Report

pushpita panigrahi(pxp171530), akash chand(axc173730), siddharth swarup panda(ssp171730)

Read the token data

We chose networkbnbTX token as our dataset.

```
file <-'/Users/pushpitapanigrahi/Desktop/PushpitaFiles/GitHub/statistics-for-DS/netwo
rkbnbTX.txt'
col_names <- c("FROMNODE","TONODE","TIME","TOKENAMOUNT")
mydata <- read.csv( file, header = FALSE, sep = " ", dec = ".", col.names = col_names
)
amounts <- mydata[4]

totalSupply <- 192443301
subUnits <- 18
totalAmount <- totalSupply * (10 ^ subUnits)

head(mydata)
```

```
##   FROMNODE  TONODE       TIME  TOKENAMOUNT
## 1       82 1443996 1524611372 4.071000e+19
## 2       82 1443997 1524611836 2.291000e+19
## 3        5 1443998 1524611992 2.297303e+18
## 4  1443999 1444000 1524612337 8.740000e+18
## 5       44 1444001 1524612660 1.180000e+18
## 6        5 1444002 1524612970 3.276959e+20
```

Preprocessing

The preprocessing step involves removal of fraudulent transactions which might affect the distribution estimate negatively. The total supply of the networkbnb token is 192443301 (quoted from etherscan.io) and the range of subunits for the token is 18 decimal units. Thus any transaction that attempts to log a value greater than the product of total supply and subunits is deemed as fraudulent.

The token networkbnb does not have any fraudulent transactions.

```
temp <- which(mydata< totalAmount)
#print meta data
message('Maximum allowed amount : ', totalAmount)
```

```
## Maximum allowed amount : 1.92443301e+26
```

```r
count <- 0
outliers <- 0
for( a in 1:nrow(amounts)){
  if( a > totalAmount){
    outliers <- outliers + 1
  }
  else{
    count <- count + 1
  }
}
message('Number of outliers : ',outliers)
```

```
## Number of outliers : 0
```

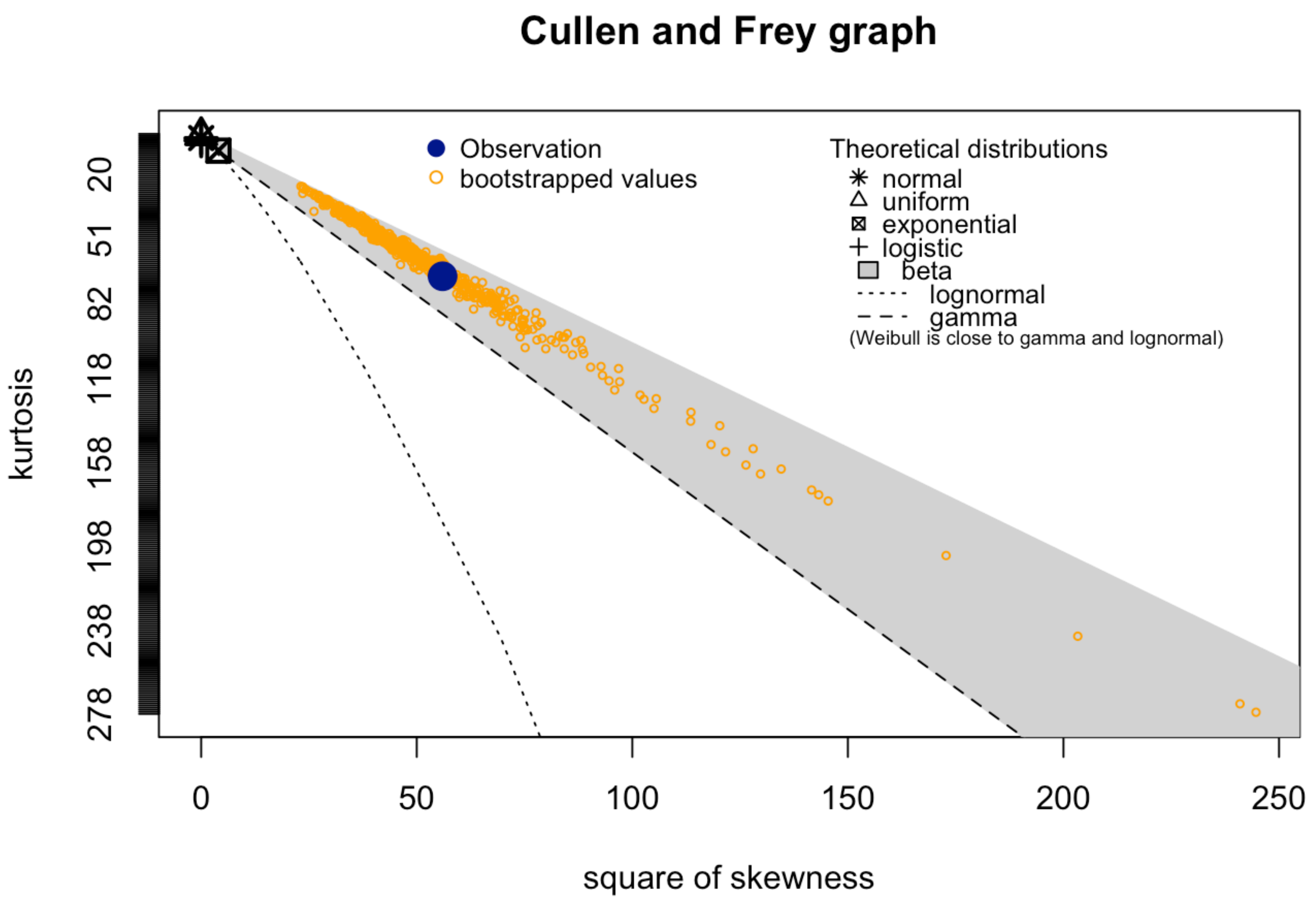
```r
message('Number of valid amounts : ',count)
```

```
## Number of valid amounts : 357142
```

Calculating and plotting selling frequency

```
## Using freq as weighting variable
```

```
##   Users_Count Sell_Count
## 1           1      16575
## 2           2       3962
## 3           3       2115
## 4           4       1284
## 5           5        870
## 6           6        702
```




```
## summary statistics
## ------
## min:  65   max:  34809
## median:  399
## mean:  994.8245
## estimated sd:  2931.883
## estimated skewness:  7.48285
## estimated kurtosis:  68.98622
```

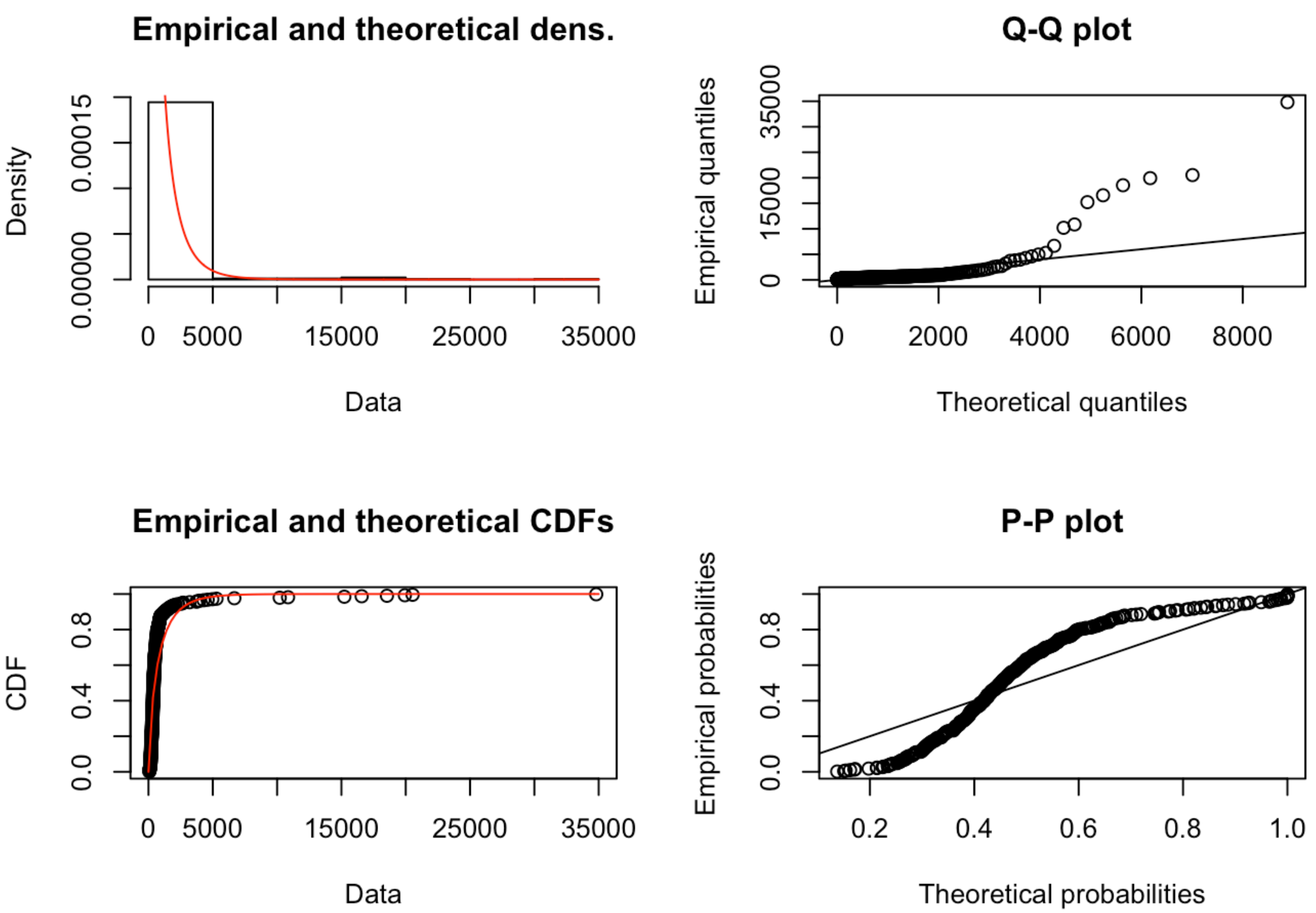
Approximating the selling distributions

From the above Cullen and Frey graph we could narrow down our distribution selection to Weibull, lognormal, gamma and poisson.

```
distributionFit_Seller_pois <- fitdist(countFromFf$Sell_Count, "pois", method ="mle")
distributionFit_Seller_wb <- fitdist(countFromFf$Sell_Count, "weibull", method ="mle"
)
distributionFit_Seller_ln <- fitdist(countFromFf$Sell_Count, "lnorm", method ="mle")
distributionFit_Seller_gm <- fitdist(countFromFf$Sell_Count, "gamma" ,method="mme")
distributionFit_Seller_wb
```

```
## Fitting of the distribution ' weibull ' by maximum likelihood
## Parameters:
##          estimate  Std. Error
## shape   0.7719378  0.02515483
## scale 774.1209761 56.42896598
```

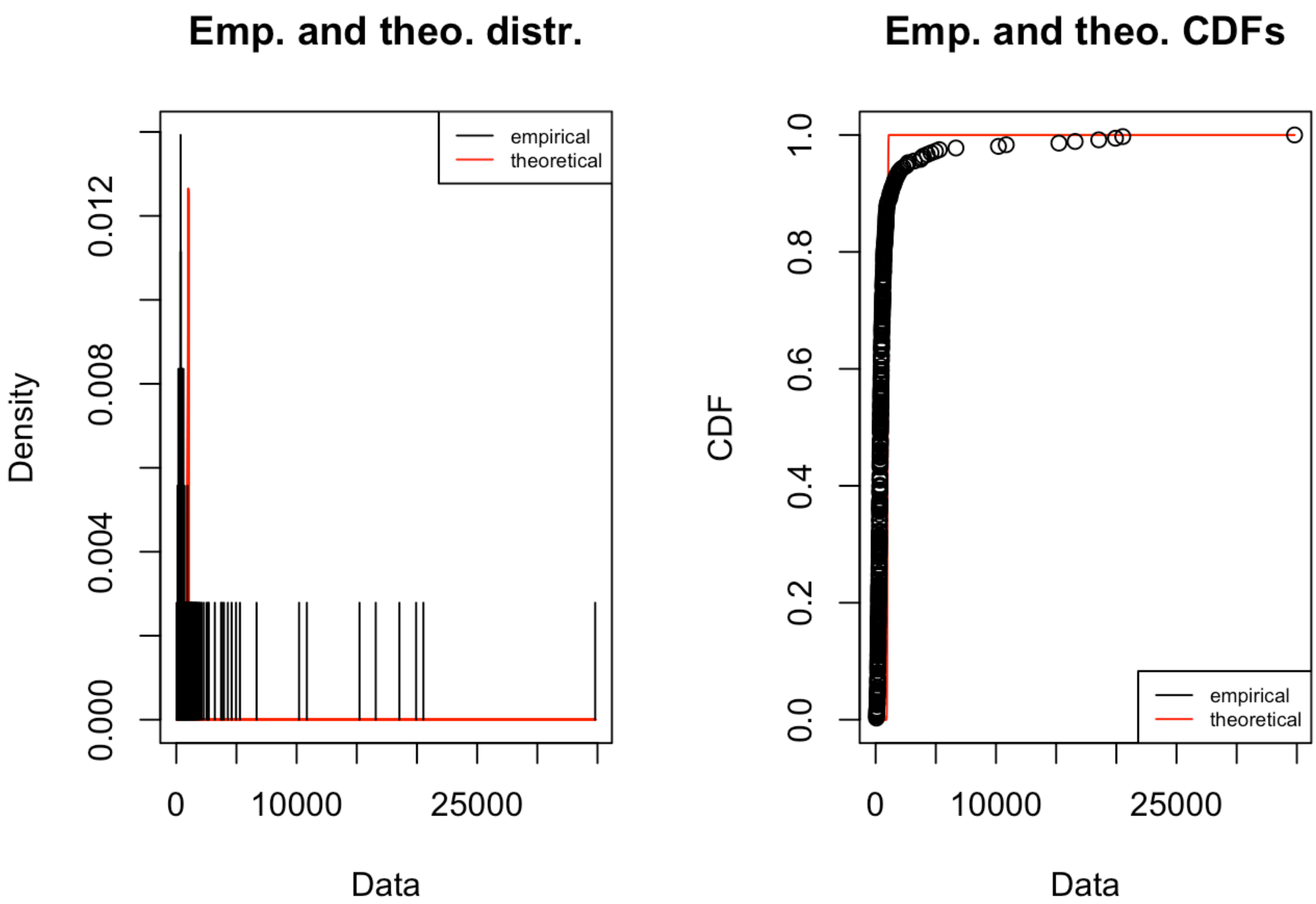
```
plot(distributionFit_Seller_wb)
```




```
distributionFit_Seller_pois
```

```
## Fitting of the distribution ' pois ' by maximum likelihood
## Parameters:
##        estimate Std. Error
## lambda 994.8245   1.664616
```

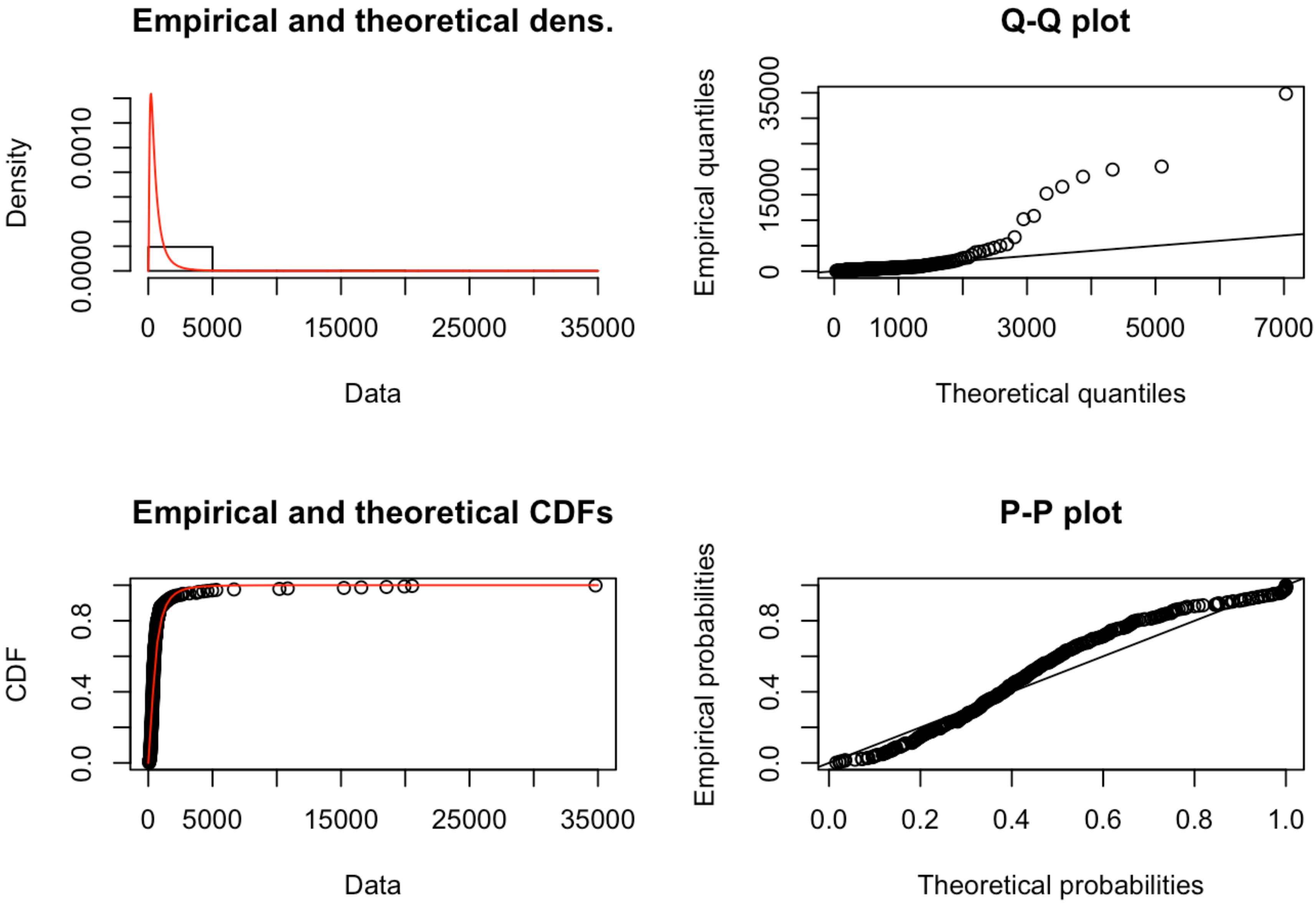
```
plot(distributionFit_Seller_pois)
```




```
distributionFit_Seller_ln
```

```
## Fitting of the distribution ' lnorm ' by maximum likelihood
## Parameters:
##          estimate Std. Error
## meanlog 6.1329835 0.04809244
## sdlog   0.9112216 0.03400630
```

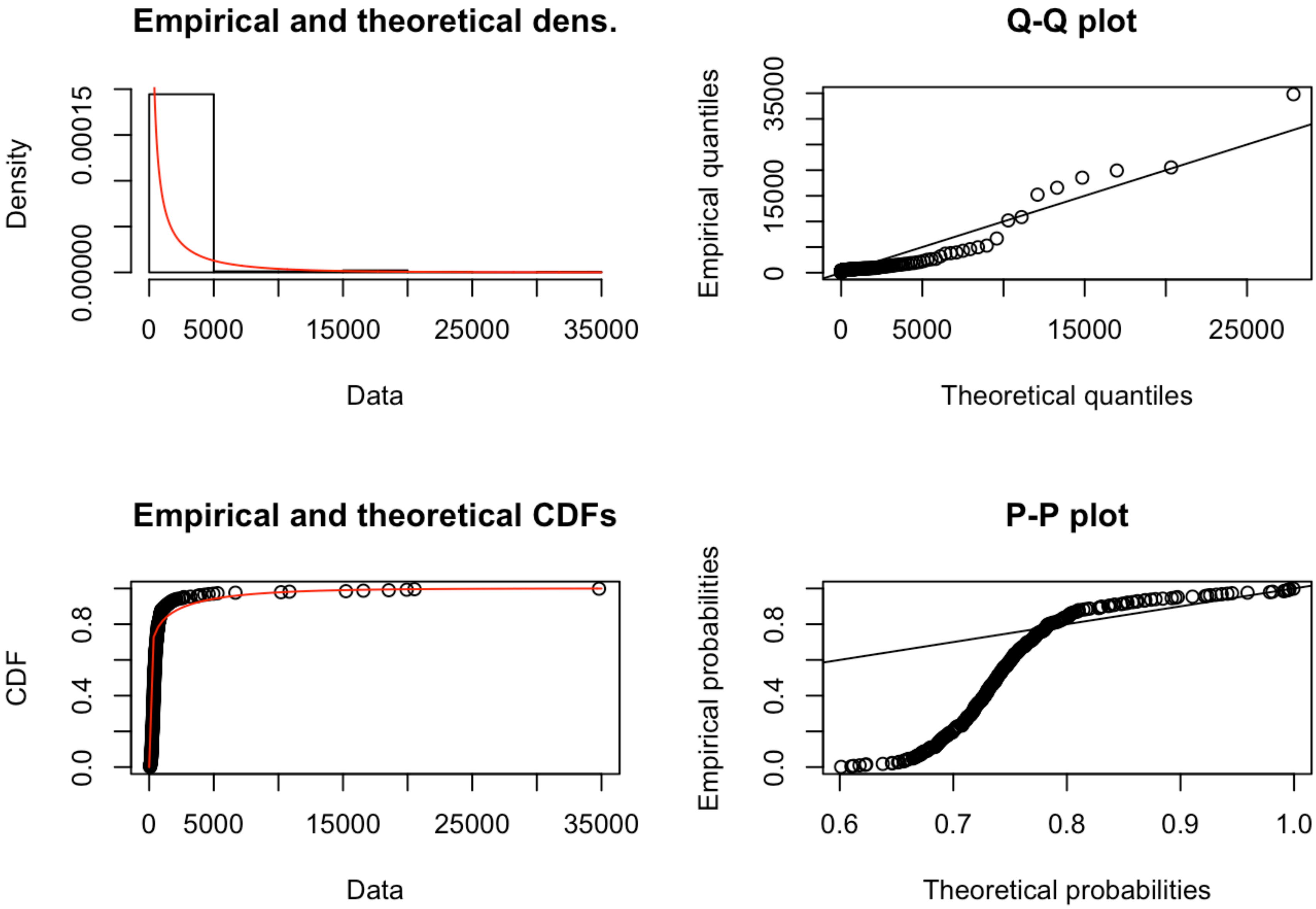
```
plot(distributionFit_Seller_ln)
```




```
distributionFit_Seller_gm
```

```
## Fitting of the distribution ' gamma ' by matching moments
## Parameters:
##           estimate
## shape 0.1154545321
## rate  0.0001160552
```

```
plot(distributionFit_Seller_gm)
```



Calculating the buying frequency

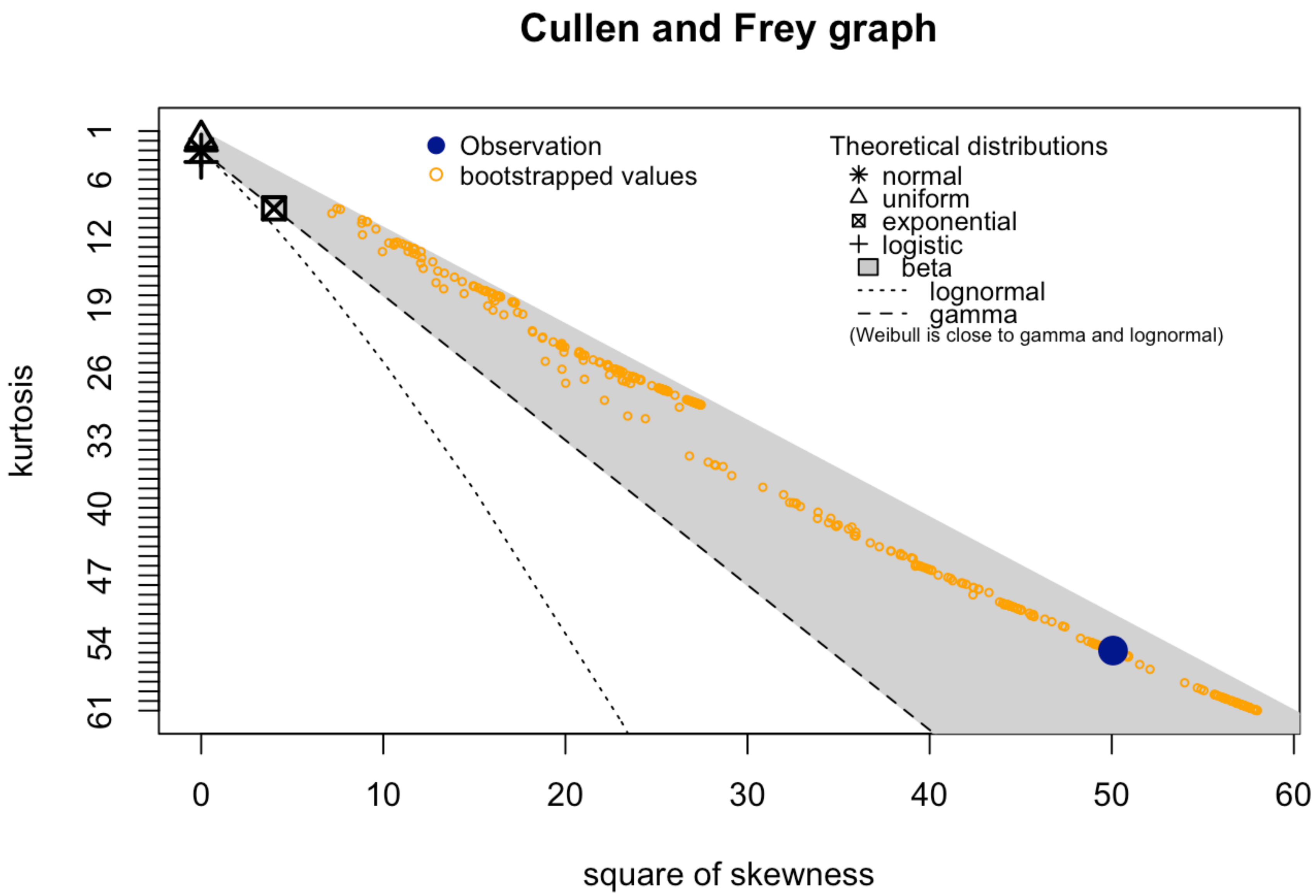
```
countToDf <- count(mydata, "TONODE")
countToFf <- count(countToDf, "freq")
```

```
## Using freq as weighting variable
```

```
colnames(countToFf) <- c("Users_Count", "Buy_Count")
head(countToFf)
```

```
##   Users_Count Buy_Count
## 1           1    252994
## 2           2     56706
## 3           3     16029
## 4           4      5184
## 5           5      2240
## 6           6      1452
```

```
descdist(countToFf$Buy_Count, boot=500)
```




```
## summary statistics
## ------
## min:  24   max:  252994
## median:  117.5
## mean:  6157.621
## estimated sd:  33890.53
## estimated skewness:  7.076031
## estimated kurtosis:  54.78311
```

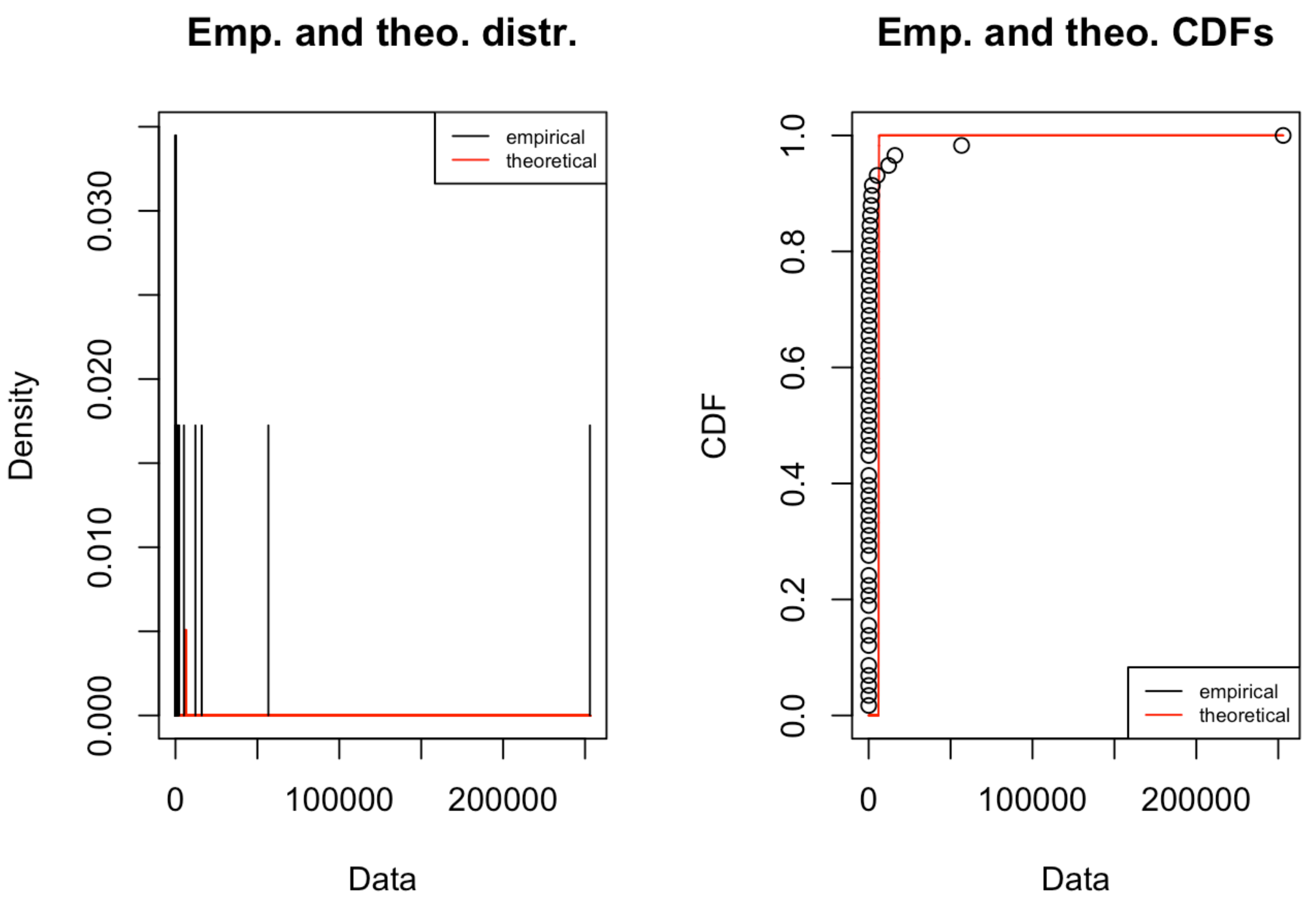
Approximating the buying distributions

```
distributionFit_Buyer_pois <- fitdist(countToFf$Buy_Count, "pois", method ="mle")
distributionFit_Buyer_wb <- fitdist(countToFf$Buy_Count, "weibull", method ="mle")
distributionFit_Buyer_ln <- fitdist(countToFf$Buy_Count, "lnorm", method ="mle")
distributionFit_Buyer_gm <- fitdist(countToFf$Buy_Count, "gamma", method ="mme")

distributionFit_Buyer_pois
```

```
## Fitting of the distribution ' pois ' by maximum likelihood
## Parameters:
##        estimate Std. Error
## lambda 6157.621   10.26635
```

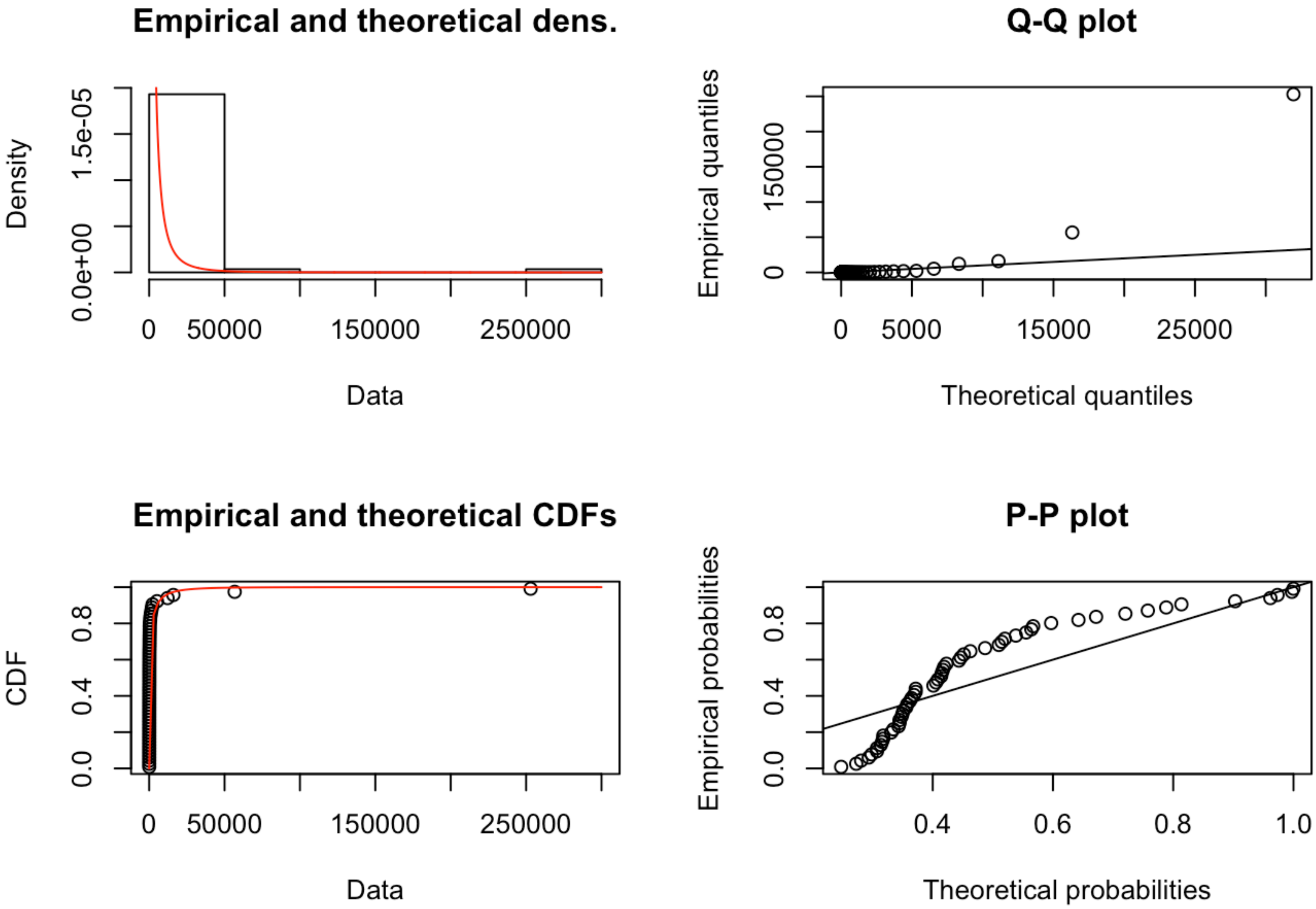
```
plot(distributionFit_Buyer_pois)
```




```
distributionFit_Buyer_wb
```

```
## Fitting of the distribution ' weibull ' by maximum likelihood
## Parameters:
##          estimate   Std. Error
## shape   0.3913615   0.03285488
## scale 595.1275712 213.11455385
```

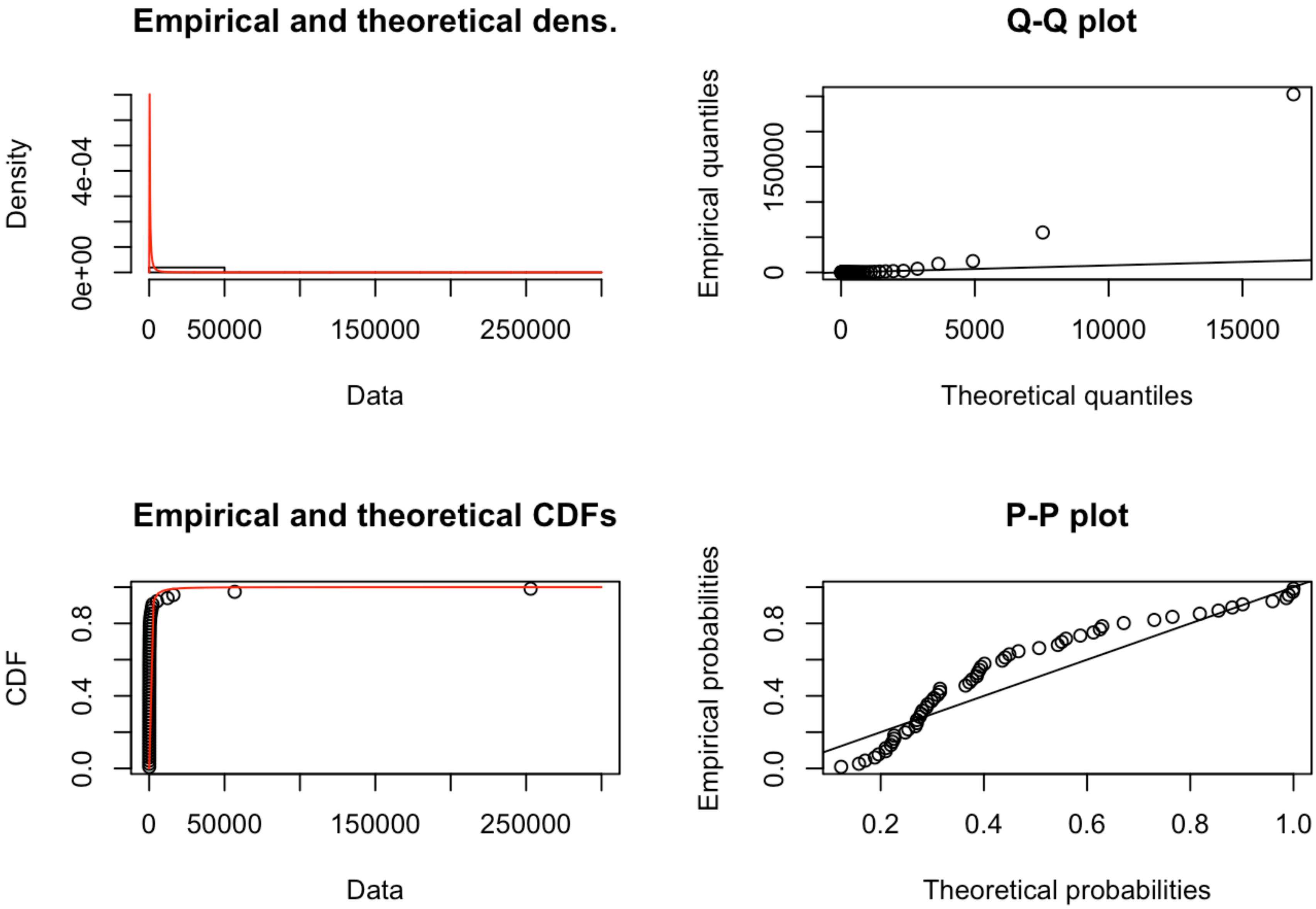
```
plot(distributionFit_Buyer_wb)
```




```
distributionFit_Buyer_ln
```

```
## Fitting of the distribution ' lnorm ' by maximum likelihood
## Parameters:
##          estimate Std. Error
## meanlog 5.323868  0.2432331
## sdlog   1.852408  0.1719915
```

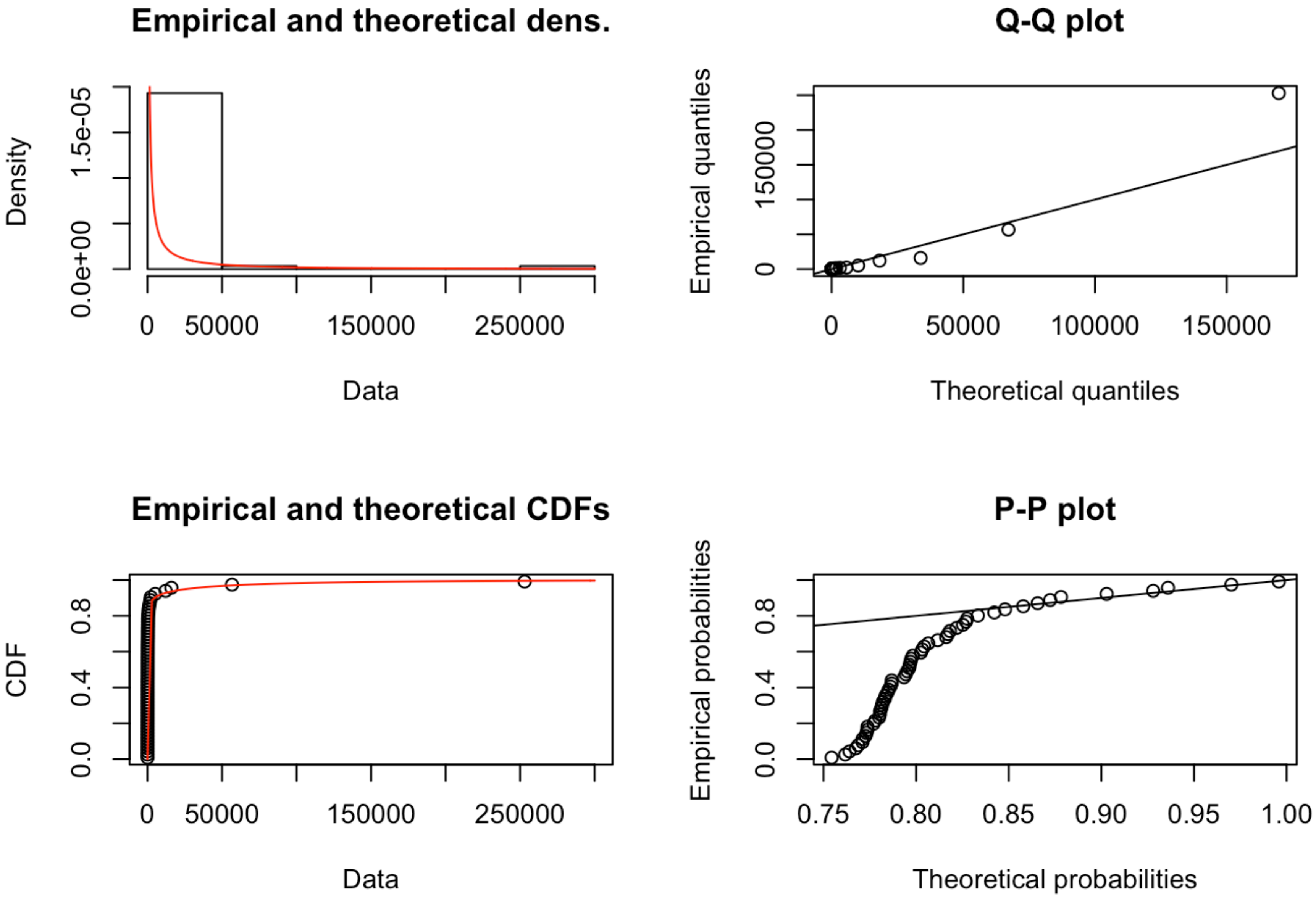
```
plot(distributionFit_Buyer_ln)
```




```
distributionFit_Buyer_gm
```

```
## Fitting of the distribution ' gamma ' by matching moments
## Parameters:
##            estimate
## shape 3.359095e-02
## rate  5.455183e-06
```

```
plot(distributionFit_Buyer_gm)
```



Conclusion

From the above graph estimates, both buy and sell frequency for our dataset follows LOG-NORMAL distribution as the standard error is least and the emperical distribution curve follows the theoritical distribution curve most accurately.