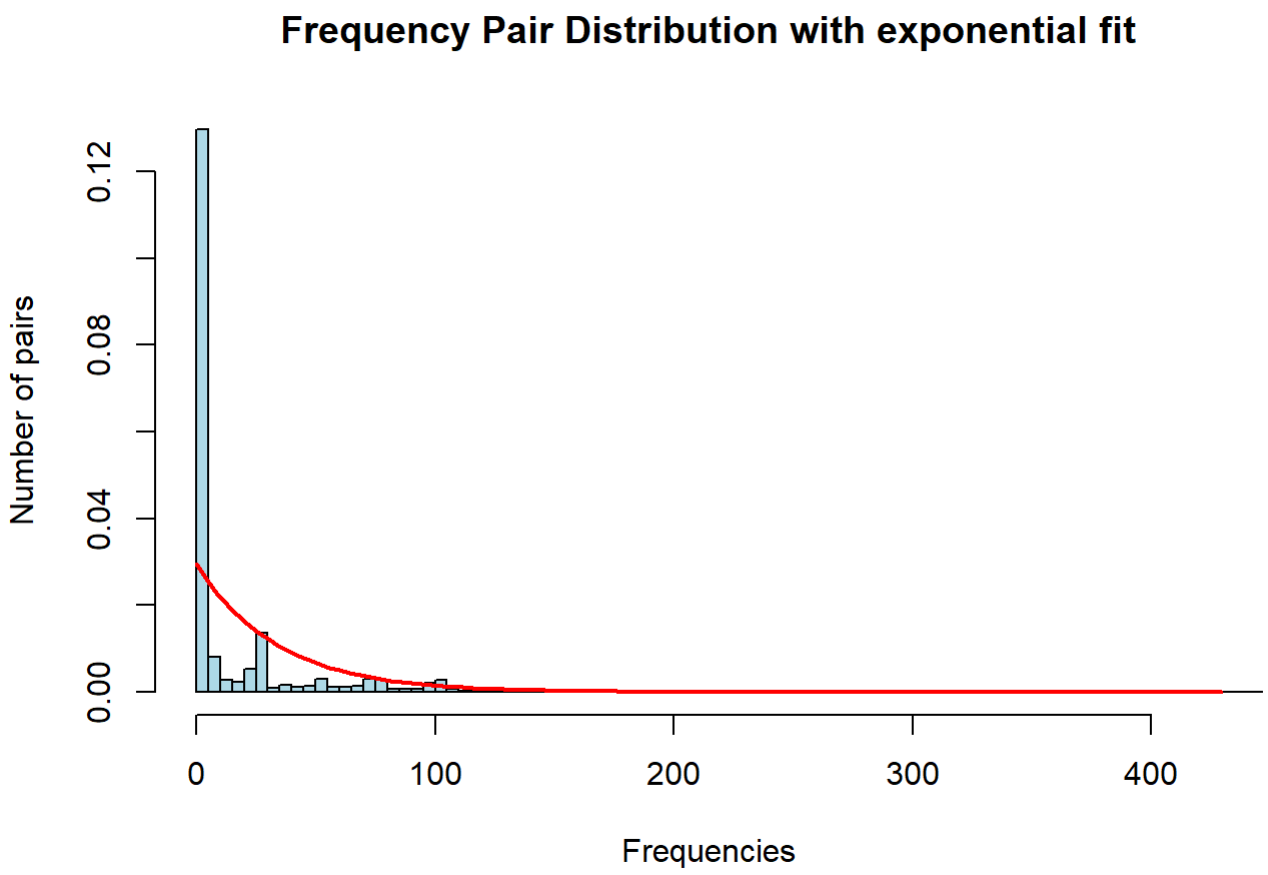
```
freq <- Buyer_Seller_Pair_Frequencies$Frequency
library(MASS)

h <- hist(freq, main = 'Frequency Pair Distribution',
     xlab = 'Frequencies', ylab = 'Number of pairs', col = 'pink',
     breaks = 1000, xlim = c(0, 100), ylim = c(0, 1000))
```



```
h <- hist(freq, main = 'Frequency Pair Distribution with exponential fit',
     xlab = 'Frequencies', ylab = 'Number of pairs', col = 'lightblue', freq = FALSE,
     breaks = 850, xlim = c(0, quantile(freq, 0.99)))

fit <- fitdistr(freq, 'exponential')
curve(dexp(x, rate = fit$estimate), from = 0, col = 'red', add = TRUE, lwd = 2)
```

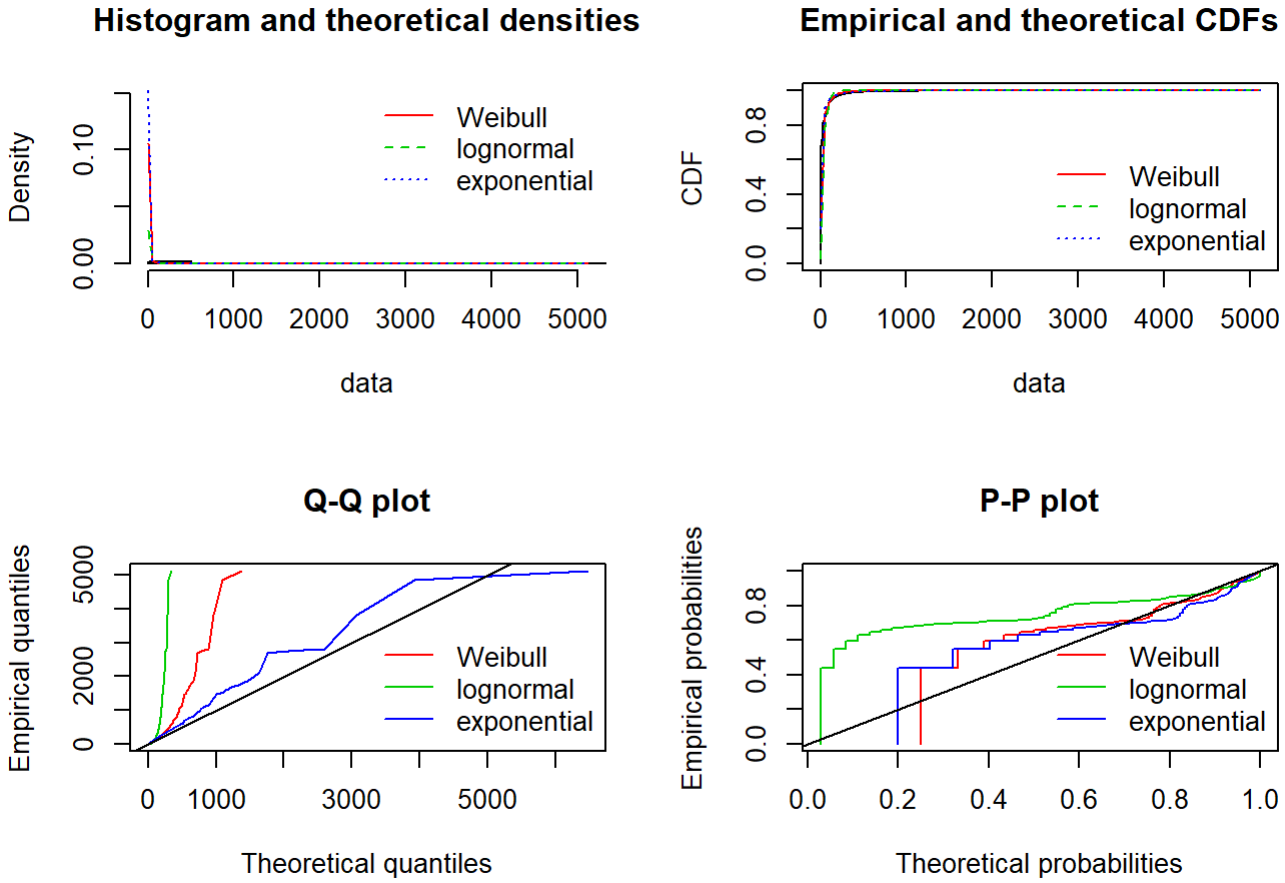



```
library(fitdistrplus)
```

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

```
fit_w  <- fitdist(freq, "weibull")
fit_ln <- fitdist(freq, "lnorm")
fit_ex <- fitdist(freq, "exp")

par(mfrow=c(2,2))
plot.legend <- c("Weibull", "lognormal", "exponential")
denscomp(list(fit_w, fit_ex, fit_ln), legendtext = plot.legend)
cdfcomp (list(fit_w, fit_ex, fit_ln), legendtext = plot.legend)
qqcomp  (list(fit_w, fit_ex, fit_ln), legendtext = plot.legend)
ppcomp  (list(fit_w, fit_ex, fit_ln), legendtext = plot.legend)
```



PART 2 BEGINS HERE

```
preprocessed_data$TokenAmount <- preprocessed_data$TokenAmount/10^18
Time <- as.Date(as.POSIXct(preprocessed_data$Timestamp, origin = '1970-01-01'))
preprocessed_data$Timestamp <- Time
preprocessed_data
```

	Buyer <int>	Seller <int>	Timestamp <date>	TokenAmount <dbl>
1	309659	9820517	2018-04-24	4.989940e-09
2	309659	9820518	2018-04-25	9.979994e-08
3	309659	9820519	2018-04-25	2.461906e-09
4	309659	9820520	2018-04-25	9.940000e-12
5	309659	9820517	2018-04-25	2.474413e-09
6	309659	9206179	2018-04-25	2.396602e-07
7	309659	9820521	2018-04-25	1.199994e-08
8	309659	9820522	2018-04-25	9.994000e-11
9	9206179	9820523	2018-04-25	2.000000e-09

	Buyer <int>	Seller <int>	Timestamp <date>	TokenAmount <dbl>
10	309659	9820524	2018-04-25	2.000040e-09

1-10 of 10,000 rows Previous **1** 2 3 4 5 6 … 1000 Next

```
library("readxl")
my_data <- read_excel("Veros_CoinMarketCap.xlsx")
colnames(my_data) <- c('Timestamp', 'Open', 'High', 'Low', 'Close', 'Volume', 'MarketCap')
my_data$Timestamp <- as.Date(my_data$Timestamp, "%d%B%Y")
```

```
## Warning in as.POSIXlt.POSIXct(x, tz = tz): unknown timezone '%d%B%Y'
```

```
my_data$MarketCap <- as.double(my_data$MarketCap)
my_data
```

Timestamp <date>	Open <dbl>	High <dbl>	Low <dbl>	Close <dbl>	Volume <chr>	MarketCap <dbl>
2019-05-03	0.006521	0.007001	0.006445	0.006462	3130	195297
2019-05-02	0.006469	0.006531	0.006206	0.006522	2749	197103
2019-05-01	0.006784	0.006970	0.006245	0.006470	15926	195519
2019-04-30	0.006363	0.006794	0.004815	0.006783	14516	204972
2019-04-29	0.006771	0.006927	0.005619	0.006445	18867	194773
2019-04-28	0.006836	0.006994	0.006705	0.006773	14020	204678
2019-04-27	0.006096	0.007031	0.006055	0.006754	32484	204112
2019-04-26	0.005618	0.007183	0.005513	0.006210	22706	187645
2019-04-25	0.006490	0.007334	0.005544	0.005597	16348	169135
2019-04-24	0.006924	0.007069	0.005771	0.006491	65912	196160

1-10 of 869 rows Previous **1** 2 3 4 5 6 … 87 Next

```
new_data <- preprocessed_data[order (- preprocessed_data$TokenAmount),]
new_data$Seller <- NULL
new_data
```

	Buyer <int>	Timestamp <date>	TokenAmount <dbl>
4055	9821909	2017-09-23	1.000000e+07
172247	9821354	2017-05-06	4.970000e+00
510	9820591	2018-04-29	1.707758e-02

	Buyer <int>	Timestamp <date>	TokenAmount <dbl>
353	9820591	2018-04-28	1.707757e-02
511	9820591	2018-04-29	1.707757e-02
515	9820591	2018-04-29	1.707757e-02
517	9820591	2018-04-29	1.707757e-02
534	9820591	2018-04-29	1.707757e-02
91	3692842	2018-04-27	4.500000e-03
361843	9822957	2017-01-14	6.052045e-04

1-10 of 10,000 rows Previous **1** 2 3 4 5 6 … 1000 Next

```
joined_df <- join(new_data, my_data)
```

```
## Joining by: Timestamp
```

```
joined_df <- na.omit(joined_df)
joined_df
```

	Buyer <int>	Timestamp <date>	TokenAmount <dbl>	Open <dbl>	High <dbl>	Low <dbl>	Close <dbl>	Volu… <chr>	MarketC… <dbl>
1	9821909	2017-09-23	1.000000e+07	0.000073	0.000109	0.000073	0.000099	2091	4838
2	9821354	2017-05-06	4.970000e+00	0.025338	0.025833	0.015454	0.015753	6322	126965
3	9820591	2018-04-29	1.707758e-02	0.052167	0.054266	0.038284	0.042727	20083	38945
4	9820591	2018-04-28	1.707757e-02	0.051992	0.055744	0.049829	0.051959	20090	47360
5	9820591	2018-04-29	1.707757e-02	0.052167	0.054266	0.038284	0.042727	20083	38945
6	9820591	2018-04-29	1.707757e-02	0.052167	0.054266	0.038284	0.042727	20083	38945
7	9820591	2018-04-29	1.707757e-02	0.052167	0.054266	0.038284	0.042727	20083	38945
8	9820591	2018-04-29	1.707757e-02	0.052167	0.054266	0.038284	0.042727	20083	38945
9	3692842	2018-04-27	4.500000e-03	0.056162	0.064861	0.051953	0.052111	19054	47499
10	9822957	2017-01-14	6.052045e-04	0.035403	0.050024	0.033510	0.033769	6941	71749

1-10 of 10,000 rows Previous **1** 2 3 4 5 6 … 1000 Next

```
joined_df$percentage <- (joined_df$TokenAmount/joined_df$MarketCap)*100
joined_df
```

	Buyer <int>	Timestamp <date>	TokenAmount <dbl>	Open <dbl>	High <dbl>	Low <dbl>	Close <dbl>	Volu… <chr>	MarketC.. <dbl>
1	9821909	2017-09-23	1.000000e+07	0.000073	0.000109	0.000073	0.000099	2091	4838
2	9821354	2017-05-06	4.970000e+00	0.025338	0.025833	0.015454	0.015753	6322	126965
3	9820591	2018-04-29	1.707758e-02	0.052167	0.054266	0.038284	0.042727	20083	38945
4	9820591	2018-04-28	1.707757e-02	0.051992	0.055744	0.049829	0.051959	20090	47360
5	9820591	2018-04-29	1.707757e-02	0.052167	0.054266	0.038284	0.042727	20083	38945
6	9820591	2018-04-29	1.707757e-02	0.052167	0.054266	0.038284	0.042727	20083	38945
7	9820591	2018-04-29	1.707757e-02	0.052167	0.054266	0.038284	0.042727	20083	38945
8	9820591	2018-04-29	1.707757e-02	0.052167	0.054266	0.038284	0.042727	20083	38945
9	3692842	2018-04-27	4.500000e-03	0.056162	0.064861	0.051953	0.052111	19054	47499
10	9822957	2017-01-14	6.052045e-04	0.035403	0.050024	0.033510	0.033769	6941	71749

1-10 of 10,000 rows | 1-10 of 11 columns Previous **1** 2 3 4 5 6 … 1000 Next

```
Top_Buyers <- subset(joined_df, percentage < 100)
track_k_buyers <- head(Top_Buyers,10)
nk <- (unique(track_k_buyers))
Top_Buyers
```

	Buyer <int>	Timestamp <date>	TokenAmount <dbl>	Open <dbl>	High <dbl>	Low <dbl>	Close <dbl>	Volu… <chr>	MarketC.. <dbl>
2	9821354	2017-05-06	4.970000e+00	0.025338	0.025833	0.015454	0.015753	6322	126965
3	9820591	2018-04-29	1.707758e-02	0.052167	0.054266	0.038284	0.042727	20083	38945
4	9820591	2018-04-28	1.707757e-02	0.051992	0.055744	0.049829	0.051959	20090	47360
5	9820591	2018-04-29	1.707757e-02	0.052167	0.054266	0.038284	0.042727	20083	38945
6	9820591	2018-04-29	1.707757e-02	0.052167	0.054266	0.038284	0.042727	20083	38945
7	9820591	2018-04-29	1.707757e-02	0.052167	0.054266	0.038284	0.042727	20083	38945
8	9820591	2018-04-29	1.707757e-02	0.052167	0.054266	0.038284	0.042727	20083	38945
9	3692842	2018-04-27	4.500000e-03	0.056162	0.064861	0.051953	0.052111	19054	47499
10	9822957	2017-01-14	6.052045e-04	0.035403	0.050024	0.033510	0.033769	6941	71749
11	9820565	2018-04-17	5.661065e-04	0.000080	0.000154	0.000073	0.000141	60428	6860

1-10 of 10,000 rows | 1-10 of 11 columns Previous **1** 2 3 4 5 6 … 1000 Next

```
message('The value of K is: ',nrow(count(nk)))
```

```
## The value of K is: 7
```

```
cor.test(track_k_buyers$TokenAmount, track_k_buyers$MarketCap, method = "pearson")
```

```
##
##  Pearson's product-moment correlation
##
## data:  track_k_buyers$TokenAmount and track_k_buyers$MarketCap
## t = 4.9049, df = 8, p-value = 0.001186
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.5206303 0.9679546
## sample estimates:
##       cor
## 0.8662859
```

```
cor.test(Top_Buyers$Open, Top_Buyers$High, method = "pearson")
```

```
##
##  Pearson's product-moment correlation
##
## data:  Top_Buyers$Open and Top_Buyers$High
## t = 3046, df = 377680, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.9801226 0.9803721
## sample estimates:
##       cor
## 0.9802477
```

```
cor.test(Top_Buyers$Close, Top_Buyers$High, method = "pearson")
```

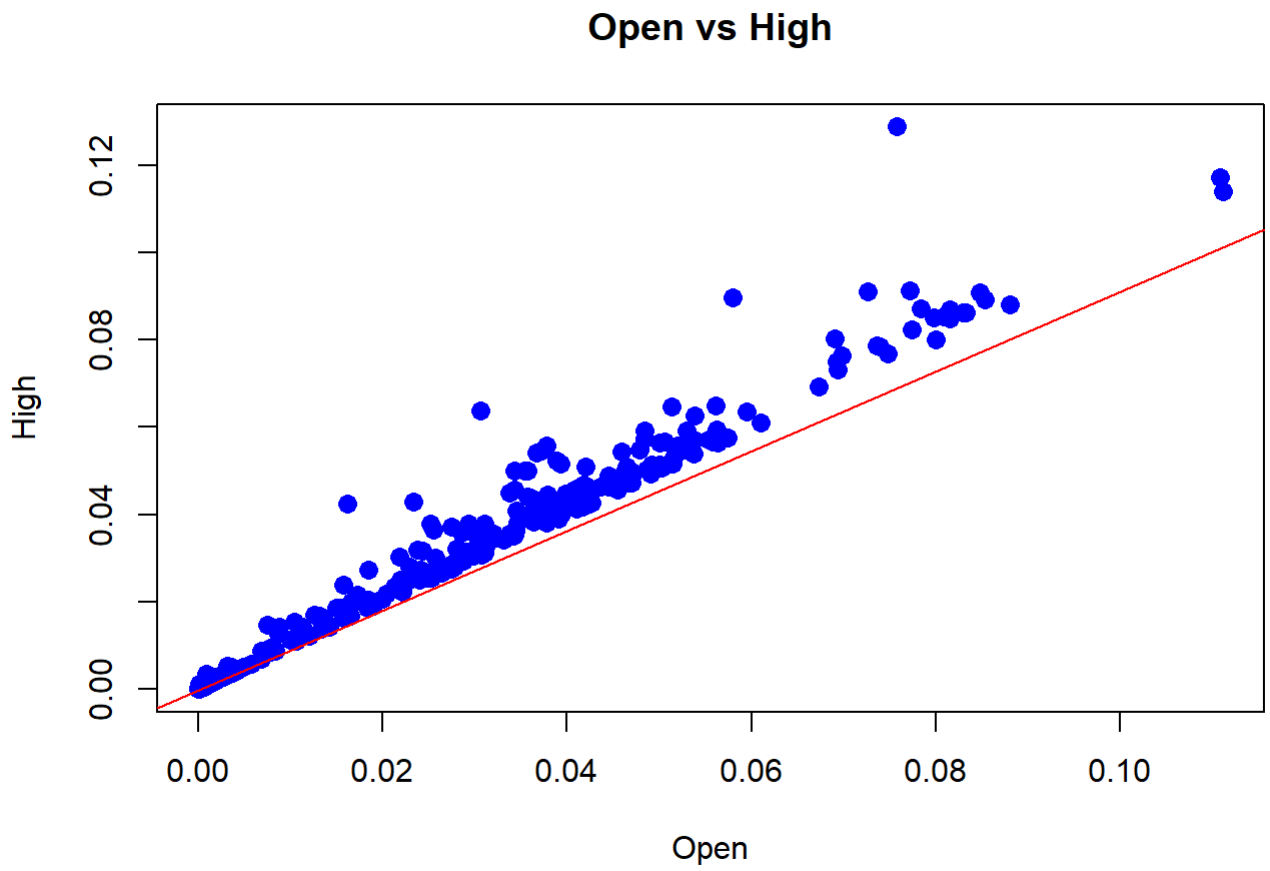
```
##
##  Pearson's product-moment correlation
##
## data:  Top_Buyers$Close and Top_Buyers$High
## t = 3283.4, df = 377680, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.9828230 0.9830389
## sample estimates:
##       cor
## 0.9829313
```

```
linearModOH <- lm(High ~ Open+Close, data=Top_Buyers)  # build linear regression model on full d
ata
linearModTM <- lm(MarketCap ~ TokenAmount, data=Top_Buyers)
summary(linearModTM)
```

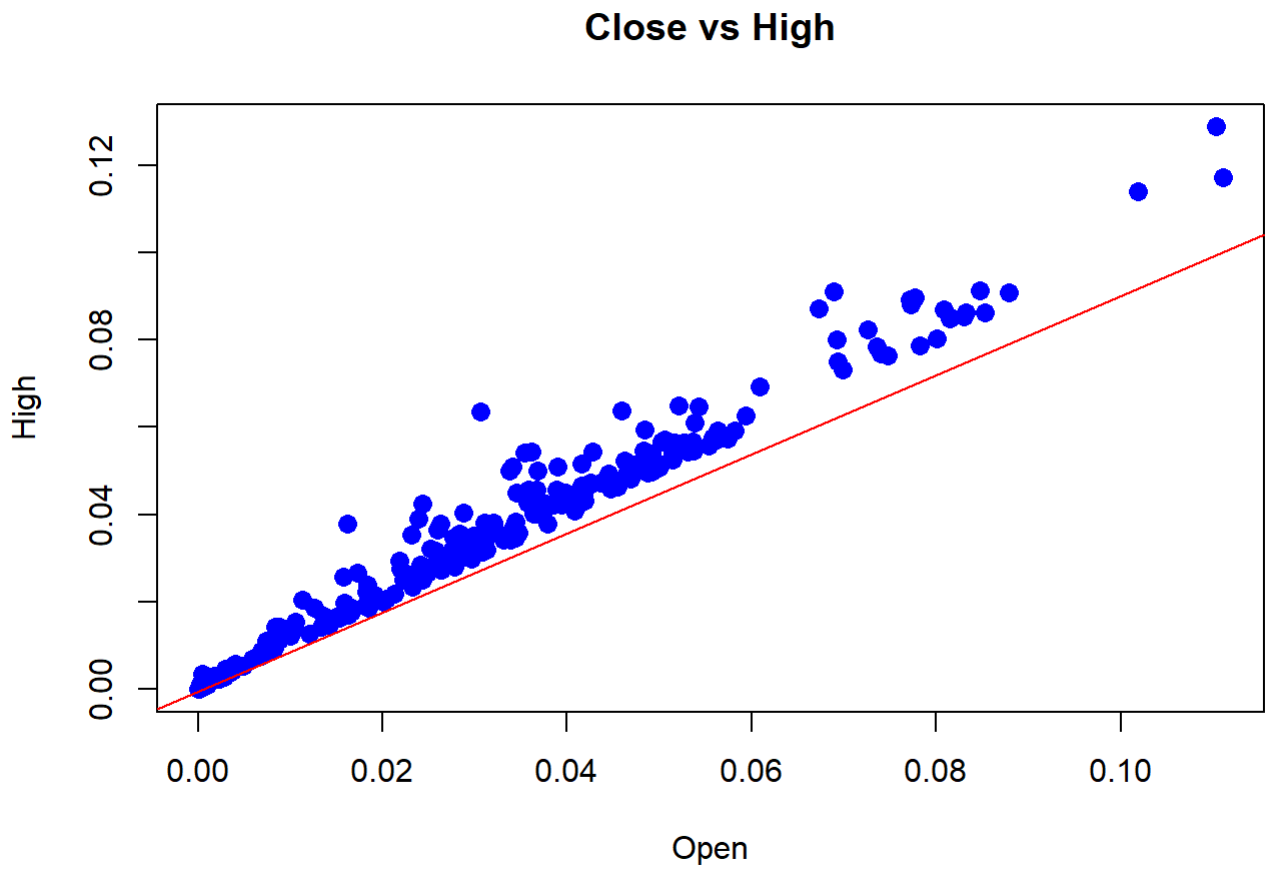
```
##
## Call:
## lm(formula = MarketCap ~ TokenAmount, data = Top_Buyers)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1364389  -638847    18757   658399  1816021
##
## Coefficients:
##             Estimate Std. Error  t value Pr(>|t|)
## (Intercept)  1369599       1312 1044.196   <2e-16 ***
## TokenAmount   -24592     162181   -0.152    0.879
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 806100 on 377676 degrees of freedom
## Multiple R-squared:  6.088e-08,  Adjusted R-squared:  -2.587e-06
## F-statistic: 0.02299 on 1 and 377676 DF,  p-value: 0.8795
```

```
modelSummary <- summary(linearModTM)  # capture model summary as an object
modelCoeffs <- modelSummary$coefficients  # model coefficients
```

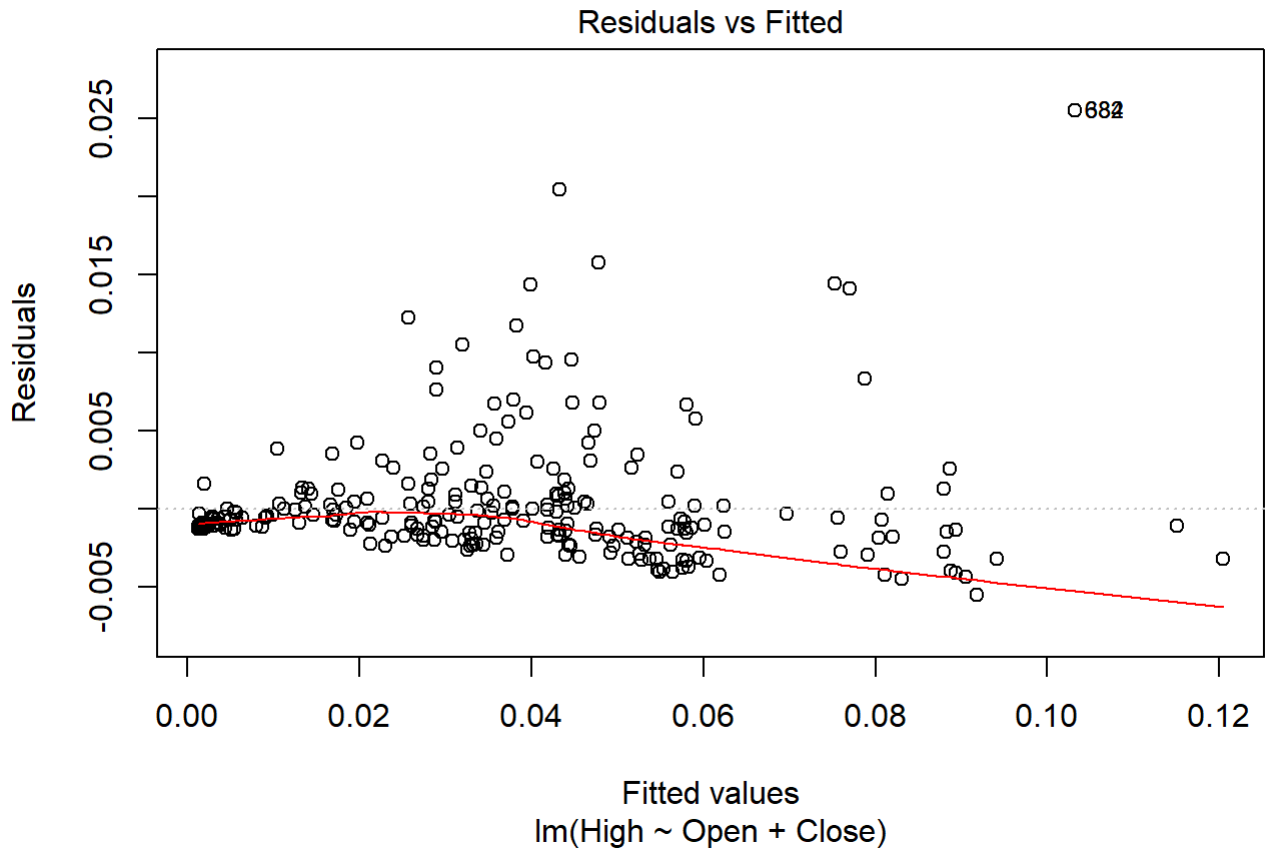
```
plot(Top_Buyers$Open, Top_Buyers$High, pch = 16, cex = 1.3, col = "blue", main = "Open vs High",
xlab = "Open", ylab = "High")
abline(lm(Top_Buyers$Open ~Top_Buyers$High), col = 'red')
```

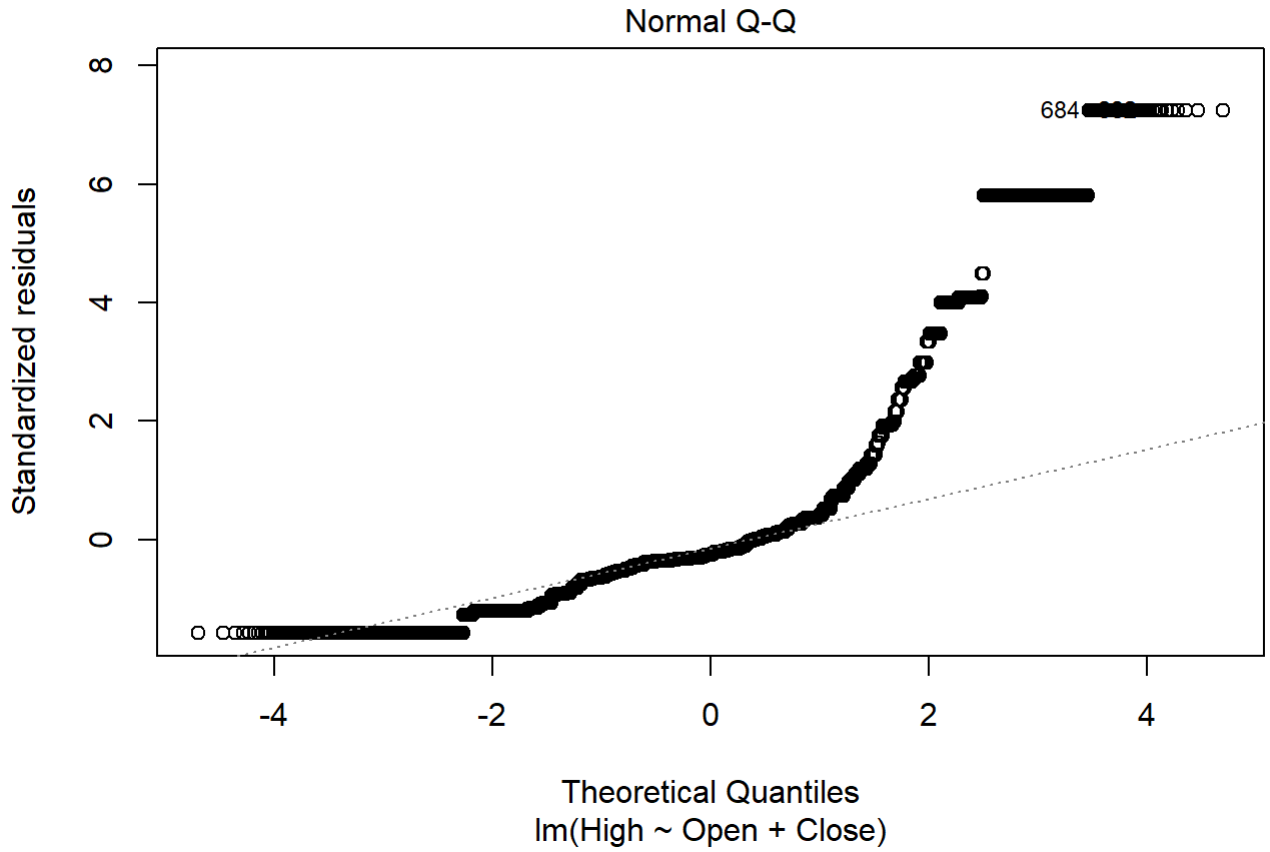


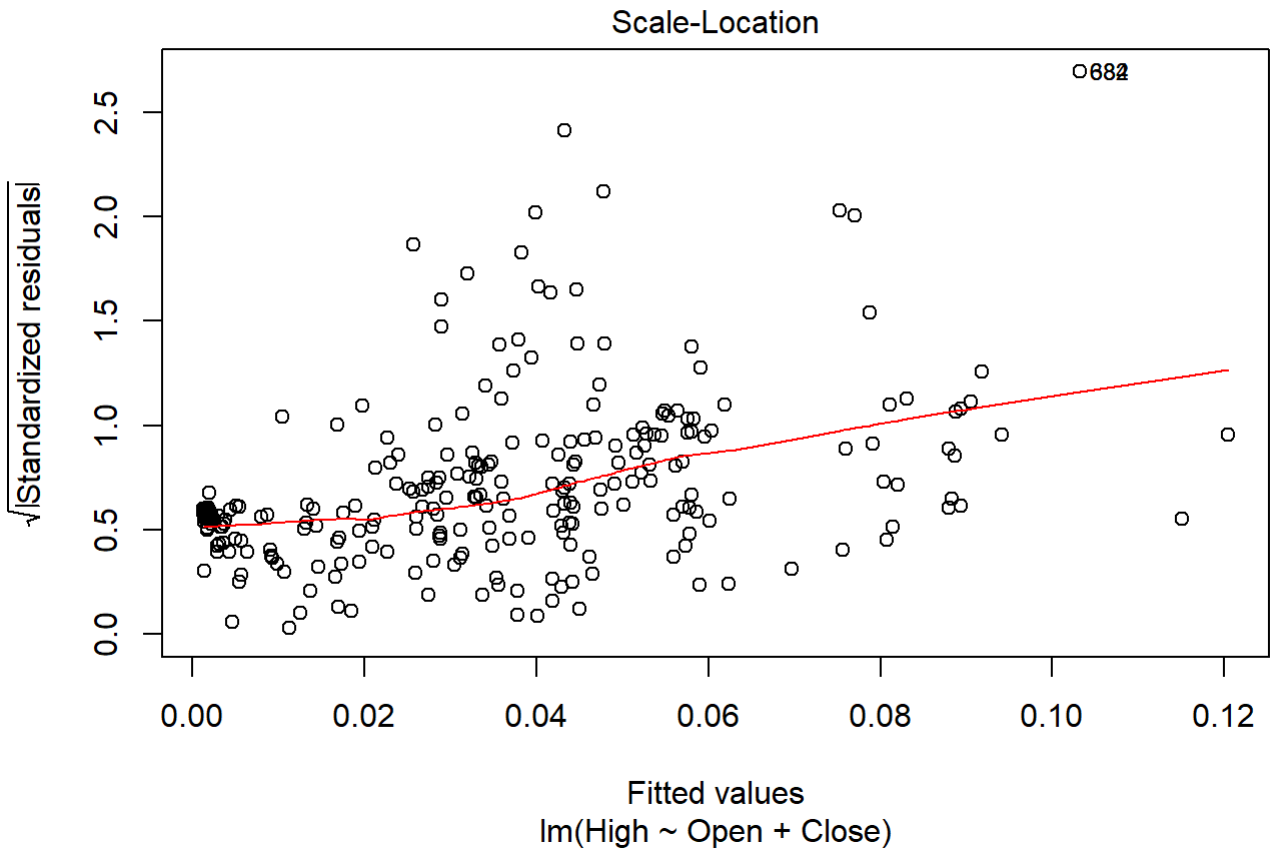

```
plot(Top_Buyers$Close, Top_Buyers$High, pch = 16, cex = 1.3, col = "blue", main = "Close vs Hig
h", xlab = "Open", ylab = "High")
abline(lm(Top_Buyers$Close ~Top_Buyers$High), col = 'red')
```

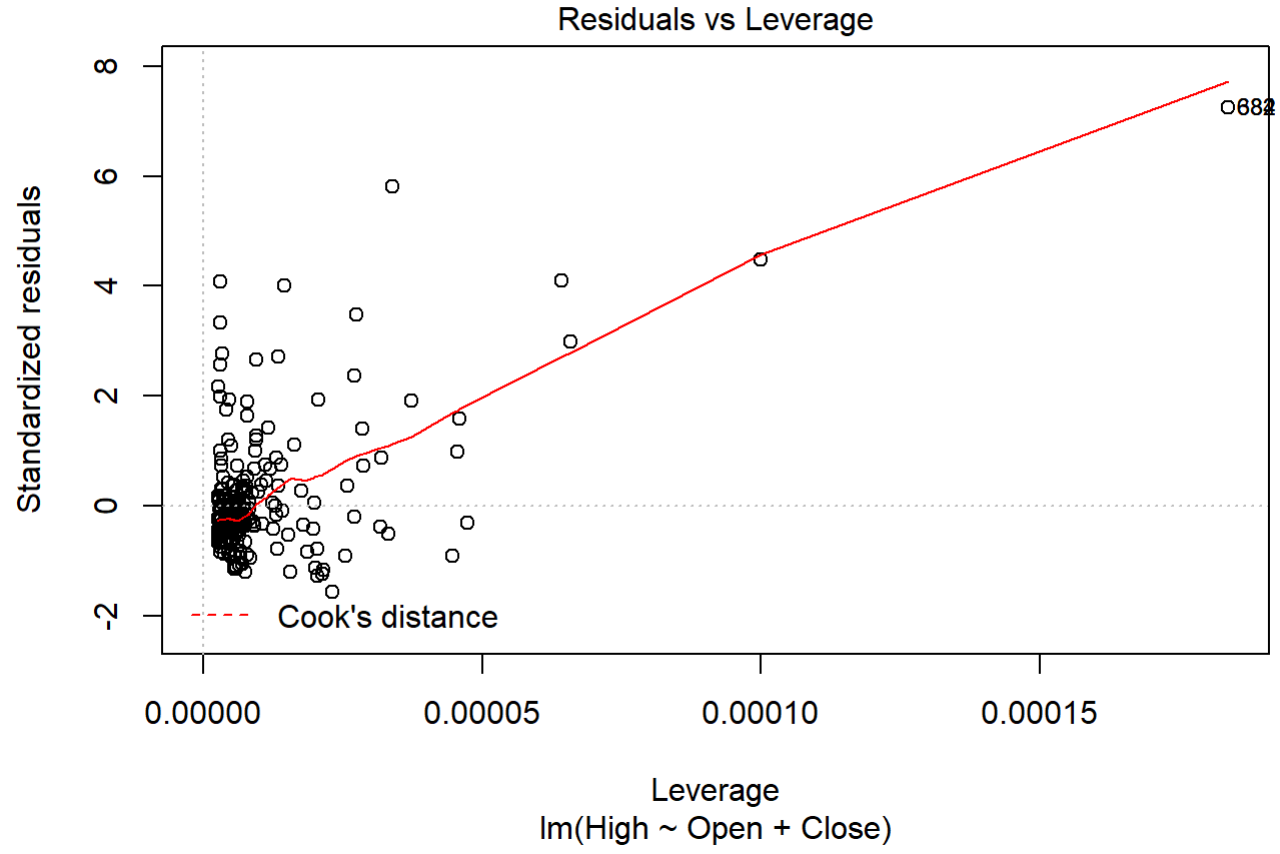



```
plot(linearModOH)
```

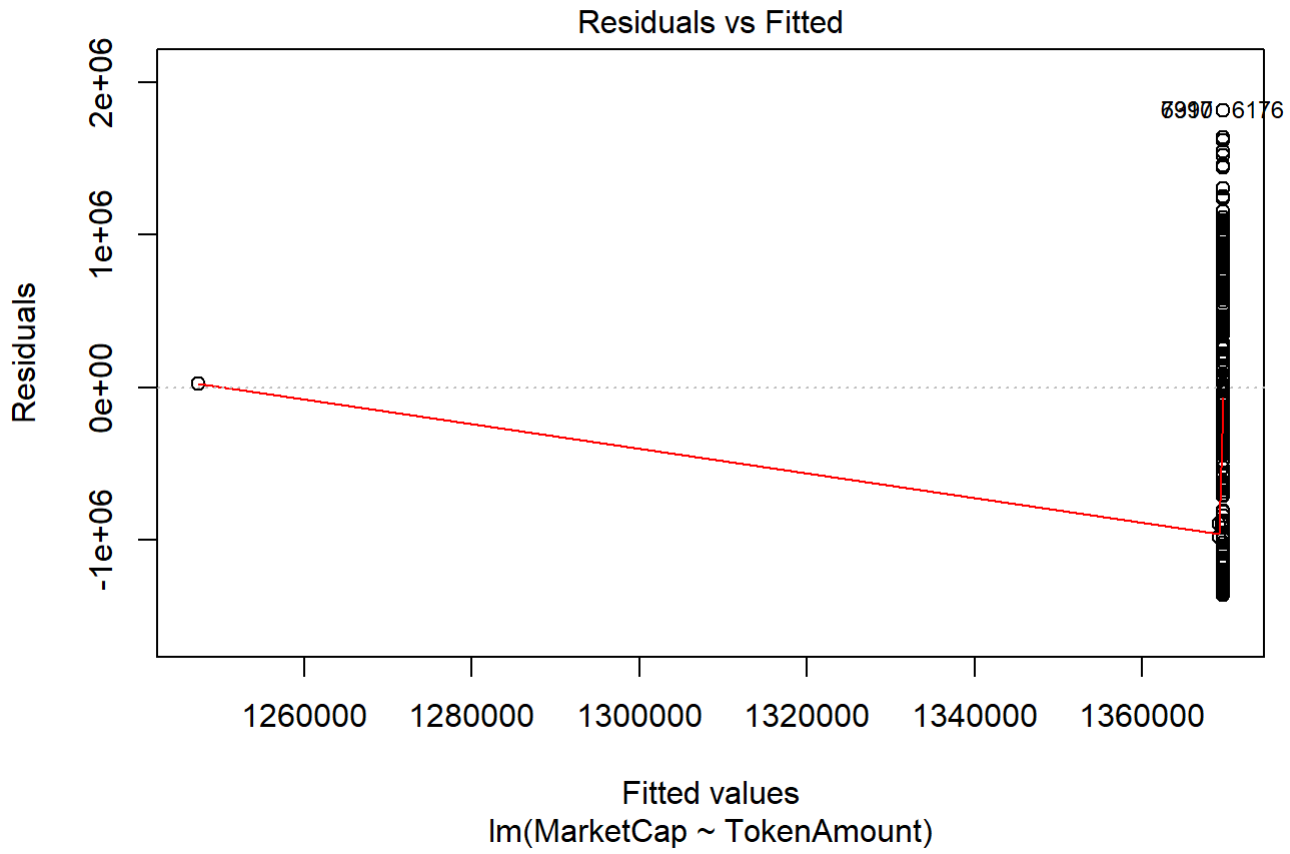

Residuals vs Fitted
Residuals
0.025
0.015
0.005
-0.005
0.00
0.02
0.04
0.06
0.08
0.10
0.12
Fitted values
lm(High ~ Open + Close)

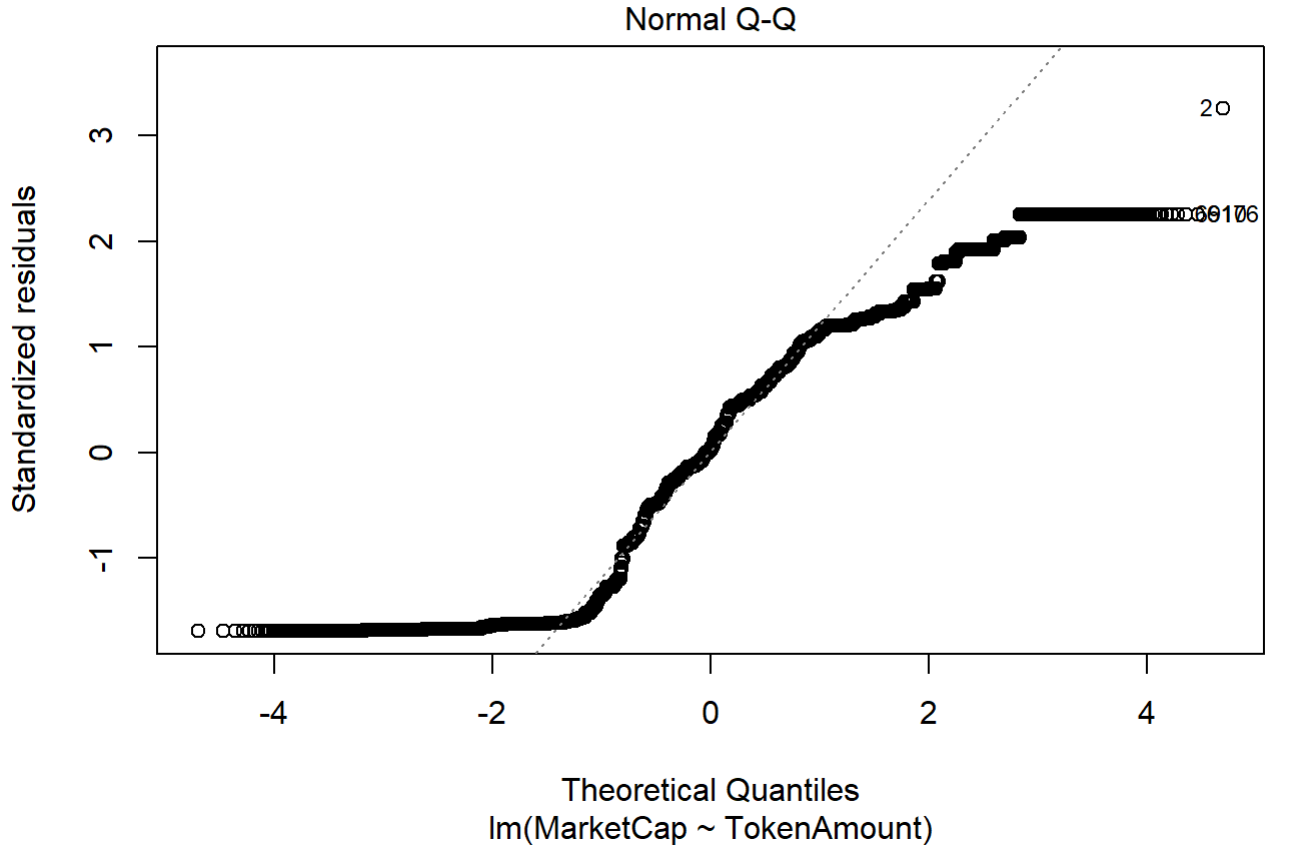

Normal Q-Q
Standardized residuals
8
6
4
2
0
684
-4
-2
0
2
4
Theoretical Quantiles
lm(High ~ Open + Close)

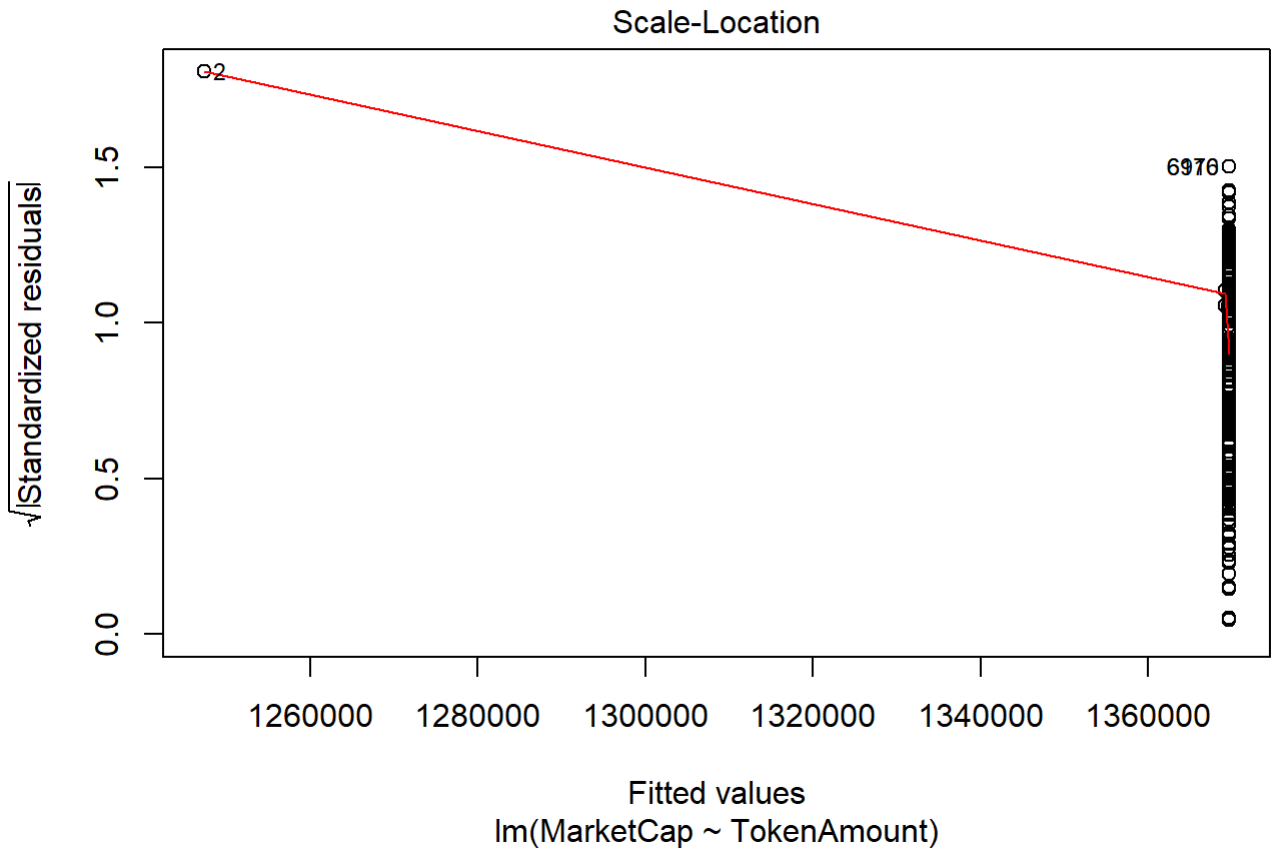

Scale-Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
2.5
0.00
0.02
0.04
0.06
0.08
0.10
0.12
Fitted values
lm(High ~ Open + Close)


Residuals vs Leverage
Standardized residuals
-2
0
2
4
6
8
Cook's distance
0.00000
0.00005
0.00010
0.00015
Leverage
lm(High ~ Open + Close)

```
plot(linearModTM)
```

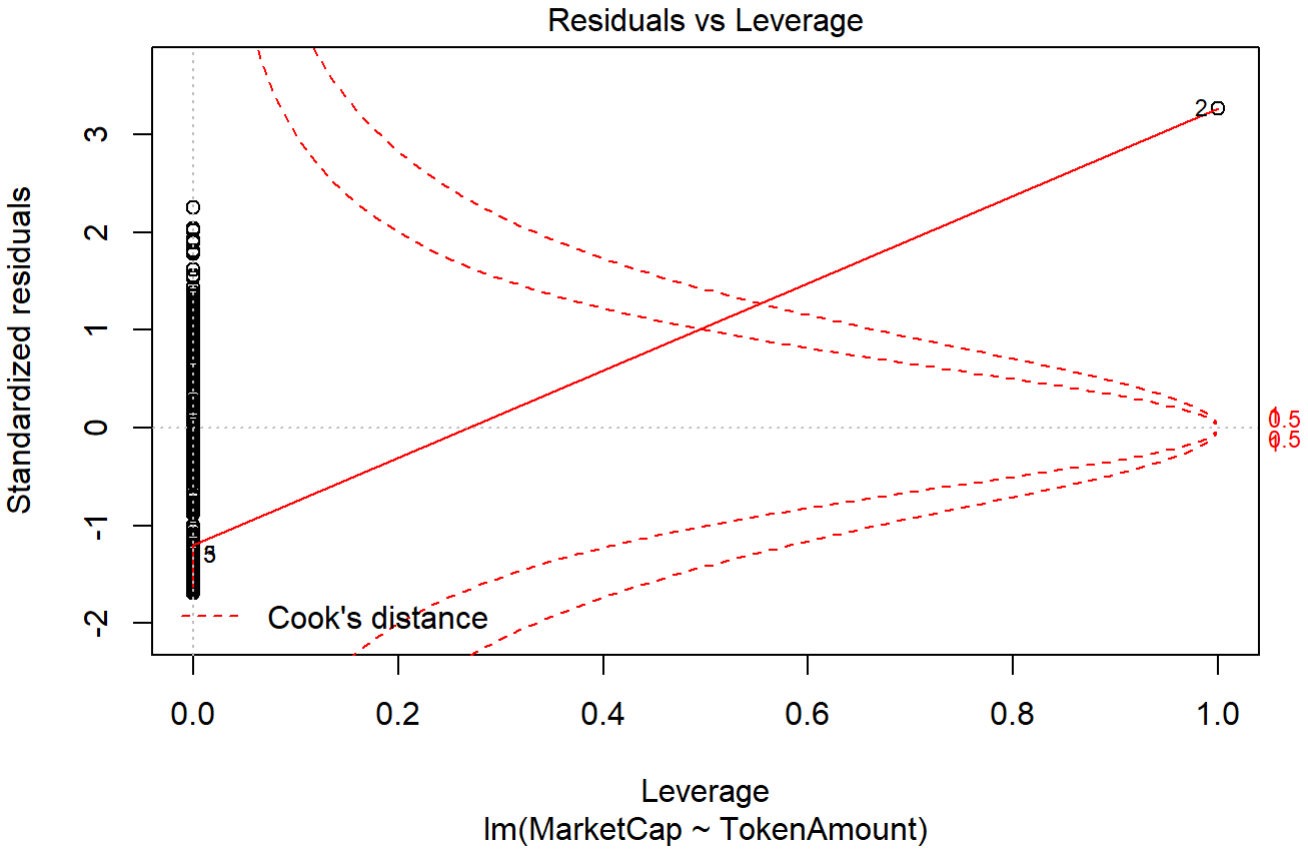

Residuals vs Fitted
Residuals
2e+06
1e+06
0e+00
-1e+06
1260000
1280000
1300000
1320000
1340000
1360000
Fitted values
lm(MarketCap ~ TokenAmount)


Normal Q-Q
Standardized residuals
3
2
1
0
-1
-4
-2
0
2
4
Theoretical Quantiles
lm(MarketCap ~ TokenAmount)




```
## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced

## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced
```


Residuals vs Leverage
Standardized residuals
3
2
1
0
-1
-2
2
5
0.5
0.5
Cook's distance
0.0
0.2
0.4
0.6
0.8
1.0
Leverage
lm(MarketCap ~ TokenAmount)