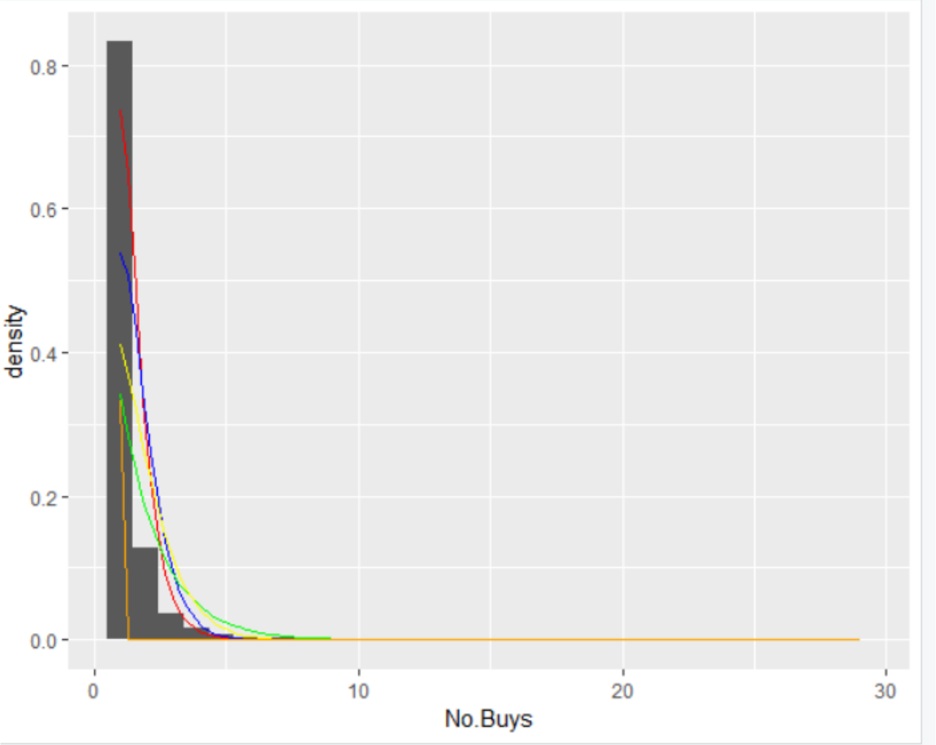
Name and NetID: Qi Gao (qxg150130) Pancham Mamania (pxm172730)

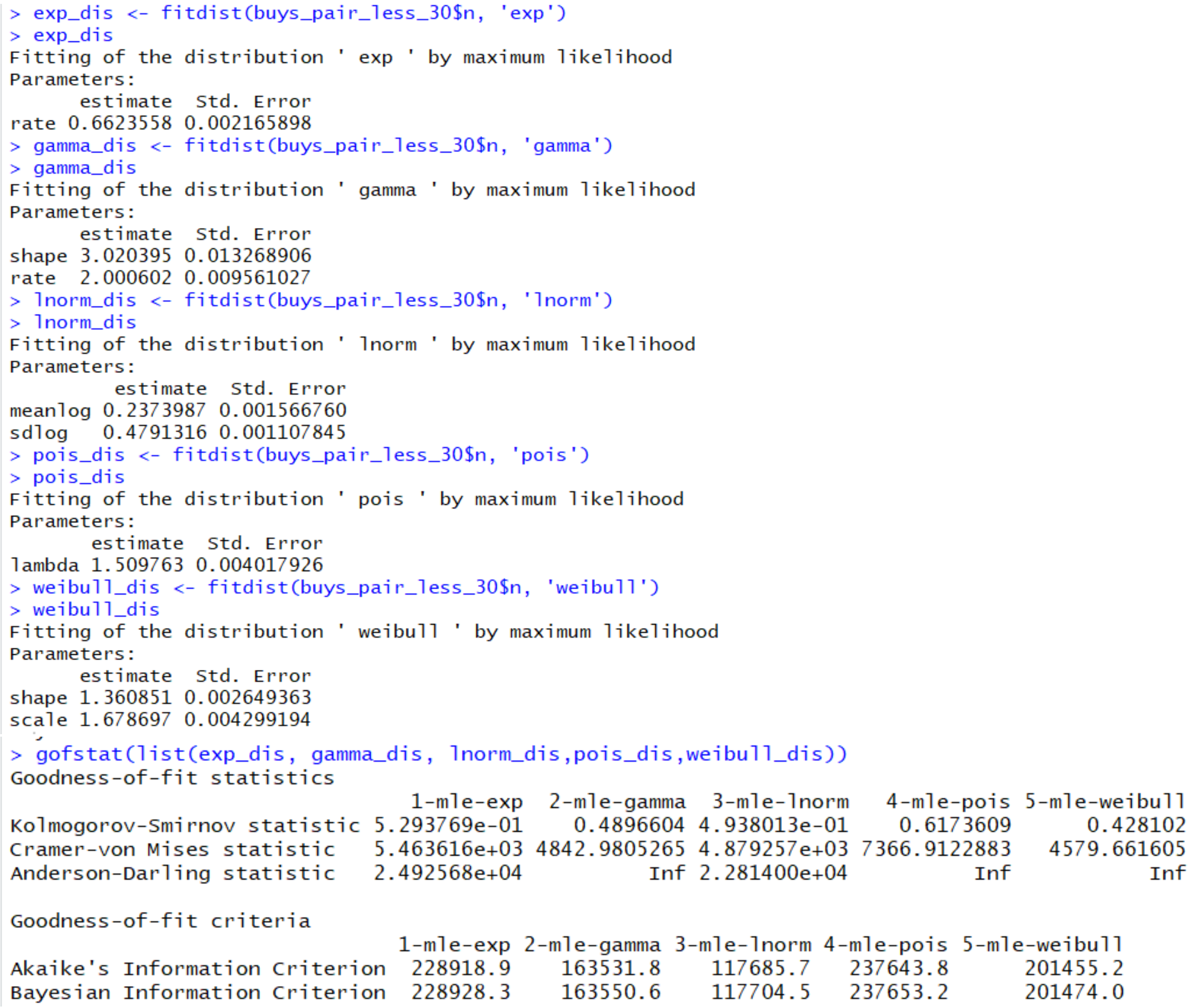
Tierion

```
library(magrittr)
library(dplyr)
library(ggplot2)
library(readr)
library(fitdistrplus)
library(DAAG)
library("ggplot2")
library(anytime)

tierion <- read_delim('C:/Users/ygaoq/OneDrive/MyDocuments/2019 Spring/Statistics/Project/Blockchain-Tokens-Data-Analytics/networktierionTX.txt', delim = " ", col_names = F)
names(tierion) <- c('fromID', 'toID', 'unixTime', 'tokenAmount')
decimals <- 10^8
supply <- 1 * 10^9
tierionFiltered <- filter(tierion,tokenAmount < decimals * supply)  #filter out all outliers

#figure out how many users indruced those unnormal transcation
tierion_outliers<- filter(tierion,tokenAmount >= decimals * supply)
user_outliers <- tierion_outliers %>% group_by(toID) %>% summarise(nn = n()) %>% ungroup
number_users_outliers<-nrow(user_outliers)
number_users_outliers

#get top X buyers data
buys<-tierionFiltered%>% group_by(toID) %>% summarise(nn = n()) %>% ungroup #change the supply and decimals amount
buys_sorted_dec<-buys[order(-buys$n),]
#top 30 active buyers and number of buys
top_30_buyers<-buys_sorted_dec%>%head(30)
top_30_buyers

########################################Question 1#######################################

#####group by user pairs#####
buys_pairs<-tierionFiltered%>% group_by(fromID, toID) %>% summarise(nn = n()) %>% ungroup
for (row in 1:nrow(buys_pairs)) {
  a<-buys_pairs[row,"fromID"]
  b<-buys_pairs[row,"toID"]
  for (inner_row in row:nrow(buys_pairs)) {
    c<-buys_pairs[inner_row,"fromID"]
    d<-buys_pairs[inner_row,"toID"]
    if(a==d&&b==c){
      buys_pairs[inner_row,"fromID"]<-d
      buys_pairs[inner_row,"toID"]<-c
    }
  }
}
```

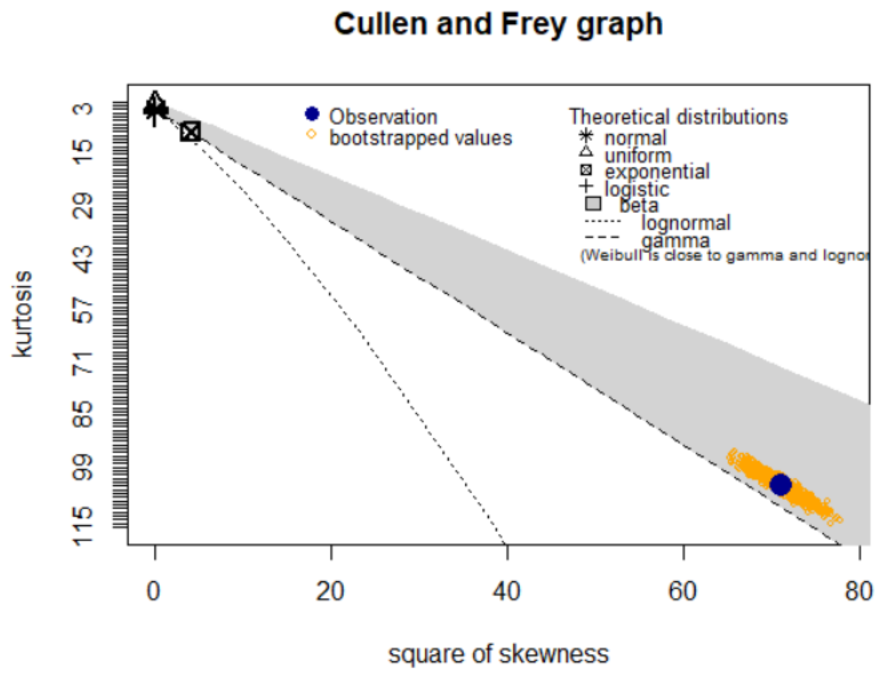
```r
buys_pairs<-tierionFiltered%>% group_by(fromID*toID+fromID+toID) %>% summarise(nn = n()) %>% ungroup
buys_pairs<-tierionFiltered%>% group_by(fromID, toID) %>% summarise(nn = n()) %>% ungroup
buys_pair_sorted_asc<-buys_pairs[order(buys_pairs$n),]
buys_pair_less_30<-subset(buys_pair_sorted_asc,n<30)

#####find out estimates of paramaters of several distribution based on the buys_pairs data set#####
exp_dis <- fitdist(buys_pair_less_30$n, 'exp')
exp_dis
gamma_dis <- fitdist(buys_pair_less_30$n, 'gamma')
gamma_dis
lnorm_dis <- fitdist(buys_pair_less_30$n, 'lnorm')
lnorm_dis
pois_dis <- fitdist(buys_pair_less_30$n, 'pois')
pois_dis
weibull_dis <- fitdist(buys_pair_less_30$n, 'weibull')
weibull_dis

gofstat(list(exp_dis, gamma_dis, lnorm_dis))
descdist(buys_sorted_asc$n,boot=1000)

#lognorm
fit_lnorm <- fitdist(buys_pair_less_30$n,"lnorm")
summary(fit_lnorm)
plot(fit_lnorm)
cdfcomp(fit_lnorm)

#exp
fit_exp <- fitdist(buys_pair_less_30$n,"exp")
summary(fit_exp)
plot(fit_exp)
cdfcomp(fit_exp)

#gamma
fit_gamma <- fitdist(buys_pair_less_30$n,"gamma")
summary(fit_gamma)
plot(fit_gamma)
cdfcomp(fit_gamma)

#weibull
fit_weibull <- fitdist(buys_pair_less_30$n,"weibull")
summary(fit_weibull)

#normal
fit_normal <- fitdist(buys_pair_less_30$n,"norm")
summary(fit_normal)

#pois
#normal
fit_pois <- fitdist(buys_pair_less_30$n,"pois")
summary(fit_pois)

#unif
fit_unif <- fitdist(buys_pair_less_30$n,"unif")
summary(fit_unif)
plot(fit_unif)
cdfcomp(fit_unif)

```

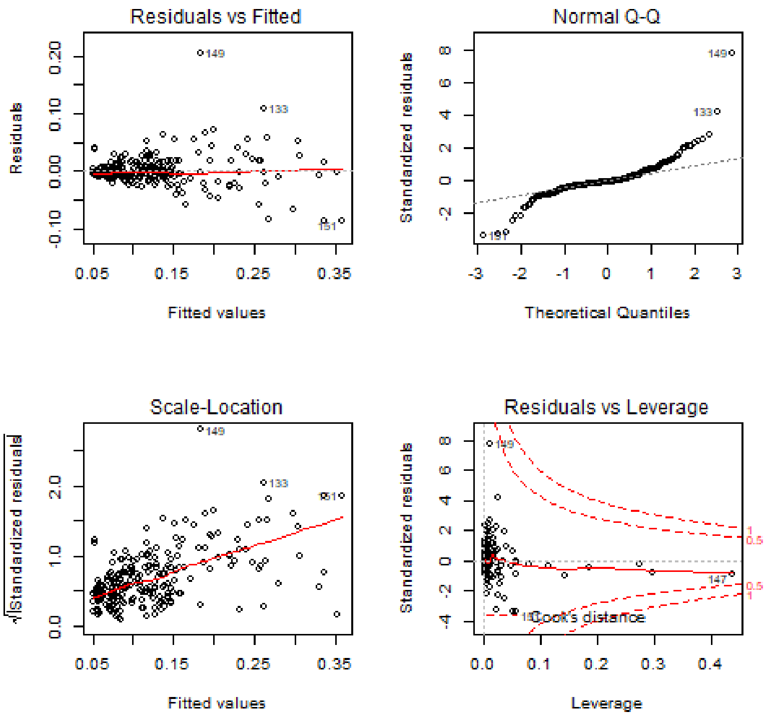
```
######################draw graph######################
all_density <- ggplot(data=buys_pair_less_30) +
  geom_histogram(bins=30,aes(x = buys_pair_less_30$n, ..density..)) +
  stat_function(fun = dlnorm, args = list(meanlog = 0.2373987, sdlog = 0.4791316),
                colour = "red")+
  stat_function(fun = dgamma, args = list(shape = 3.020395, rate=2.000602),
                colour = "blue")+
  stat_function(fun=dexp, args=list(rate=0.6623558),colour="green")+
  stat_function(fun=dweibull, args=list(shape=1.360851, scale=1.678697),colour="yellow")+
  stat_function(fun=dpois, args=list(lambda=1.509763),colour="orange")+  xlab("No.Buys")
all_density

########################################Question 1#############################################


########################################Question 2#############################################
tierion_prices <- read_delim("C:/Users/ygaoq/OneDrive/MyDocuments/2019 Spring/Statistics/Project/Blockchain-Tokens-Data-Analytics/tierion", delim = "\t", col_names = T) #load token price data
names(tierion_prices) <- make.names(names(tierion_prices))
tierion_prices <- tierion_prices %>% mutate(date = as.Date(Date, format = '%m/%d/%Y'))

tierion <- read_delim('C:/Users/ygaoq/OneDrive/MyDocuments/2019 Spring/Statistics/Project/Blockchain-Tokens-Data-Analytics/networktierionTX.txt', delim = " ", col_names = F)
names(tierion) <- c('fromID', 'toID', 'unixTime', 'tokenAmount')
decimals <- 10^8
supply <- 1 * 10^9
tierion_filtered <-filter(tierion,tokenAmount < decimals * supply)
## convert data type of unixTime
tierion_filtered <- tierion_filtered %>%
  mutate(date = anydate(unixTime))
names(tierion_filtered) <- c('fromID', 'toID', 'unixTime', 'tokenAmount', 'date')

## merge the prices and edge
tierion_merged<-merge(x = tierion_prices, y = tierion_filtered, by = "date", all.x = TRUE)

################Determin K##########################
top_30_buyers<-buys_sorted_dec%>%head(30)

top_K<-c(1:30)
count <- 1
for (val in top_K) {
  top_K_buyers<-buys_sorted_dec%>%head(val)
  filter_K_tierion_merged<-filter(tierion_merged,toID %in% top_K_buyers$toID)
  filter_K_tierion_merged=transform(filter_K_tierion_merged,average_price= (Open+Close)/2)
  filter_K_tierion_merged$num_Date <-  as.numeric(as.POSIXct(filter_K_tierion_merged$date))
  filered<-filter_K_tierion_merged%>% group_by(num_Date) %>% summarise(nn = n(),Close=mean(Close),tokenAmount=sum(tokenAmount),Open=mean(Open))
  shift <- function(x, n){
    c(x[-(seq(n))], rep(NA, n))
  }
  filered$new_Close<-shift(filered$Close,1)
  num_rows<-nrow(filered)
  filered[-num_rows,]
  regression<-lm(filered$new_Close~filered$tokenAmount+filered$n+filered$Open)

  setwd("C:/Users/ygaoq/Desktop/Tierion")
  yourfilename=paste("W",val,".txt",sep="")
  capture.output(summary(regression),append = TRUE,file = "C:/Users/ygaoq/Desktop/Tierion/Final_Result.txt")


  summary(regression)
  par(mfcol=c(2,2))
  setwd("C:/Users/ygaoq/Desktop/Tierion")
  yourfilename=paste("A",val,".png",sep="")
  png(file=yourfilename)
  opar <- par(mfrow=c(2,2))
  plot(regression)
  dev.off()
}
```

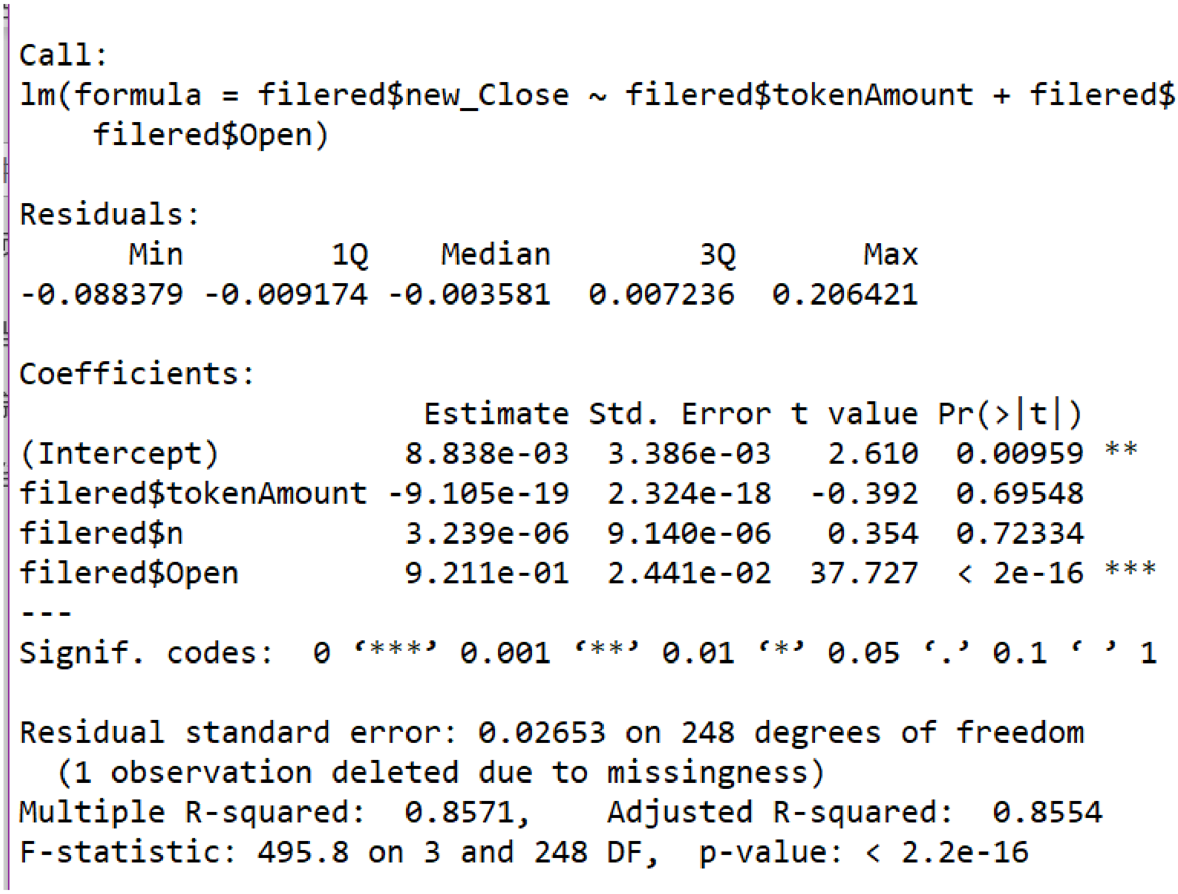




```
> exp_dis <- fitdist(buys_pair_less_30$n, 'exp')
> exp_dis
Fitting of the distribution ' exp ' by maximum likelihood
Parameters:
      estimate  Std. Error
rate 0.6623558 0.002165898
> gamma_dis <- fitdist(buys_pair_less_30$n, 'gamma')
> gamma_dis
Fitting of the distribution ' gamma ' by maximum likelihood
Parameters:
      estimate  Std. Error
shape 3.020395 0.013268906
rate  2.000602 0.009561027
> lnorm_dis <- fitdist(buys_pair_less_30$n, 'lnorm')
> lnorm_dis
Fitting of the distribution ' lnorm ' by maximum likelihood
Parameters:
         estimate  Std. Error
meanlog 0.2373987 0.001566760
sdlog   0.4791316 0.001107845
> pois_dis <- fitdist(buys_pair_less_30$n, 'pois')
> pois_dis
Fitting of the distribution ' pois ' by maximum likelihood
Parameters:
       estimate  Std. Error
lambda 1.509763 0.004017926
> weibull_dis <- fitdist(buys_pair_less_30$n, 'weibull')
> weibull_dis
Fitting of the distribution ' weibull ' by maximum likelihood
Parameters:
      estimate  Std. Error
shape 1.360851 0.002649363
scale 1.678697 0.004299194
> gofstat(list(exp_dis, gamma_dis, lnorm_dis,pois_dis,weibull_dis))
Goodness-of-fit statistics
                                1-mle-exp  2-mle-gamma  3-mle-lnorm   4-mle-pois 5-mle-weibull
Kolmogorov-Smirnov statistic 5.293769e-01    0.4896604 4.938013e-01    0.6173609      0.428102
Cramer-von Mises statistic   5.463616e+03 4842.9805265 4.879257e+03 7366.9122883   4579.661605
Anderson-Darling statistic   2.492568e+04          Inf 2.281400e+04          Inf           Inf

Goodness-of-fit criteria
                               1-mle-exp 2-mle-gamma 3-mle-lnorm 4-mle-pois 5-mle-weibull
Akaike's Information Criterion  228918.9    163531.8    117685.7   237643.8      201455.2
Bayesian Information Criterion  228928.3    163550.6    117704.5   237653.2      201474.0
```


Cullen and Frey graph
Observation
bootstrapped values
Theoretical distributions
normal
uniform
exponential
logistic
beta
lognormal
gamma
(Weibull is close to gamma and lognor
kurtosis
3
15
29
43
57
71
85
99
115
0
20
40
60
80
square of skewness


Residuals vs Fitted
Residuals
0.20
0.10
0.00
-0.10
149
133
151
0.05
0.15
0.25
0.35
Fitted values
Normal Q-Q
Standardized residuals
8
6
4
2
0
-2
149
133
151
-3
-2
-1
0
1
2
3
Theoretical Quantiles
Scale-Location
√|Standardized residuals|
2.0
1.0
0.0
149
133
151
0.05
0.15
0.25
0.35
Fitted values
Residuals vs Leverage
Standardized residuals
8
6
4
2
0
-2
-4
149
147
151
1
0.5
0.5
1
Cook's distance
0.0
0.1
0.2
0.3
0.4
Leverage



```
Call:
lm(formula = filered$new_Close ~ filered$tokenAmount + filered$n + 
    filered$Open)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.088379 -0.009174 -0.003581  0.007236  0.206421 

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)    
(Intercept)          8.838e-03  3.386e-03   2.610  0.00959 ** 
filered$tokenAmount -9.105e-19  2.324e-18  -0.392  0.69548    
filered$n            3.239e-06  9.140e-06   0.354  0.72334    
filered$Open         9.211e-01  2.441e-02  37.727  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.02653 on 248 degrees of freedom
  (1 observation deleted due to missingness)
Multiple R-squared:  0.8571,	Adjusted R-squared:  0.8554 
F-statistic: 495.8 on 3 and 248 DF,  p-value: < 2.2e-16
```

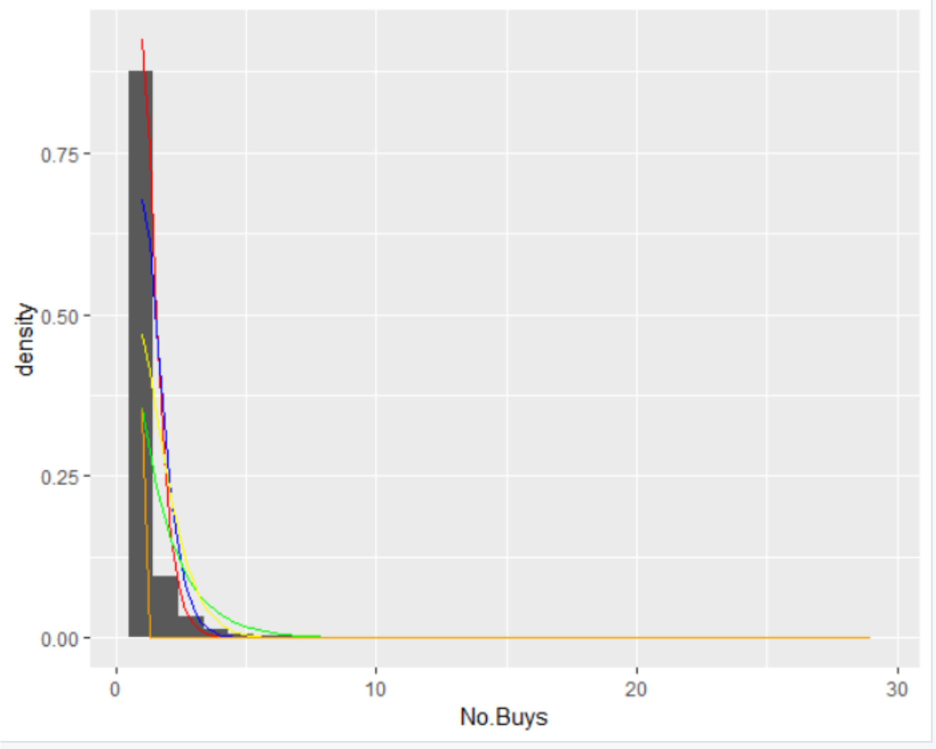
Aragon

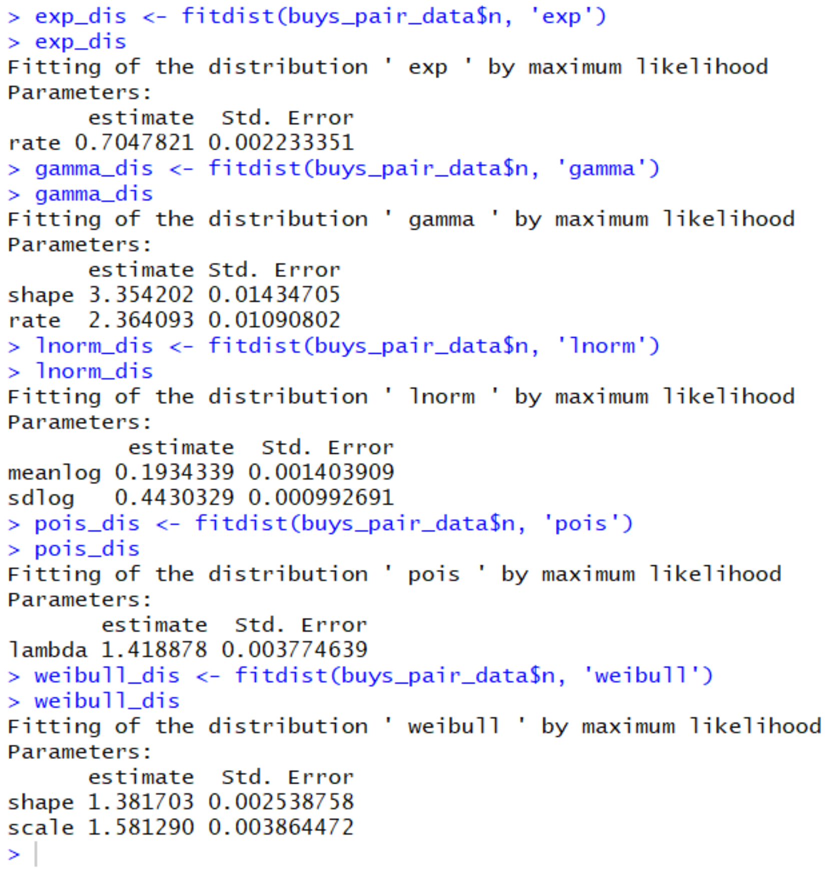
```
library(magrittr)
library(dplyr)
library(ggplot2)
library(readr)
library(fitdistrplus)
library(DAAG)
library("ggplot2")
library(anytime)

aragon <- read_delim('C:/Users/ygaoq/OneDrive/MyDocuments/2019 Spring/Statistics/Project/Blockchain-Tokens-Data-Analytics/networkaragonTX.txt', delim = " ", col_names = F)
names(aragon) <- c('fromID', 'toID', 'unixTime', 'tokenAmount')
decimals <- 10^18
supply <- 39609523
aragonFiltered <- filter(aragon,tokenAmount < decimals * supply)  #filter out all outliers

```

```r
#figure out how many users indruced those unnormal transcation
aragon_outliers<- filter(aragon,tokenAmount >= decimals * supply)
user_outliers <- aragon_outliers %>% group_by(toID) %>% summarise(nn = n()) %>% ungroup

number_users_outliers<-nrow(user_outliers)
number_users_outliers

#get top X buyers data
buys<-aragonFiltered%>% group_by(toID) %>% summarise(nn = n()) %>% ungroup #change the supply and decimals amount
buys_sorted_dec<-buys[order(-buys$n),]
#top 30 active buyers and number of buys
top_30_buyers<-buys_sorted_dec%>%head(30)
top_30_buyers

########################################Question 1############################################

#####group by user pairs#####
buys_pairs<-aragonFiltered%>% group_by(fromID, toID) %>% summarise(nn = n()) %>% ungroup
for (row in 1:nrow(buys_pairs)) {
  a<-buys_pairs[row,"fromID"]
  b<-buys_pairs[row,"toID"]
  for (inner_row in row:nrow(buys_pairs)) {
    c<-buys_pairs[inner_row,"fromID"]
    d<-buys_pairs[inner_row,"toID"]
    if(a==d&&b==c){
      buys_pairs[inner_row,"fromID"]<-d
      buys_pairs[inner_row,"toID"]<-c
    }
  }
}
buys_pairs<-aragonFiltered%>% group_by(fromID*toID+fromID+toID) %>% summarise(nn = n()) %>% ungroup

buys_pairs<-aragonFiltered%>% group_by(fromID, toID) %>% summarise(nn = n()) %>% ungroup
buys_pair_sorted_asc<-buys_pairs[order(buys_pairs$n),]
buys_pair_less_30<-subset(buys_pair_sorted_asc,n<30)
buys_pair_data<-buys_pair_less_30

#####find out estimates of paramaters of several distribution based on the buys_pairs data set#####
exp_dis <- fitdist(buys_pair_data$n, 'exp')
exp_dis
gamma_dis <- fitdist(buys_pair_data$n, 'gamma')
gamma_dis
lnorm_dis <- fitdist(buys_pair_data$n, 'lnorm')
lnorm_dis
pois_dis <- fitdist(buys_pair_data$n, 'pois')
pois_dis
weibull_dis <- fitdist(buys_pair_data$n, 'weibull')
weibull_dis

gofstat(list(exp_dis, gamma_dis, lnorm_dis))
descdist(buys_sorted_asc$n,boot=1000)

#lognorm
fit_lnorm <- fitdist(buys_pair_less_30$n,"lnorm")
```

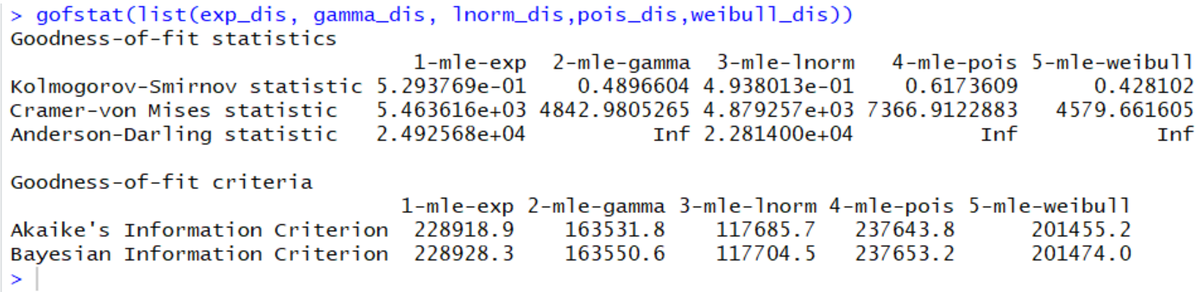
```r
summary(fit_lnorm)
plot(fit_lnorm)
cdfcomp(fit_lnorm)

#exp
fit_exp <- fitdist(buys_pair_less_30$n,"exp")
summary(fit_exp)
plot(fit_exp)
cdfcomp(fit_exp)

#gamma
fit_gamma <- fitdist(buys_pair_less_30$n,"gamma")
summary(fit_gamma)
plot(fit_gamma)
cdfcomp(fit_gamma)

#weibull
fit_weibull <- fitdist(buys_pair_less_30$n,"weibull")
summary(fit_weibull)

#normal
fit_normal <- fitdist(buys_pair_less_30$n,"norm")
summary(fit_normal)

#pois
#normal
fit_pois <- fitdist(buys_pair_less_30$n,"pois")
summary(fit_pois)

#unif
fit_unif <- fitdist(buys_pair_less_30$n,"unif")
summary(fit_unif)
plot(fit_unif)
cdfcomp(fit_unif)

######################draw graph#######################
all_density <- ggplot(data=buys_pair_less_30) +
  geom_histogram(bins=30,aes(x = buys_pair_less_30$n, ..density..)) +
  stat_function(fun = dlnorm, args = list(meanlog = 0.1537565, sdlog = 0.4012176),
                colour = "red")+
  stat_function(fun = dgamma, args = list(shape = 4.034757, rate=3.040920),
                colour = "blue")+
  stat_function(fun=dexp, args=list(rate=0.7536976),colour="green")+
  stat_function(fun=dweibull, args=list(shape=1.464309, scale=1.488009),colour="yellow")+
  stat_function(fun=dpois, args=list(lambda=1.326792),colour="orange")+
  xlab("No.Buys")
all_density

########################################Question 1############################################


########################################Question 2############################################
aragon_prices <- read_delim("C:/Users/ygaoq/OneDrive/MyDocuments/2019 Spring/Statistics/Project/Blockchain-Tokens-Data-Analytics/aragon", delim = "\t", col_names = T) #load token price data
names(aragon_prices) <- make.names(names(aragon_prices))
```

```r
aragon_prices <- aragon_prices %>% mutate(date = as.Date(Date, format = '%m/%d/%Y'))

aragon <- read_delim('C:/Users/ygaoq/OneDrive/MyDocuments/2019 Spring/Statistics/Project/Blockchain-Tokens-Data-Analytics/networkaragonTX.txt', delim = " ", col_names = F)
names(aragon) <- c('fromID', 'toID', 'unixTime', 'tokenAmount')
decimals <- 10^18
supply <- 39609523
aragon_filtered <-filter(aragon,tokenAmount < decimals * supply)
## convert data type of unixTime
aragon_filtered <- aragon_filtered %>%
  mutate(date = anydate(unixTime))
names(aragon_filtered) <- c('fromID', 'toID', 'unixTime', 'tokenAmount', 'date')

## merge the prices and edge
aragon_merged<-merge(x = aragon_prices, y = aragon_filtered, by = "date", all.x = TRUE)

################Determin K##########################
top_30_buyers<-buys_sorted_dec%>%head(30)

top_K<-c(1:30)
count <- 1
for (val in top_K) {
  top_K_buyers<-buys_sorted_dec%>%head(val)
  filter_K_aragon_merged<-filter(aragon_merged,toID %in% top_K_buyers$toID)
  filter_K_aragon_merged=transform(filter_K_aragon_merged,average_price= (Open+Close)/2)
  filter_K_aragon_merged$num_Date <-  as.numeric(as.POSIXct(filter_K_aragon_merged$date))
  filered<-filter_K_aragon_merged%>% group_by(num_Date) %>% summarise(nn = n(),Close=mean(Close),tokenAmount=sum(tokenAmount),Open=mean(Open))
  shift <- function(x, n){
    c(x[-(seq(n))], rep(NA, n))
  }
  filered$new_Close<-shift(filered$Close,1)
  num_rows<-nrow(filered)
  filered[-num_rows,]
  regression<-lm(filered$new_Close~filered$tokenAmount+filered$n+filered$Open)

  setwd("C:/Users/ygaoq/Desktop/aragon")
  yourfilename=paste("W",val,".txt",sep="")
  capture.output(summary(regression),append = TRUE,file = "C:/Users/ygaoq/Desktop/aragon/Final_Result.txt")


  summary(regression)
  par(mfcol=c(2,2))
  setwd("C:/Users/ygaoq/Desktop/aragon")
  yourfilename=paste("A",val,".png",sep="")
  png(file=yourfilename)
  opar <- par(mfrow=c(2,2))
  plot(regression)
  dev.off()
}
```

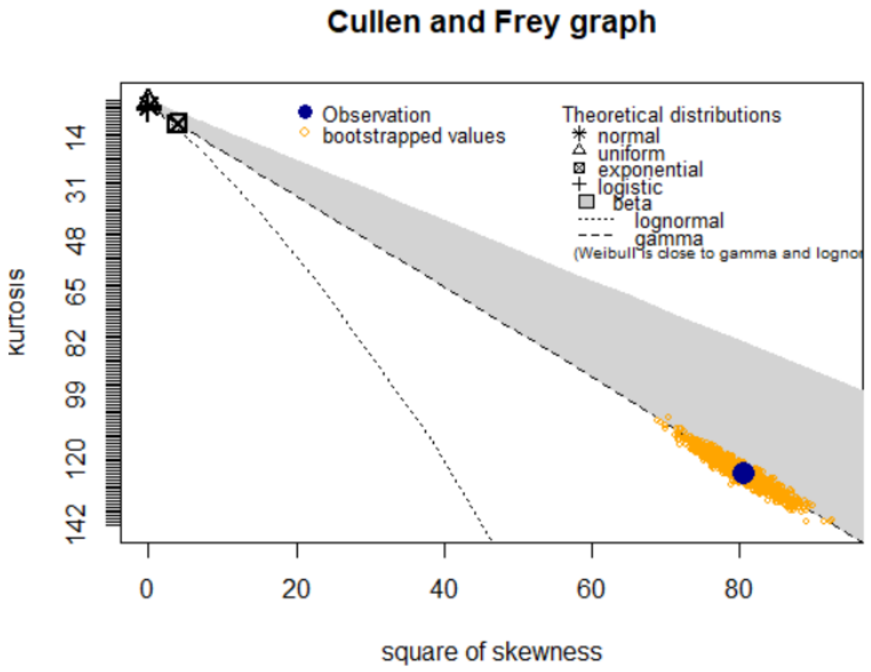


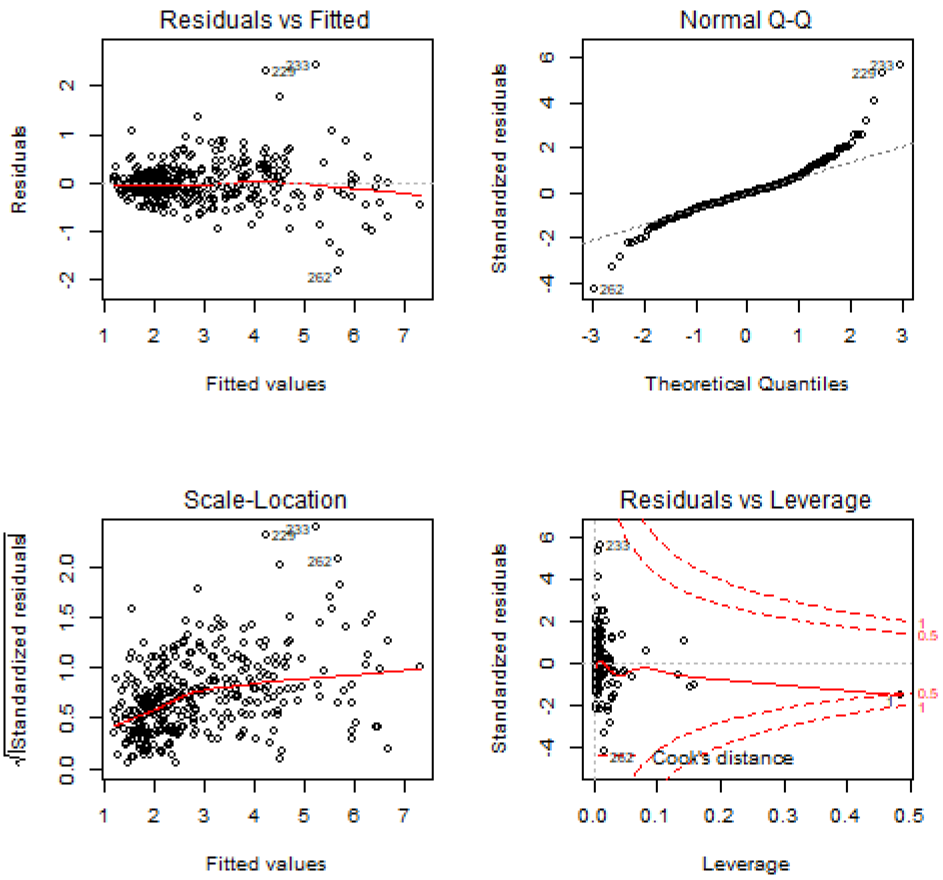


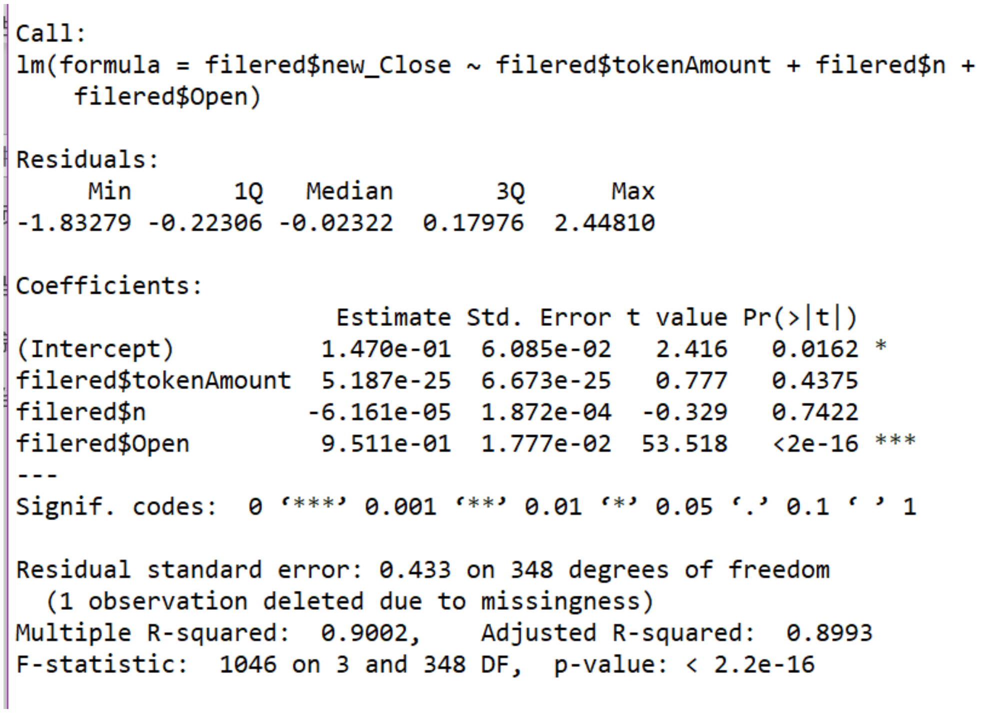
```
> exp_dis <- fitdist(buys_pair_data$n, 'exp')
> exp_dis
Fitting of the distribution ' exp ' by maximum likelihood
Parameters:
      estimate  Std. Error
rate 0.7047821 0.002233351
> gamma_dis <- fitdist(buys_pair_data$n, 'gamma')
> gamma_dis
Fitting of the distribution ' gamma ' by maximum likelihood
Parameters:
      estimate Std. Error
shape 3.354202 0.01434705
rate  2.364093 0.01090802
> lnorm_dis <- fitdist(buys_pair_data$n, 'lnorm')
> lnorm_dis
Fitting of the distribution ' lnorm ' by maximum likelihood
Parameters:
         estimate  Std. Error
meanlog 0.1934339 0.001403909
sdlog   0.4430329 0.000992691
> pois_dis <- fitdist(buys_pair_data$n, 'pois')
> pois_dis
Fitting of the distribution ' pois ' by maximum likelihood
Parameters:
       estimate  Std. Error
lambda 1.418878 0.003774639
> weibull_dis <- fitdist(buys_pair_data$n, 'weibull')
> weibull_dis
Fitting of the distribution ' weibull ' by maximum likelihood
Parameters:
      estimate  Std. Error
shape 1.381703 0.002538758
scale 1.581290 0.003864472
> 
```



```
> gofstat(list(exp_dis, gamma_dis, lnorm_dis,pois_dis,weibull_dis))
Goodness-of-fit statistics
                                1-mle-exp  2-mle-gamma  3-mle-lnorm   4-mle-pois 5-mle-weibull
Kolmogorov-Smirnov statistic 5.293769e-01    0.4896604 4.938013e-01    0.6173609      0.428102
Cramer-von Mises statistic   5.463616e+03 4842.9805265 4.879257e+03 7366.9122883   4579.661605
Anderson-Darling statistic   2.492568e+04          Inf 2.281400e+04          Inf           Inf

Goodness-of-fit criteria
                               1-mle-exp 2-mle-gamma 3-mle-lnorm 4-mle-pois 5-mle-weibull
Akaike's Information Criterion  228918.9    163531.8    117685.7   237643.8      201455.2
Bayesian Information Criterion  228928.3    163550.6    117704.5   237653.2      201474.0
> 
```







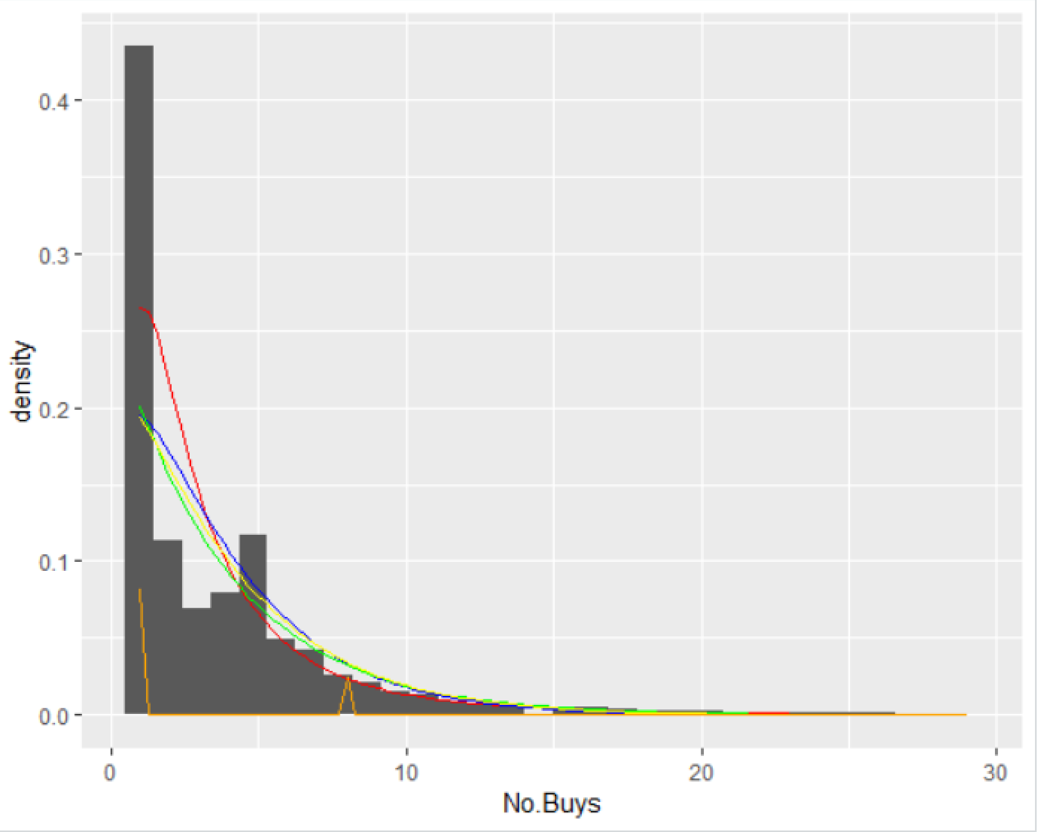
```
Call:
lm(formula = filered$new_Close ~ filered$tokenAmount + filered$n +
    filered$Open)

Residuals:
     Min       1Q   Median       3Q      Max
-1.83279 -0.22306 -0.02322  0.17976  2.44810

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)          1.470e-01  6.085e-02   2.416   0.0162 *
filered$tokenAmount  5.187e-25  6.673e-25   0.777   0.4375
filered$n           -6.161e-05  1.872e-04  -0.329   0.7422
filered$Open         9.511e-01  1.777e-02  53.518   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.433 on 348 degrees of freedom
  (1 observation deleted due to missingness)
Multiple R-squared:  0.9002,    Adjusted R-squared:  0.8993
F-statistic:  1046 on 3 and 348 DF,  p-value: < 2.2e-16
```

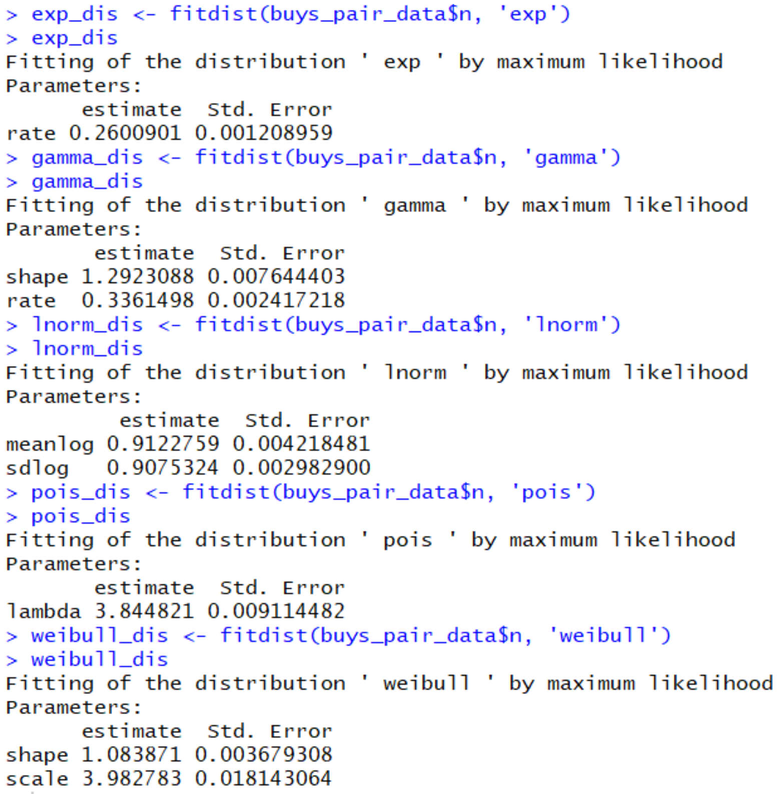
Bitqy

```
library(magrittr)
library(dplyr)
library(ggplot2)
library(readr)
library(fitdistrplus)
library(DAAG)
library("ggplot2")
library(anytime)

bitqy <- read_delim('C:/Users/ygaoq/OneDrive/MyDocuments/2019 Spring/Statistics/Project/Blockchain-Tokens-Data-Analytics/networkbitqyTX.txt', delim = " ", col_names = F)
names(bitqy) <- c('fromID', 'toID', 'unixTime', 'tokenAmount')
decimals <- 2
supply <- 1 * 10^10
bitqyFiltered <- filter(bitqy,tokenAmount < decimals * supply)  #filter out all outliers

#figure out how many users indruced those unnormal transcation
bitqy_outliers<- filter(bitqy,tokenAmount >= decimals * supply)
user_outliers <- bitqy_outliers %>% group_by(toID) %>% summarise(nn = n()) %>% ungroup

number_users_outliers<-nrow(user_outliers)
number_users_outliers

#get top X buyers data
buys<- bitqyFiltered%>% group_by(toID) %>% summarise(nn = n()) %>% ungroup #change the supply and decimals amount
buys_sorted_dec<-buys[order(-buys$n),]
#top 30 active buyers and number of buys
top_30_buyers<-buys_sorted_dec%>%head(30)
top_30_buyers

########################################Question 1############################################

#####group by user pairs#####
buys_pairs<- bitqyFiltered%>% group_by(fromID, toID) %>% summarise(nn = n()) %>% ungroup
for (row in 1:nrow(buys_pairs)) {
  a<-buys_pairs[row,"fromID"]
  b<-buys_pairs[row,"toID"]
  for (inner_row in row:nrow(buys_pairs)) {
    c<-buys_pairs[inner_row,"fromID"]
    d<-buys_pairs[inner_row,"toID"]
    if(a==d&&b==c){
      buys_pairs[inner_row,"fromID"]<-d
      buys_pairs[inner_row,"toID"]<-c
    }
  }
}
buys_pairs<- bitqyFiltered%>% group_by(fromID*toID+fromID+toID) %>% summarise(nn = n()) %>% ungroup

buys_pairs<- bitqyFiltered%>% group_by(fromID, toID) %>% summarise(nn = n()) %>% ungroup
```

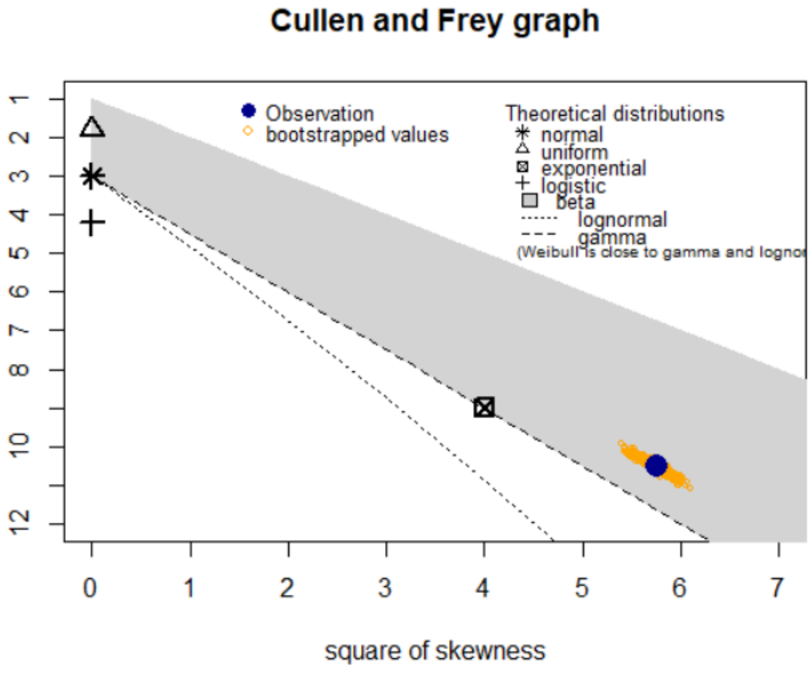
```r
buys_pair_sorted_asc<-buys_pairs[order(buys_pairs$n),]
buys_pair_less_30<-subset(buys_pair_sorted_asc,n<30)
buys_pair_data<-buys_pair_less_30

#####find out estimates of paramaters of several distribution based on the buys_pairs data set#####
exp_dis <- fitdist(buys_pair_data$n, 'exp')
exp_dis
gamma_dis <- fitdist(buys_pair_data$n, 'gamma')
gamma_dis
lnorm_dis <- fitdist(buys_pair_data$n, 'lnorm')
lnorm_dis
pois_dis <- fitdist(buys_pair_data$n, 'pois')
pois_dis
weibull_dis <- fitdist(buys_pair_data$n, 'weibull')
weibull_dis

gofstat(list(exp_dis, gamma_dis, lnorm_dis))
descdist(buys_sorted_asc$n,boot=1000)

#lognorm
fit_lnorm <- fitdist(buys_pair_less_30$n,"lnorm")
summary(fit_lnorm)
plot(fit_lnorm)
cdfcomp(fit_lnorm)

#exp
fit_exp <- fitdist(buys_pair_less_30$n,"exp")
summary(fit_exp)
plot(fit_exp)
cdfcomp(fit_exp)

#gamma
fit_gamma <- fitdist(buys_pair_less_30$n,"gamma")
summary(fit_gamma)
plot(fit_gamma)
cdfcomp(fit_gamma)

#weibull
fit_weibull <- fitdist(buys_pair_less_30$n,"weibull")
summary(fit_weibull)

#normal
fit_normal <- fitdist(buys_pair_less_30$n,"norm")
summary(fit_normal)

#pois
#normal
fit_pois <- fitdist(buys_pair_less_30$n,"pois")
summary(fit_pois)

#unif
fit_unif <- fitdist(buys_pair_less_30$n,"unif")
summary(fit_unif)
plot(fit_unif)
cdfcomp(fit_unif)

######################draw graph######################
all_density <- ggplot(data=buys_pair_less_30) +
  geom_histogram(bins=30,aes(x = buys_pair_less_30$n, ..density..)) +
  stat_function(fun = dlnorm, args = list(meanlog = 0.9122759, sdlog = 0.9075324),
                colour = "red")+
  stat_function(fun = dgamma, args = list(shape = 1.2923088, rate=0.3361498),
                colour = "blue")+
  stat_function(fun=dexp, args=list(rate=0.2600901),colour="green")+
  stat_function(fun=dweibull, args=list(shape=1.083871, scale=3.982783),colour="yellow")+
  stat_function(fun=dpois, args=list(lambda=3.844821),colour="orange")+
  xlab("No.Buys")
all_density

########################################Question 1############################################


########################################Question 2############################################
bitqy_prices <- read_delim("C:/Users/ygaoq/OneDrive/MyDocuments/2019 Spring/Statistics/Project/Blockchain-Tokens-Data-Analytics/bitqy", delim = "\t", col_names = T) #load token price data
names(bitqy_prices) <- make.names(names(bitqy_prices))
bitqy_prices <- bitqy_prices %>% mutate(date = as.Date(Date, format = '%m/%d/%Y'))

bitqy <- read_delim('C:/Users/ygaoq/OneDrive/MyDocuments/2019 Spring/Statistics/Project/Blockchain-Tokens-Data-Analytics/networkbitqyTX.txt', delim = " ", col_names = F)
names(bitqy) <- c('fromID', 'toID', 'unixTime', 'tokenAmount')
decimals <- 2
supply <- 1 * 10^10
bitqy_filtered <-filter(bitqy,tokenAmount < decimals * supply)
## convert data type of unixTime
bitqy_filtered <- bitqy_filtered %>%
  mutate(date = anydate(unixTime))
names(bitqy_filtered) <- c('fromID', 'toID', 'unixTime', 'tokenAmount', 'date')

## merge the prices and edge
bitqy_merged<-merge(x = bitqy_prices, y = bitqy_filtered, by = "date", all.x = TRUE)

################Determin K##########################
top_30_buyers<-buys_sorted_dec%>%head(30)

top_K<-c(1:30)
count <- 1
for (val in top_K) {
  top_K_buyers<-buys_sorted_dec%>%head(val)
  filter_K_bitqy_merged<-filter(bitqy_merged,toID %in% top_K_buyers$toID)
  filter_K_bitqy_merged=transform(filter_K_bitqy_merged,average_price= (Open+Close)/2)
  filter_K_bitqy_merged$num_Date <-  as.numeric(as.POSIXct(filter_K_bitqy_merged$date))
  filered<-filter_K_bitqy_merged%>% group_by(num_Date) %>% summarise(nn = n(),Close=mean(Close),tokenAmount=sum(tokenAmount),Open=mean(Open))
  shift <- function(x, n){
    c(x[-(seq(n))], rep(NA, n))
  }
  filered$new_Close<-shift(filered$Close,1)
```

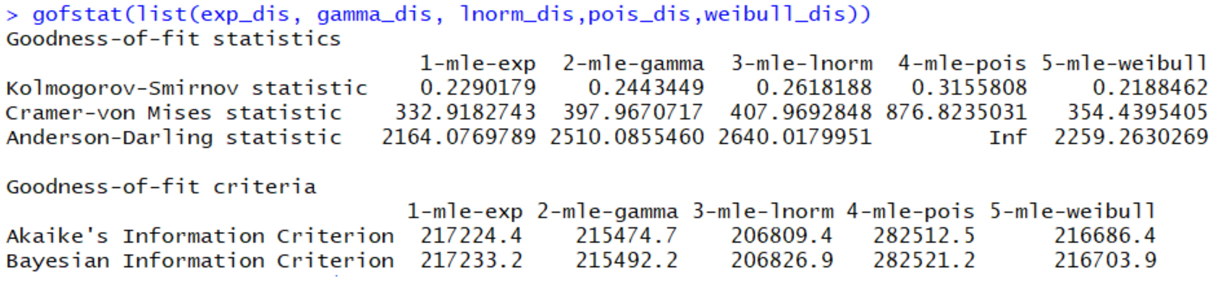
```
        num_rows<-nrow(filered)
        filered[-num_rows,]
        regression<-lm(filered$new_Close~filered$tokenAmount+filered$n+filered$Open)

        setwd("C:/Users/ygaoq/Desktop/bitqy")
        yourfilename=paste("W",val,".txt",sep="")
        capture.output(summary(regression),append = TRUE,file = "C:/Users/ygaoq/Desktop/bitqy/Final_Result.txt")


        summary(regression)
        par(mfcol=c(2,2))
        setwd("C:/Users/ygaoq/Desktop/bitqy")
        yourfilename=paste("A",val,".png",sep="")
        png(file=yourfilename)
        opar <- par(mfrow=c(2,2))
        plot(regression)
        dev.off()
      }
```



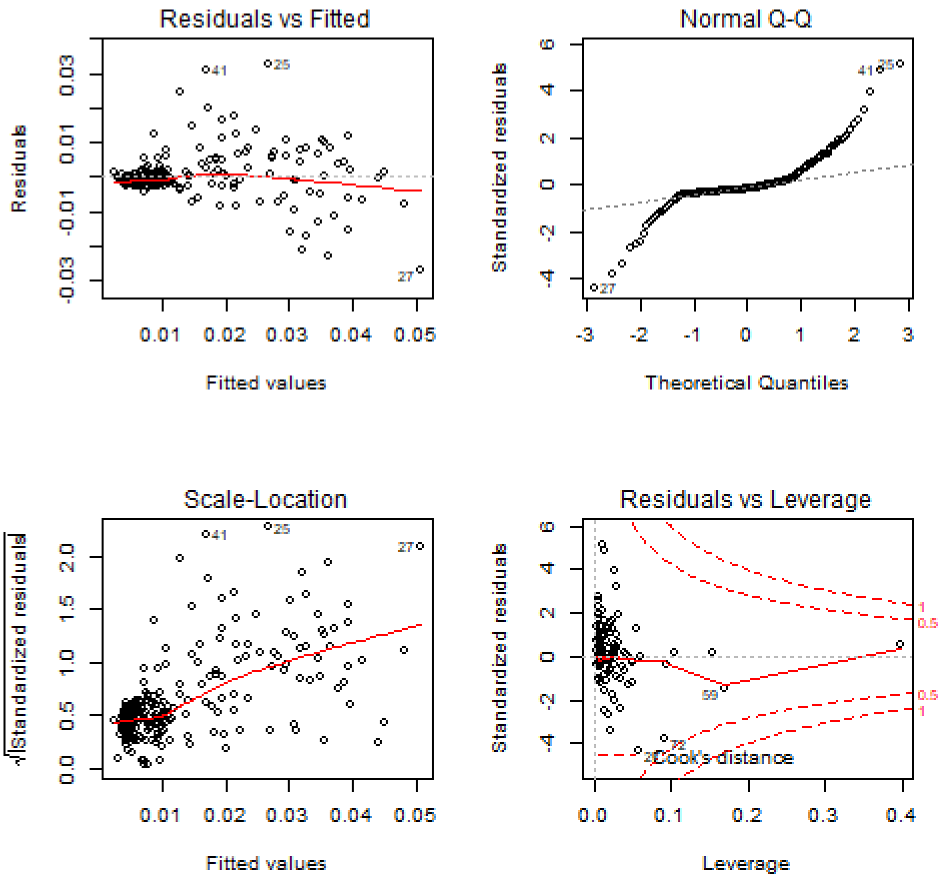


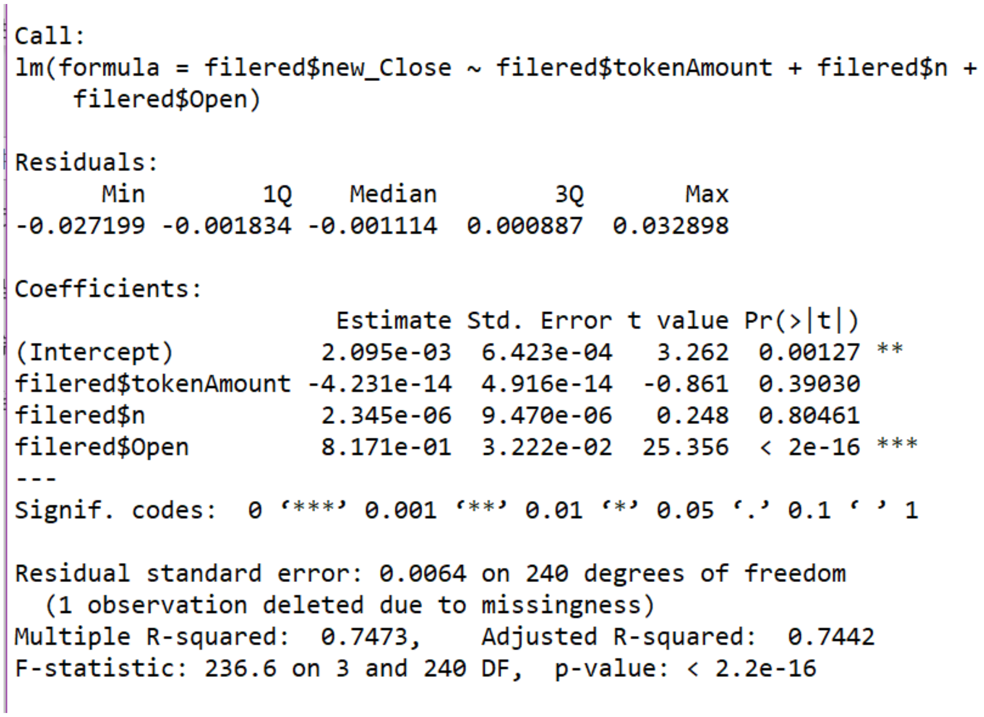
```
> exp_dis <- fitdist(buys_pair_data$n, 'exp')
> exp_dis
Fitting of the distribution ' exp ' by maximum likelihood
Parameters:
      estimate  Std. Error
rate 0.2600901 0.001208959
> gamma_dis <- fitdist(buys_pair_data$n, 'gamma')
> gamma_dis
Fitting of the distribution ' gamma ' by maximum likelihood
Parameters:
       estimate  Std. Error
shape 1.2923088 0.007644403
rate  0.3361498 0.002417218
> lnorm_dis <- fitdist(buys_pair_data$n, 'lnorm')
> lnorm_dis
Fitting of the distribution ' lnorm ' by maximum likelihood
Parameters:
         estimate  Std. Error
meanlog 0.9122759 0.004218481
sdlog   0.9075324 0.002982900
> pois_dis <- fitdist(buys_pair_data$n, 'pois')
> pois_dis
Fitting of the distribution ' pois ' by maximum likelihood
Parameters:
       estimate  Std. Error
lambda 3.844821 0.009114482
> weibull_dis <- fitdist(buys_pair_data$n, 'weibull')
> weibull_dis
Fitting of the distribution ' weibull ' by maximum likelihood
Parameters:
      estimate  Std. Error
shape 1.083871 0.003679308
scale 3.982783 0.018143064
```





```
> gofstat(list(exp_dis, gamma_dis, lnorm_dis,pois_dis,weibull_dis))
Goodness-of-fit statistics
                                1-mle-exp  2-mle-gamma  3-mle-lnorm  4-mle-pois 5-mle-weibull
Kolmogorov-Smirnov statistic    0.2290179    0.2443449    0.2618188   0.3155808     0.2188462
Cramer-von Mises statistic    332.9182743  397.9670717  407.9692848 876.8235031   354.4395405
Anderson-Darling statistic   2164.0769789 2510.0855460 2640.0179951         Inf  2259.2630269

Goodness-of-fit criteria
                               1-mle-exp 2-mle-gamma 3-mle-lnorm 4-mle-pois 5-mle-weibull
Akaike's Information Criterion  217224.4    215474.7    206809.4   282512.5      216686.4
Bayesian Information Criterion  217233.2    215492.2    206826.9   282521.2      216703.9
```





```
Call:
lm(formula = filered$new_Close ~ filered$tokenAmount + filered$n + 
    filered$Open)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.027199 -0.001834 -0.001114  0.000887  0.032898 

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)    
(Intercept)          2.095e-03  6.423e-04   3.262  0.00127 ** 
filered$tokenAmount -4.231e-14  4.916e-14  -0.861  0.39030    
filered$n            2.345e-06  9.470e-06   0.248  0.80461    
filered$Open         8.171e-01  3.222e-02  25.356  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.0064 on 240 degrees of freedom
  (1 observation deleted due to missingness)
Multiple R-squared:  0.7473,	Adjusted R-squared:  0.7442 
F-statistic: 236.6 on 3 and 240 DF,  p-value: < 2.2e-16
```