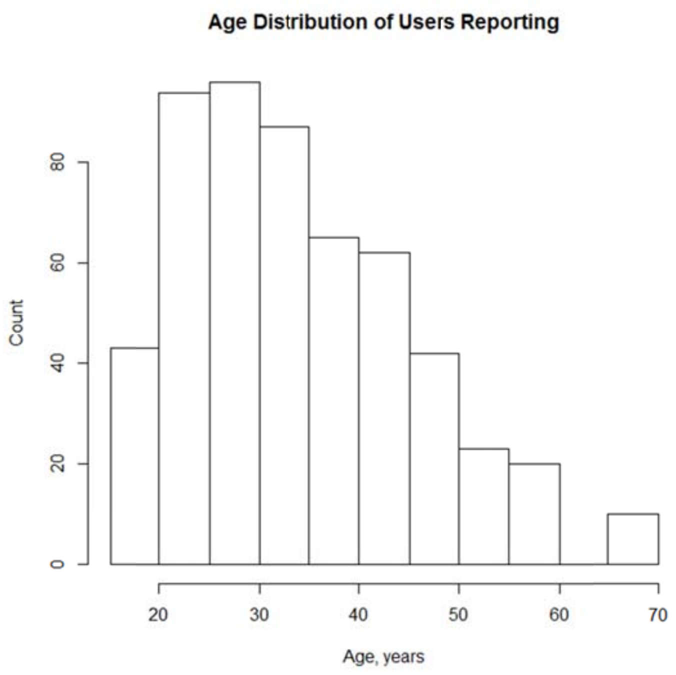
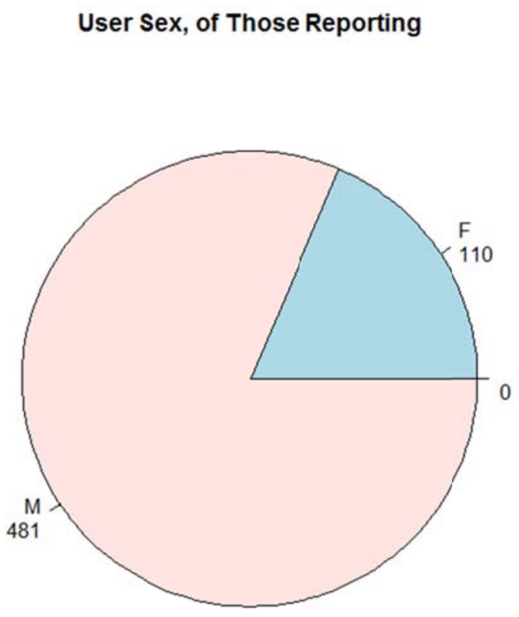
QUESTION 1: (Describing large data)

We decided to analyze our project data for static user characteristics. The data is from a stream of usage feedback sent from a mobile app. Raw data is reported from user devices and collected in a comma-delimited text stream which describes user characteristics, actions, and timestamp. Raw data sample after formatting:

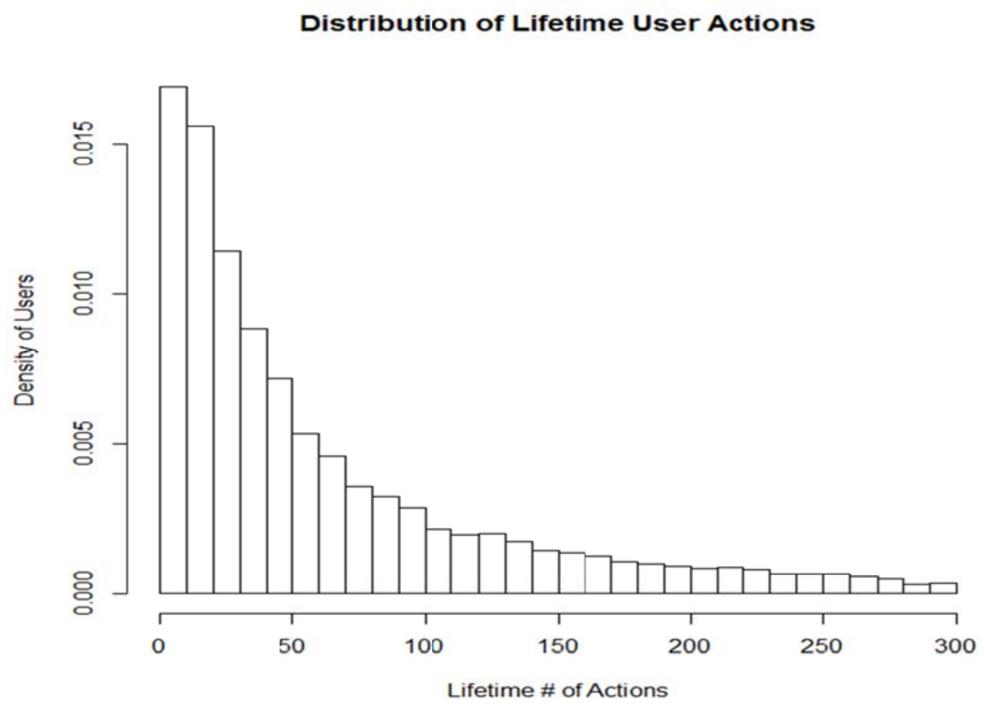
```
        ID Age Sex Location Action              Time Hardware
1 22129605  24   F            1000 2013-06-01 07:00:38        Y
2 22129605  24   F            1000 2013-06-01 08:05:41        Y
3 22129605  24   F            1000 2013-06-01 08:23:04        Y
4 22129605  24   F            1001 2013-06-01 07:12:42        Y
…
```

a. The *60MB* of usage data contains over *1.6 million rows*, representing activity of *12,991 unique users* between *06/01/2013* and *09/15/2013*.
b. While the data contains demographic information, only *4.5%* of users reported gender and *4.2%* reported Age (mean of *34.5 years*, median *34*). Of those reporting, the age and gender distributions are below:

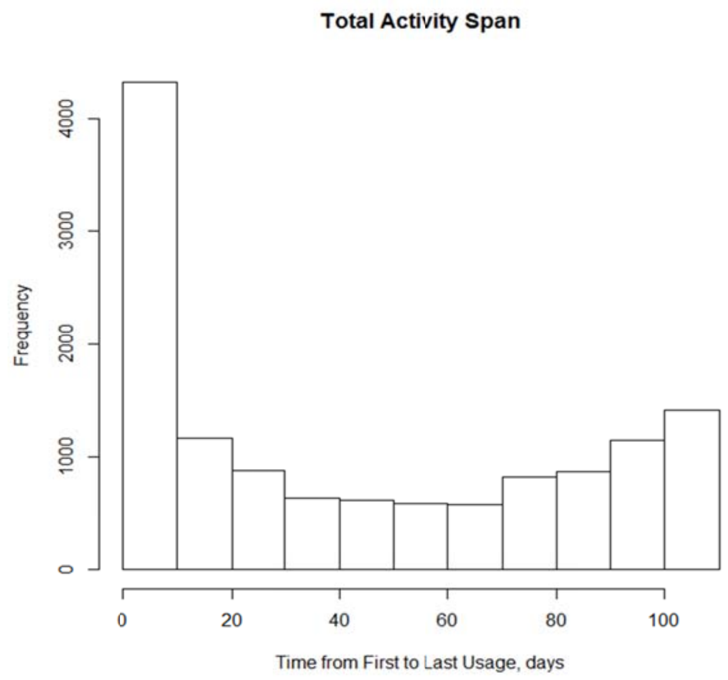




c. An optional hardware device is available to pair with the app. *72.6%* of users have this hardware.
d. During the analysis window, each user has accumulated a number of actions as distributed below:



e. During the window, the average user's total activity span lasted a number of days as distributed below:



QUESTION 2 (Find relationships)

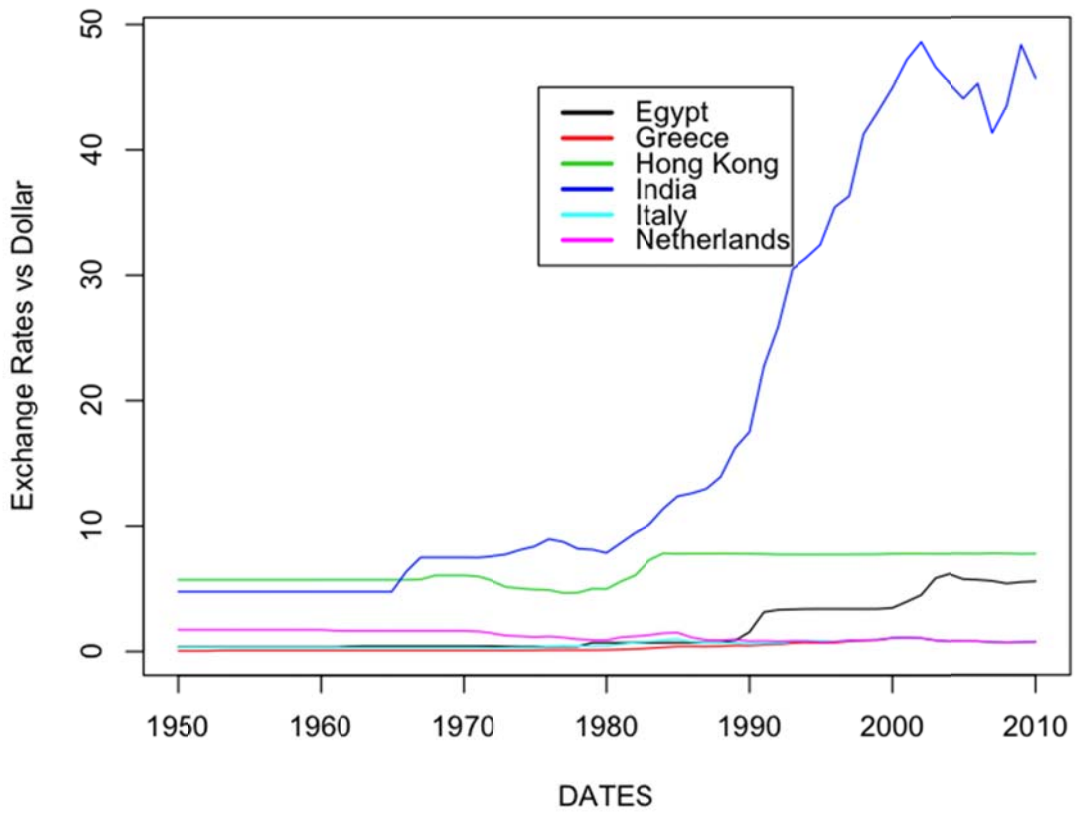
Below is a section of the exchange rate data for 6 countries:

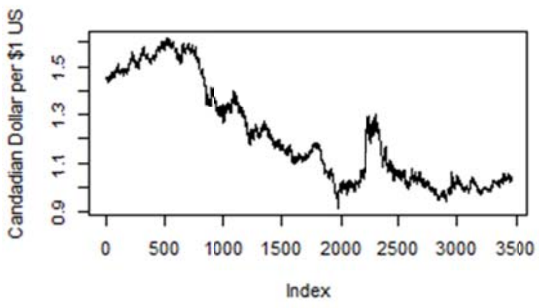
	DATE	US	Egypt	Greece	Hong.Kong	India	Italy	Netherlands
1	1/1/50	1	0.3481407	0.04399120	5.71	4.7629	0.3226230	1.723911
2	1/1/51	1	0.3482420	0.04399118	5.71	4.7629	0.3222620	1.724365
3	1/1/52	1	0.3482420	0.04399118	5.71	4.7629	0.3222105	1.724819
4	1/1/53	1	0.3481407	0.07621422	5.71	4.7629	0.3222105	1.724365
5	1/1/54	1	0.3481407	0.08804109	5.71	4.7629	0.3222105	1.724365
6	1/1/55	1	0 .3482420	0.08804109	5.71	4.7639	0.3222105	1.724365

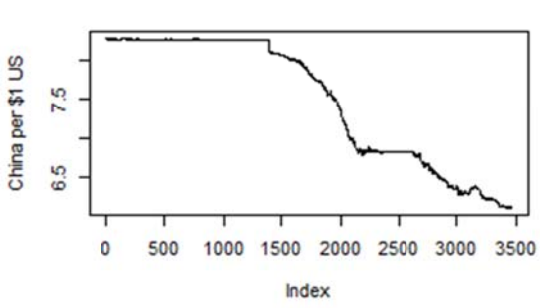
This data can be downloaded in .CSV format from the St. Louis Fed, or alternatively we can use QUANTMOD to pull from the FRED as a source:

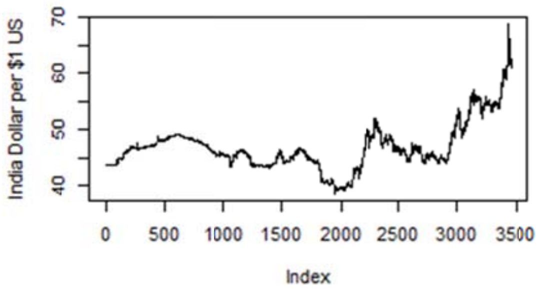
```
getSymbols(c("DEXCAUS", "DEXCHUS", "DEXINUS", "DEXJPUS", "DEXUSEU",
"DEXUSUK"), src='FRED')
```

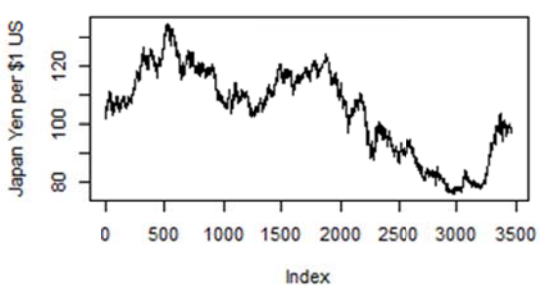
Below is a time series of these exchange rates, both superimposed and as a plot matrix:

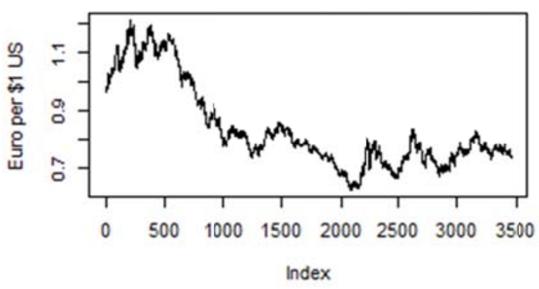


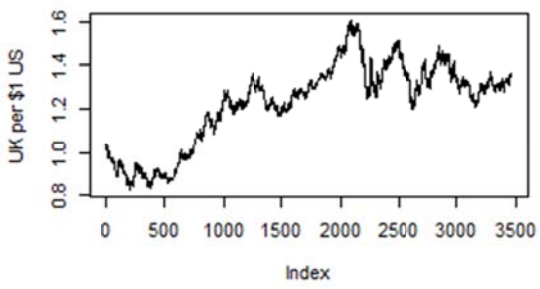




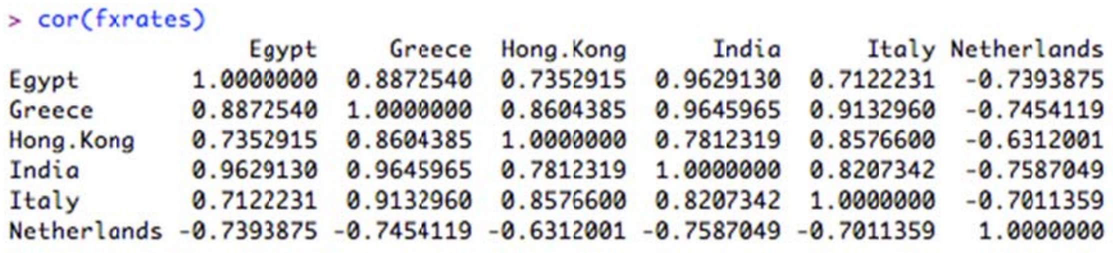




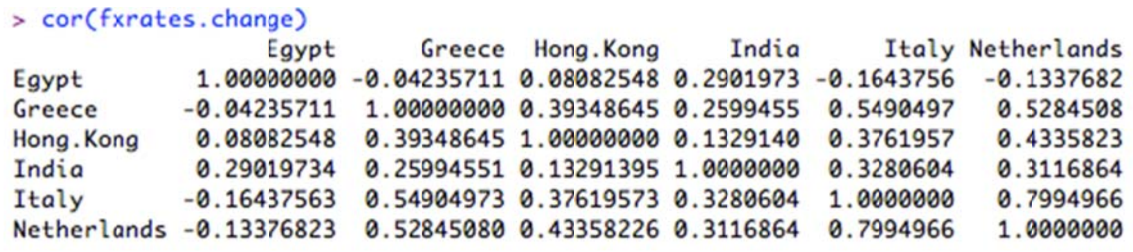




Below is the correlation of the rates themselves:


```
> cor(fxrates)
                 Egypt     Greece  Hong.Kong      India      Italy Netherlands
Egypt        1.0000000  0.8872540  0.7352915  0.9629130  0.7122231  -0.7393875
Greece       0.8872540  1.0000000  0.8604385  0.9645965  0.9132960  -0.7454119
Hong.Kong    0.7352915  0.8604385  1.0000000  0.7812319  0.8576600  -0.6312001
India        0.9629130  0.9645965  0.7812319  1.0000000  0.8207342  -0.7587049
Italy        0.7122231  0.9132960  0.8576600  0.8207342  1.0000000  -0.7011359
Netherlands -0.7393875 -0.7454119 -0.6312001 -0.7587049 -0.7011359   1.0000000
```

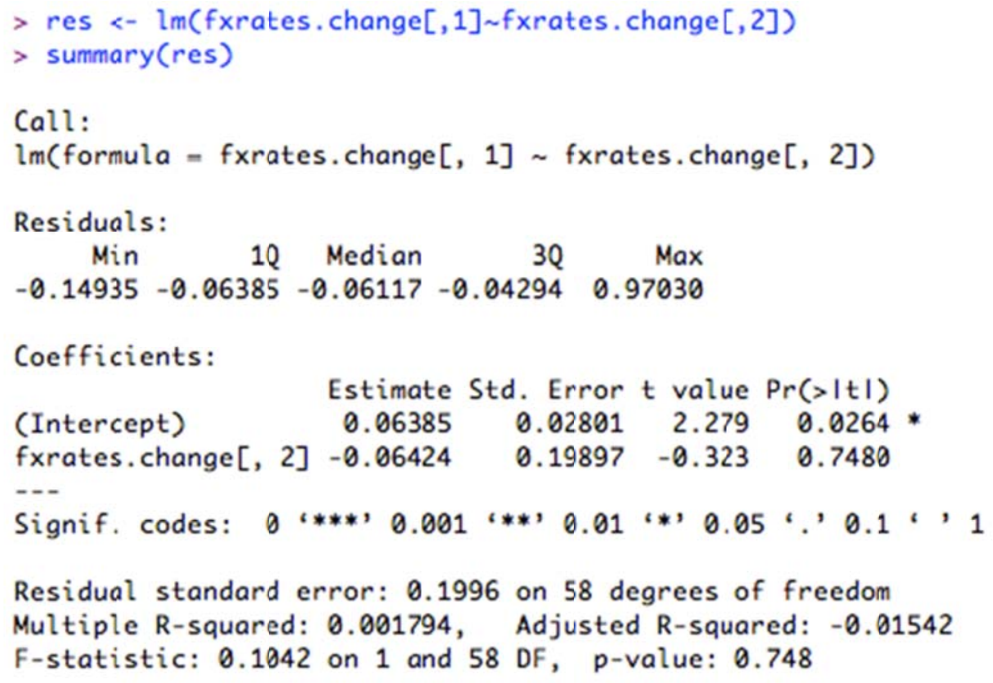
Below is the correlation of the changes of the rates:



```
> cor(fxrates.change)
                  Egypt      Greece  Hong.Kong     India      Italy Netherlands
Egypt        1.00000000 -0.04235711 0.08082548 0.2901973 -0.1643756  -0.1337682
Greece      -0.04235711  1.00000000 0.39348645 0.2599455  0.5490497   0.5284508
Hong.Kong    0.08082548  0.39348645 1.00000000 0.1329140  0.3761957   0.4335823
India        0.29019734  0.25994551 0.13291395 1.0000000  0.3280604   0.3116864
Italy       -0.16437563  0.54904973 0.37619573 0.3280604  1.0000000   0.7994966
Netherlands -0.13376823  0.52845080 0.43358226 0.3116864  0.7994966   1.0000000
```

For any pair of rates we can see the correlation above. However, to establish statistical significance, we need to test this correlation against the null hypothesis that the coefficient is actually zero.

Below we regress the change in exchange rate of Egypt vs. Greece:



```
> res <- lm(fxrates.change[,1]~fxrates.change[,2])
> summary(res)

Call:
lm(formula = fxrates.change[, 1] ~ fxrates.change[, 2])

Residuals:
     Min       1Q   Median       3Q      Max 
-0.14935 -0.06385 -0.06117 -0.04294  0.97030 

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)  
(Intercept)          0.06385    0.02801   2.279   0.0264 *
fxrates.change[, 2] -0.06424    0.19897  -0.323   0.7480  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1996 on 58 degrees of freedom
Multiple R-squared: 0.001794,	Adjusted R-squared: -0.01542 
F-statistic: 0.1042 on 1 and 58 DF,  p-value: 0.748 
```

The p-value of 0.7480 indicates a 74.8% chance that the null hypothesis is valid, i.e. the changes in currency value of Greece is most likely not correlated to the change in currency value of Egypt.

Using the data series you have, can you build a model that uses lagged values of changes in exchange rates to predict future exchange rate changes with a high level of accuracy? (This requires some experimentation and playing with the data. Ideally program R to run all possible cases.)

```
n <- length(fxrates.change$canadaUS)
      res <- lm(fxrates.change$canadaUS[5:n] ~ fxrates.change$canadaUS[4:(n-
      1)] +   fxrates.change$canadaUS[3:(n-2)] +
      fxrates.change$canadaUS[2:(n-3)] + fxrates.change$canadaUS[1:(n-4)])
summary(res)
```

Looking at the lagged values of Canada:US on the current Canada:US exchange rate, we don't see any significant relationship:

```
(Intercept)                        -7.738e-05  1.012e-04  -0.765    0.445
fxrates.change$canadaUS[4:(n - 1)]  1.502e-03  1.701e-02   0.088    0.930
fxrates.change$canadaUS[3:(n - 2)]  1.577e-03  1.700e-02   0.093    0.926
fxrates.change$canadaUS[2:(n - 3)]  2.132e-02  1.701e-02   1.253    0.210
fxrates.change$canadaUS[1:(n - 4)] -5.679e-03  1.701e-02  -0.334    0.738

Residual standard error: 0.005953 on 3456 degrees of freedom
Multiple R-squared: 0.0004908,Adjusted R-squared: -0.000666
F-statistic: 0.4243 on 4 and 3456 DF,  p-value: 0.7912
```

More generally, we can create a linear model of one exchange rate (Canada:US in this case) as a function of all the other rates:

```
res <- lm(fxrates.change$canadaUS ~ fxrates.change$chinaUS +
fxrates.change$indiaUS + fxrates.change$japanUS +
      fxrates.change$EUUS + fxrates.change$UKUS)
summary(res)
```

```
Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)              4.369e-05  9.659e-05   0.452  0.65108
fxrates.change$chinaUS  3.462e-02  9.447e-02   0.366  0.71406
fxrates.change$indiaUS  1.857e-01  1.894e-02   9.804  < 2e-16 ***
fxrates.change$japanUS -1.099e-01  1.358e-02  -8.089 8.24e-16 ***
fxrates.change$EUUS    -2.178e+00  1.011e+00  -2.155  0.03124 *
fxrates.change$UKUS    -2.612e+00  1.011e+00  -2.585  0.00977 **
---
Residual standard error: 0.005078 on 3459 degrees of freedom
Multiple R-squared: 0.2724,       Adjusted R-squared: 0.2713
F-statistic:   259 on 5 and 3459 DF,  p-value: < 2.2e-16
```

We can see above the Canadian rate has a correlation to changes to India, Japan, UK, and EU. What about lagged versions of each of these? This model is with one lag from each of these rates:

```
Coefficients:
                                   Estimate Std. Error t value Pr(>|t|)
(Intercept)                      -0.0001270  0.0001131  -1.122   0.2618
fxrates.change$chinaUS[1:(n - 1)]  0.0668808  0.1106283   0.605   0.5455
fxrates.change$indiaUS[1:(n - 1)]  0.0339124  0.0221841   1.529   0.1264
fxrates.change$japanUS[1:(n - 1)] -0.0315011  0.0159039  -1.981   0.0477 *
fxrates.change$EUUS[1:(n - 1)]     1.1739568  1.1837346   0.992   0.3214
fxrates.change$UKUS[1:(n - 1)]     1.1803939  1.1832612   0.998   0.3186
---
Residual standard error: 0.005945 on 3458 degrees of freedom
Multiple R-squared: 0.002603,   Adjusted R-squared: 0.001161
F-statistic: 1.805 on 5 and 3458 DF,  p-value: 0.1084
```

Run a vector autoregression (VAR) on the data of exchange rate changes that you have. What inferences can you make from the results?

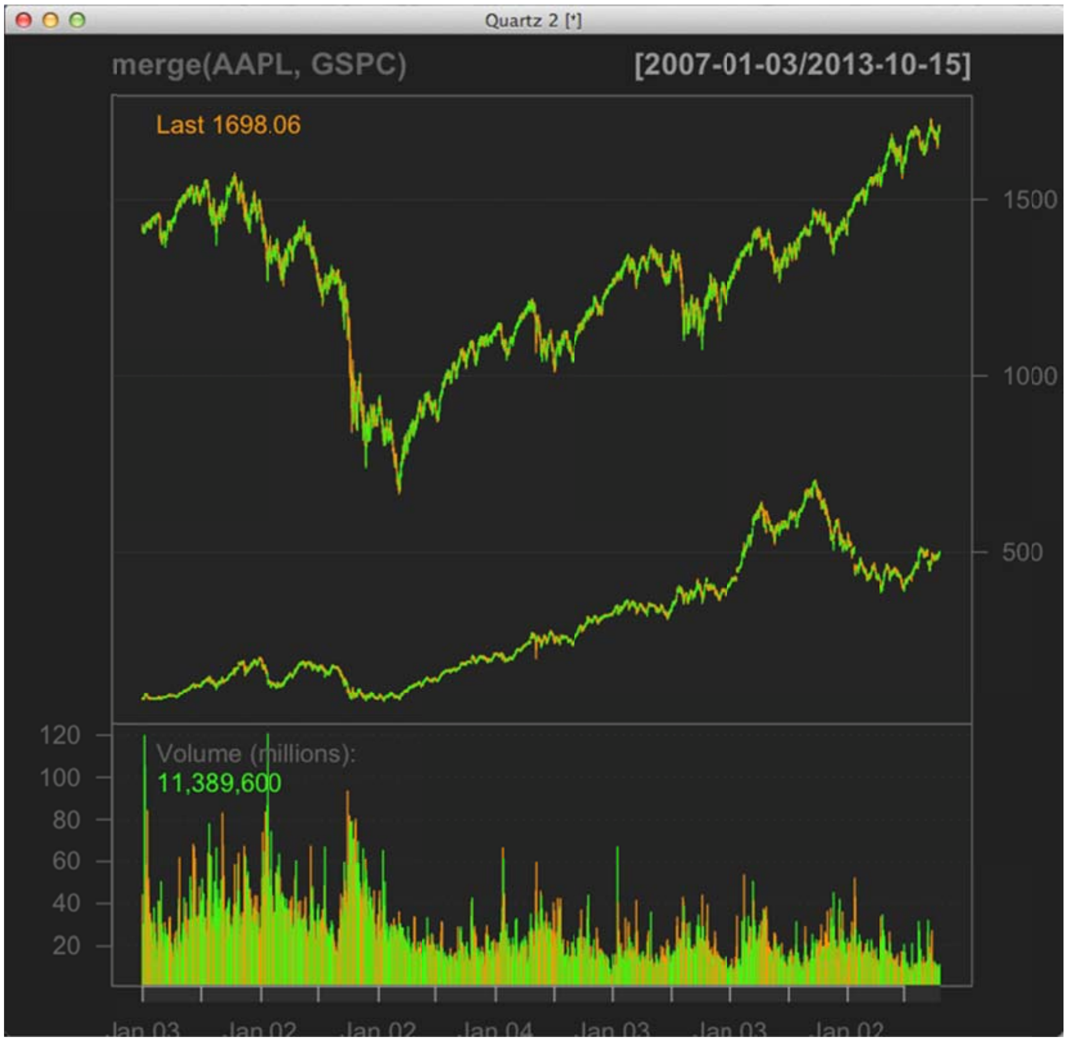
```
var6 = ar(fxrates.change,aic=TRUE,order=6)
var6$ar
```

> var6$ar

chinaUS.canadaUS	indiaUS.canadaUS	japanUS.canadaUS	EUUS.canadaUS	UKUS.canadaUS	canadaUS.chinaUS
-0.0124	0.0026	0.0358	0.0043	-0.0314	0.0316
0.0040	0.0046	0.0458	0.0193	0.0149	-0.0144
0.0237	0.0014	0.0275	0.0268	0.0218	-0.0212
-0.0213	-0.0012	0.0277	0.0170	0.0285	-0.0287
-0.0327	-0.0036	-0.0322	0.0432	-0.0032	0.0035
-0.0067	-0.0002	0.0028	-0.0181	-0.0507	0.0509
chinaUS.chinaUS	indiaUS.chinaUS	japanUS.chinaUS	EUUS.chinaUS	UKUS.chinaUS	canadaUS.indiaUS
0.0236	-0.1663	0.0034	-0.0679	0.0940	-0.0935
-0.2675	-0.0586	-0.1041	0.0190	-0.0087	0.0091
-0.2636	0.0712	0.0520	-0.3550	-0.0624	0.0635
-0.1080	0.0072	-0.3012	0.1945	0.1007	-0.1024
0.0657	0.0345	0.0313	-0.1671	-0.0458	0.0481
0.0957	-0.0173	0.0482	0.0922	0.0653	-0.0679
chinaUS.indiaUS	indiaUS.indiaUS	japanUS.indiaUS	EUUS.indiaUS	UKUS.indiaUS	canadaUS.japanUS
0.0453	0.0052	-0.0992	-0.0274	-0.0009	0.0000
0.0540	-0.0006	-0.0724	0.0359	0.0177	-0.0184
0.0246	0.0011	0.0256	-0.0466	-0.0108	0.0107
-0.0127	0.0011	0.0351	0.0061	-0.0300	0.0298
-0.0429	-0.0028	0.0511	0.0098	-0.0075	0.0085
0.0142	-0.0023	-0.0150	-0.0428	-0.0008	0.0004
chinaUS.japanUS	indiaUS.japanUS	japanUS.japanUS	EUUS.japanUS	UKUS.japanUS	canadaUS.EUUS
-0.0343	0.0152	0.0014	-0.0256	-0.0231	0.0231
-0.0020	-0.0009	0.0023	-0.0077	-0.0298	0.0300
0.0164	0.0011	0.0220	-0.0064	0.0433	-0.0440
-0.0250	-0.0036	-0.0036	0.0182	0.0037	-0.0040
0.0180	0.0010	0.0044	-0.0256	0.0078	-0.0079
0.0397	-0.0025	0.0245	-0.0074	0.0077	-0.0078
chinaUS.EUUS	indiaUS.EUUS	japanUS.EUUS	EUUS.EUUS	UKUS.EUUS	canadaUS.UKUS
1.0849	0.4456	2.2415	2.4309	3.3561	-3.3541
1.9317	0.3735	0.2586	0.4621	0.7860	-0.6647
-2.0394	-0.1991	0.1498	-0.6658	0.9835	-0.9460
-0.5302	0.4248	0.8361	-0.5528	0.0544	0.0292
1.3542	-0.0073	-1.5067	1.5001	-1.1147	1.1653
4.3631	-0.2067	1.8628	-2.5938	2.4067	-2.3028
chinaUS.UKUS	indiaUS.UKUS	japanUS.UKUS	EUUS.UKUS	UKUS.UKUS	
1.0853	0.4307	2.2045	2.4207	3.3241	-3.3223
1.9472	0.3710	0.2477	0.4838	0.8093	-0.6876
-2.0283	-0.1988	0.1547	-0.6701	0.9929	-0.9552
-0.5653	0.4275	0.8333	-0.5269	0.0286	0.0548
1.3580	-0.0069	-1.5264	1.5127	-1.0813	1.1320
4.3624	-0.2079	1.8669	-2.6077	2.3621	-2.2584

QUESTION 3: (using a package)

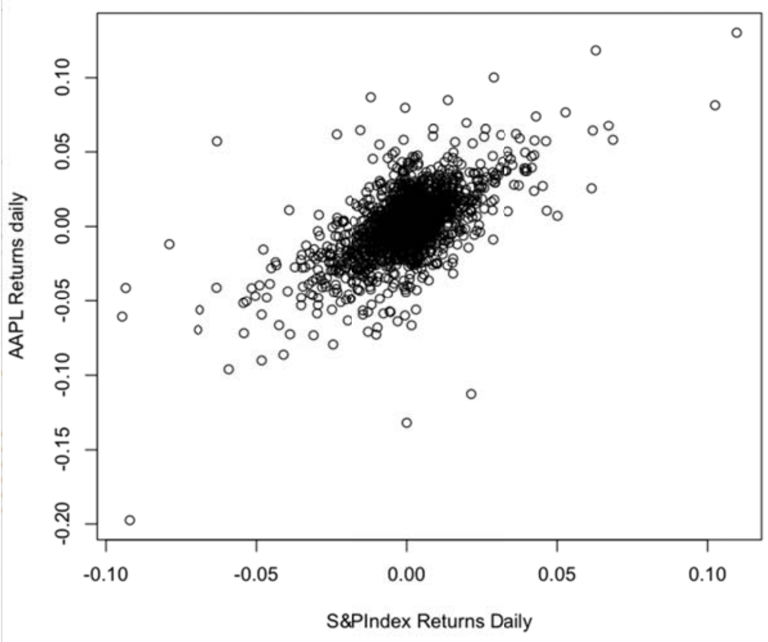
Below is the chart plot output from Quantmod, showing daily price of AAPL and the S&P500 index:



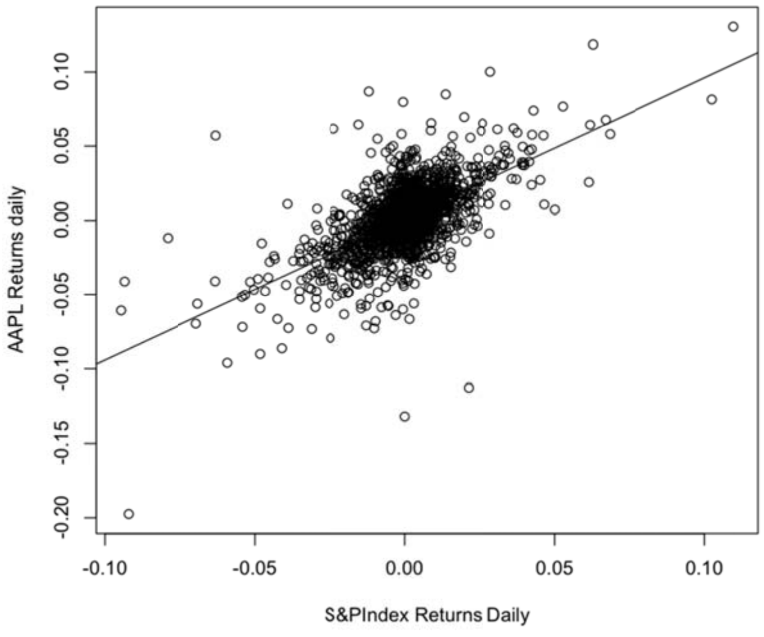
We use log returns to create a matrix of returns for both the target stock and the index:

```
stocks.ret <- log(stocks[2:n,]/stocks[1:(n-1),])
> head(stocks.ret)
            aapl          gspc
[1,]  0.021965570  0.0012275394
[2,] -0.007107179 -0.0061031642
[3,]  0.004823936  0.0022178536
[4,]  0.079857601 -0.0005168233
[5,]  0.046762330  0.0019384787
[6,] -0.012479496  0.0063198821
```

Below is a scatter plot of the stock return and index return. The scatterplot shows that there is most likely a significant positive correlation between AAPL stocks and the S&P500 Index. Based on the distribution of data points, AAPL looks to have a 1:1 return with the market, i.e. it has a beta of 1.



To quantify, we regress the stock return on the index return and report the results:



```
Call: lm(formula = stocks.ret[, 1] ~ stocks.ret[, 2])

Coefficients:

  (Intercept)  stocks.ret[, 2]

    0.0009595       0.9518297
```

The slope of the OLS fit is close to 1. This indicates that the β value is close to 1 and the movement of the AAPL stock is very much correlated to the movement of the S&P500 Index.

The intercept, α, is near-zero which is expected for a highly-traded, efficiently-priced asset such as AAPL stock.

To quantify risk, we look at the variation of the returns. More variation means higher range of possible outcomes, hence more risk:

```
mean(stocks.ret[,1])
[1] 0.001060516
> var(stocks.ret[,1])
[1] 0.0005307758
> mean(stocks.ret[,2])
[1] 0.0001061047
> var(stocks.ret[,2])
[1] 0.0002241071
```

We calculated the mean and variance of the both AAPL and GSPC's stock returns and found that the mean was around the same, but the variance was the much wider for AAPL than the index. This indicates that AAPL is more volatile that the index and is therefore riskier between the two.

To test for market efficiency, we can test to see if the lagged values of the independent variable contribute significantly to the regression of the dependent variable (AAPL). We first created a column of the one-day-lagged market returns, and then ran a regression including this as a new independent variable:

```
             aapl            gspc           gspc_lag
[1,]  0.021965570   0.0012275394   0.0000000000
[2,] -0.007107179  -0.0061031642   0.0012275394
[3,]  0.004823936   0.0022178536  -0.0061031642
[4,]  0.079857601  -0.0005168233   0.0022178536
[5,]  0.046762330   0.0019384787  -0.0005168233
```

```
[6,] -0.012479496   0.0063198821    0.0019384787

> res = lm(stocks.ret[,1] ~ stocks.ret[,2] + stocks.ret[,3])
Coefficients:
                  Estimate Std. Error                          t value  Pr(>|t|)
(Intercept)        0.0009544  0.0004381                2.179    0.0295 *
stocks.ret.rag[, 2] 0.9567642  0.0294766              32.458   <2e-16 ***
stocks.ret.rag[, 3] 0.0417314  0.0294786               1.416   0.1571
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.0181 on 1705 degrees of freedom
Multiple R-squared: 0.3833,       Adjusted R-squared: 0.3825
F-statistic: 529.8 on 2 and 1705 DF,  p-value: < 2.2e-16
```

From the linear model summary above, we see there is not a statistically significant relationship between the one-day-lagged return of the market to AAPL, thus we maintain our null hypothesis that this coefficient is zero.

From *Investopedia*, Autocorrelation is "[a] mathematical representation of the degree of similarity between a given time series and a lagged version of itself over successive time intervals. It is the same as calculating the correlation between two different time series, except that the same time series is used twice - once in its original form and once lagged one or more time periods. The term can also be referred to as "lagged correlation" or "serial correlation"."

To test for lagged autocorrelation, we can test if today's AAPL returns has a correlation with its own return from yesterday, thus is our dependent variable correlated to lagged values of itself? To test for this we regress today's returns against 4 lags:

```
      > res <- lm(logRets$aapl[5:n] ~ logRets$aapl[4:(n-1)] +
      logRets$aapl[3:(n-2)] + logRets$aapl[2:(n-3)] + logRets$aapl[1:(n-4)])
```

```
Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)            0.0002031  0.0020497   0.099   0.9211
logRets$aapl[4:(n - 1)] -0.1081782  0.0098778 -10.952  < 2e-16 ***
logRets$aapl[3:(n - 2)] -0.0619467  0.0099279  -6.240 4.56e-10 ***
logRets$aapl[2:(n - 3)] -0.0392087  0.0099279  -3.949 7.89e-05 ***
logRets$aapl[1:(n - 4)]  0.0219478  0.0098778   2.222   0.0263 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2075 on 10244 degrees of freedom
```

```
  (1 observation deleted due to missingness)
Multiple R-squared: 0.01576,     Adjusted R-squared: 0.01537
F-statistic:    41 on 4 and 10244 DF,  p-value: < 2.2e-16
```

We can see from the results that the first 4 lags are all arguably statistically significant based on the p-values. However if we perform a similar test on the entire S&P500 index, there is no such correlation:

```
Coefficients:
                        Estimate Std. Error t value Pr(>|t|)
(Intercept)            1.937e-05  2.184e-03   0.009    0.993
logRets$gspc[4:(n - 1)] -6.653e-02  9.880e-03  -6.734 1.74e-11 ***
logRets$gspc[3:(n - 2)] -6.924e-03  9.902e-03  -0.699    0.484
logRets$gspc[2:(n - 3)] -7.529e-03  9.902e-03  -0.760    0.447
logRets$gspc[1:(n - 4)]  5.594e-03  9.880e-03   0.566    0.571
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2211 on 10244 degrees of freedom
  (1 observation deleted due to missingness)
Multiple R-squared: 0.004504,   Adjusted R-squared: 0.004115
F-statistic: 11.59 on 4 and 10244 DF,  p-value: 2.187e-09
```

The single stock AAPL had a much higher degree of autocorrelation when compared to the market as a whole. Given efficient market theory, we would expect the single asset to demonstrate momentum instead of the entire market.

We can automate this test for autocorrelation using the Durbin-Watson test in the r “car” library:

```
durbin.watson(logRets$aapl,max.lag=10)
[1] 2.201417 2.096316 2.059938 1.935752 1.999292 2.038623 2.058636 1.995265
 [9] 1.977384 1.943195
```

The DW values which are <2 or >2 could indicate potential auto-correlation, and as expected the first four lags show potential based on the DW test.

QUESTION 4 (using Twitter)

- We will use the twitteR package (http://cran.r-project.org/web/packages/twitteR/index.html)

- Pick some twitter feeds and extract tweets from them.

After authorizing using the steps outline in the lecture notes, we are able to search for tweets in a few different ways:

```
#### some examples of twitter searches, retrieving m tweets
m <- 500

# search for m tweets based on string
#tweets <- searchTwitter("#beer", n=m)
#tweets <- searchTwitter("#GOOG", n=m)

## Search between two dates
# tweets <- searchTwitter("charlie sheen", since="2011-03-01", until="2011-
03-02")

## geocoded results
#tweets <- searchTwitter("big data", geocode="37.3628,-121.9292,50mi", n=m)

# get m tweets from user
#user <- "username"
#userInfo <- getUser(user,cainfo="cacert.pem")
#tweets <- userTimeline(user, n=m,cainfo="cacert.pem")
```

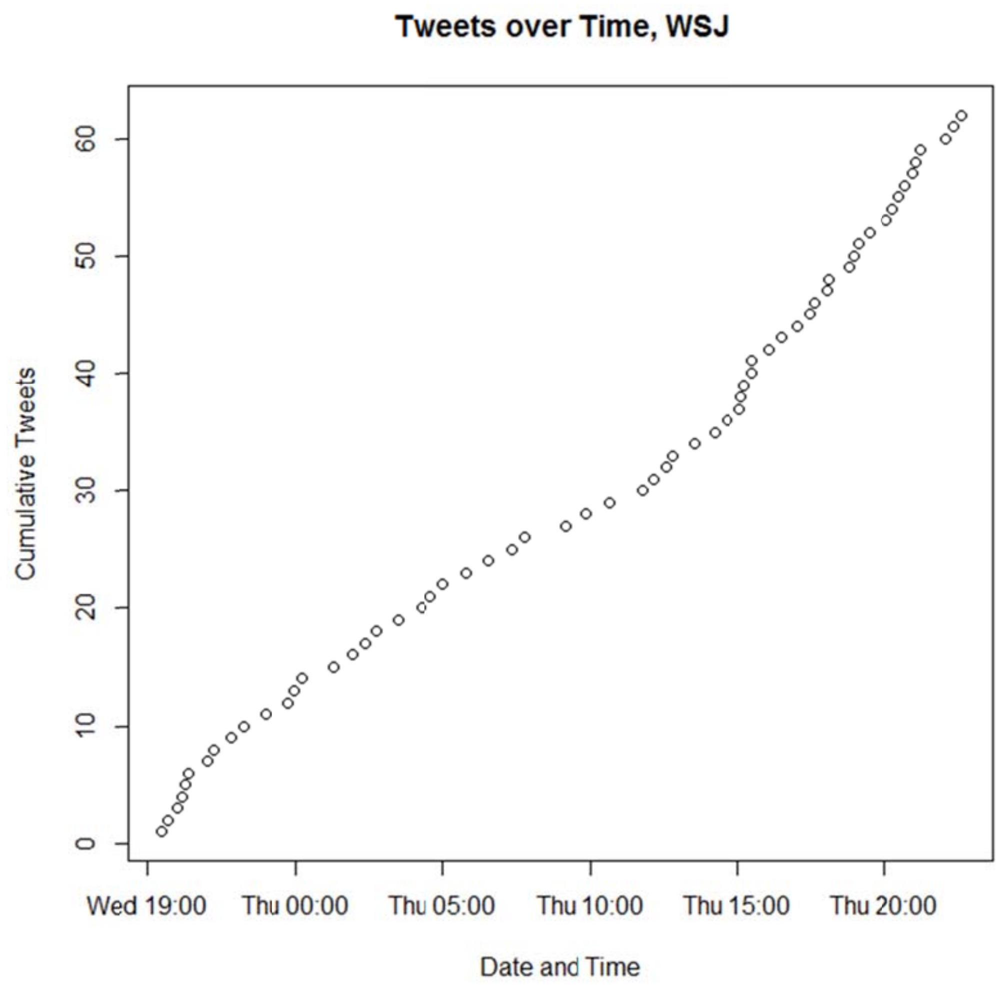
It is very helpful to use the twitteR function twListToDF(), which will create a data frame from the returned tweets:

```
# create a data frame
tweets_df = twListToDF(tweets)
```

In the following analysis, we choose to follow the @WSJ user:

```
# get m tweets from user
user <- "WSJ"
userInfo <- getUser(user,cainfo="cacert.pem")
tweets <- userTimeline(user, n=m,cainfo="cacert.pem")

# create a plot of time series of cumulative tweets:
n <- length(tweets_df$created)
y <- seq(1,n)
y <- y[n:1]
plot(y ~ tweets_df$created, main="Tweets over Time, WSJ", xlab="Date and
Time",ylab="Cumulative Tweets")
```



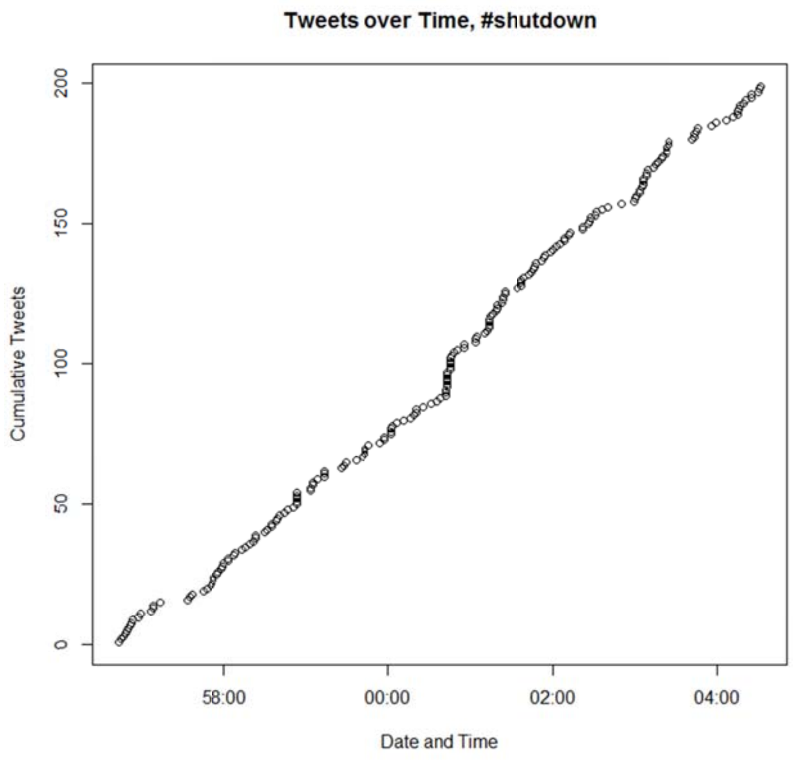
The Wall Street Journal is a regular tweeter, as displayed on the time series plot above. On average they tweeted *2.3 times per hour* over this analysis window:

```
# what is the time window for which we got tweets?
timeWindow <-
difftime(max(tweets_df$created),min(tweets_df$created),units="hours")
# what is the average frequency of tweets per hour
tweetFreq <- length(y)/as.numeric(timeWindow)
```

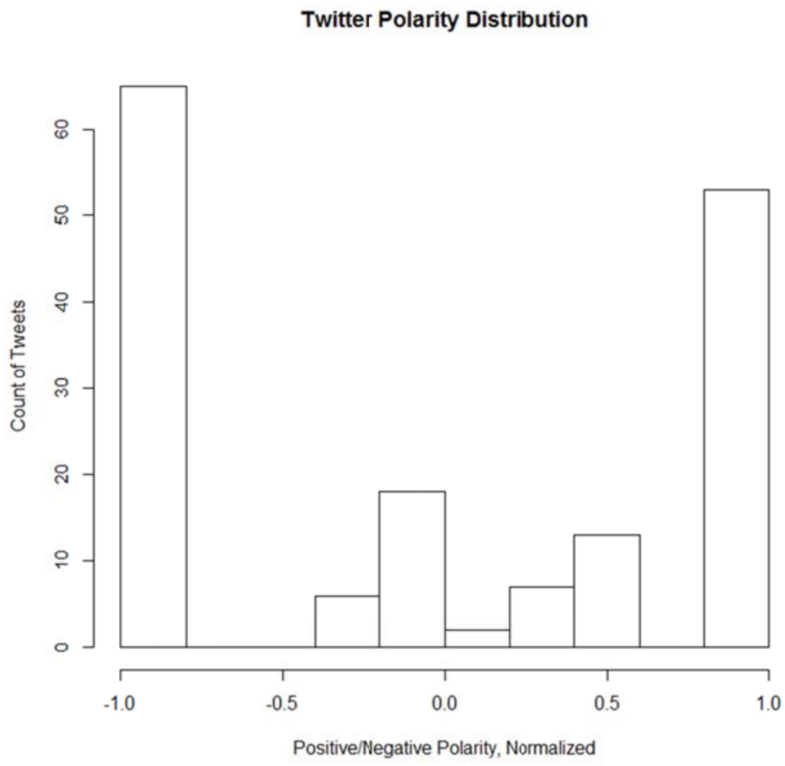
Next, we extracted tweets based on a hashtag *#shutdown*:

```
tweets <- searchTwitter("#shutdown", n=m)
```

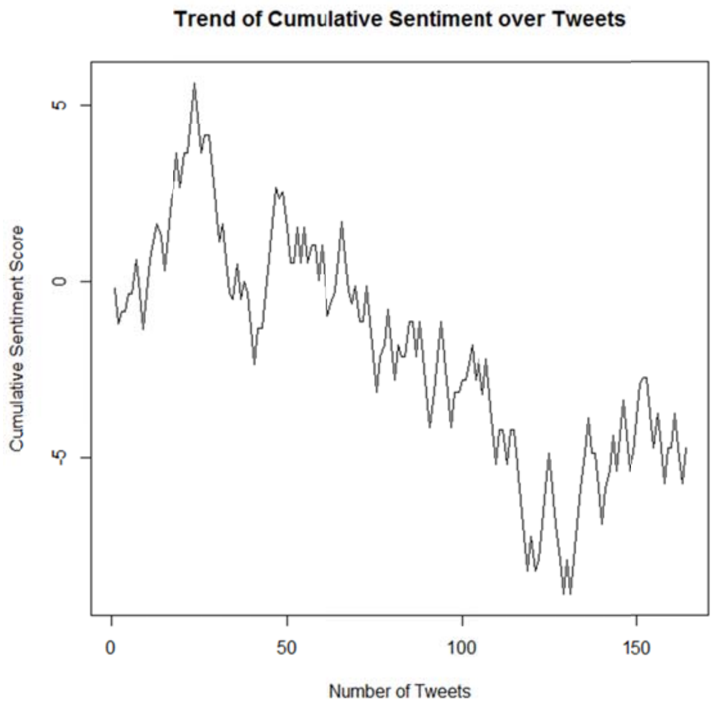
On average there were *1527.5 tweets per hour* using the hashtag #shutdown. The last 200 tweets (the max we could pull) only happened in the last 10 minutes or so:



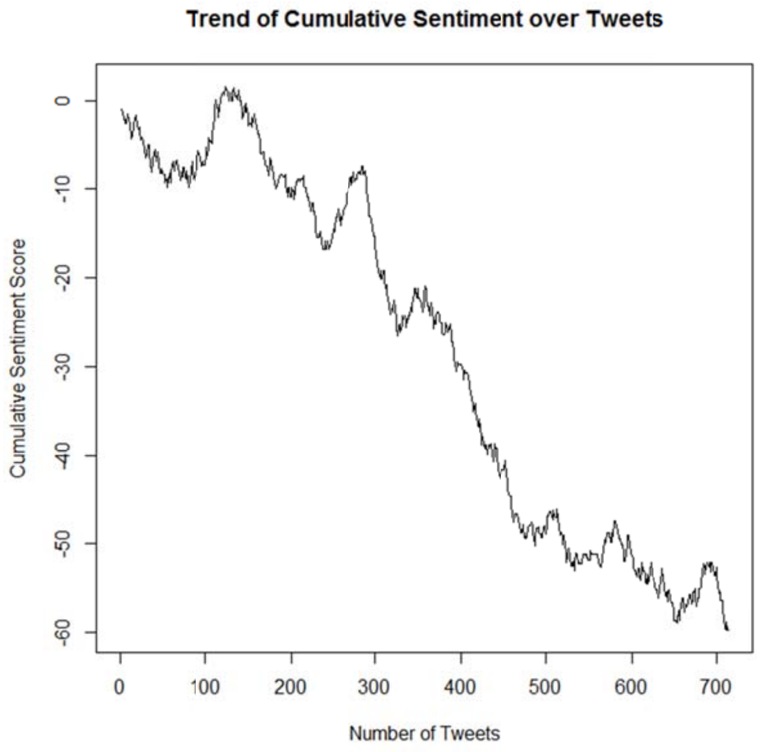
To further analyze this tweet stream, we used the Harvard General Inquirer to score the sentiment of tweets from -1 to 1. Each tweet is cleaned and compared to the dictionary, and assigned a score from -1 to 1. For example, here is the distribution of 164 most recent #shutdown tweets:



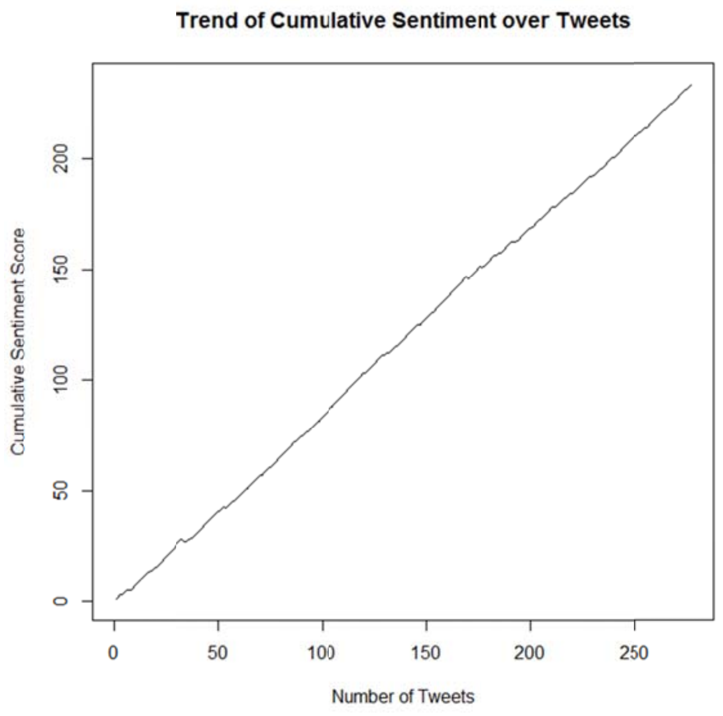
We can also see how this sentiment has trended over time, with a slight uptick towards the end. This is approximately 15 minutes worth of tweets:



Closely associated, here is the recent cumulative sentiment for search term “boehner”:



Compare this against a cumulative tweet sentiment for something positive, like "candy":



Appendix – Code for problems

#Q1

```
# read in the txt file
# Customer ID, Age, Sex, Location, Action, Time the action was performed, Is
the remote hardware setup with the app?
data <- read.csv("e:/datamining/Peel/0000_part_00.csv", header=FALSE,
col.names=c("ID", "Age", "Sex", "Location", "Action", "Time", "Hardware" ))
data1 <- read.csv("e:/datamining/Peel/0001_part_00.csv", header=FALSE,
col.names=c("ID", "Age", "Sex", "Location", "Action", "Time", "Hardware" ))
data2 <- read.csv("e:/datamining/Peel/0002_part_00.csv", header=FALSE,
col.names=c("ID", "Age", "Sex", "Location", "Action", "Time", "Hardware" ))
data3 <- read.csv("e:/datamining/Peel/0003_part_00.csv", header=FALSE,
col.names=c("ID", "Age", "Sex", "Location", "Action", "Time", "Hardware" ))

# bind all rows into a singe frame
data <- rbind(data,data1,data2, data3)

# change Time column to Date and time type
data$Time <- as.POSIXct(data$Time)

# Sort By ascending ID then by ascending time
dataSorted <- data[with(data, order(ID, Time)), ]
summary(dataSorted)

# How many unique IDs are there? Create a vector of users
IDlist <- unique(data$ID)
n <- length(IDlist)
# 12991 unique user IDs


# initialize a user data frame
user <-NULL

# initialize data types
user$Location <- factor(levels=levels(dataSorted$Location))
user$Sex <- factor(levels=levels(dataSorted$Sex))
user$Hardware <- factor(levels=levels(dataSorted$Hardware))
user$FirstAction <- as.POSIXct(dataSorted$Time[1])
user$LastAction <- as.POSIXct(dataSorted$Time[1])
user$TotalTime <- numeric(1)
user$ID <- IDlist

# create STATIC user data frame with demographics and characteristics from
master data
for (i in 1:n) {
# get a subset of data and store in a temporary "oneUserData" frame
oneUserData <- subset(dataSorted, dataSorted$ID==IDlist[i])

# generate single data points for each user from complete data, usually by
taking the first entry of the subset from each column
user$Age[i] <- oneUserData$Age[1]
user$Location[i] <- oneUserData$Location[1]
user$Sex[i] <- oneUserData$Sex[1]
user$Hardware[i] <- oneUserData$Hardware[1]
user$LifeActions[i] <- length(oneUserData$Action)
user$FirstAction[i] <- min(oneUserData$Time)
user$LastAction[i] <- max(oneUserData$Time)
user$TotalTime[i] <- difftime(user$LastAction[i],user$FirstAction[i],
units="days")
}

# re-assemble user data back into a single data frame
user <- data.frame(user)
str(user)
head(user)

### Usage information
hist(user$TotalTime, main="Total Activity Span", xlab="Time from First to
Last Usage, days")

#### AGE INFORMATION
# replace 0 or 100 with NA
user$Age[user$Age==0] <- NA
user$Age[user$Age==100] <- NA
# remove NA values into a clean vector
withAge <- na.omit(user$Age)
# How many users have specified an age as percentage of total?
haveAge <- length(withAge)/length(user$Age)
print(haveAge)
print("report Age")
hist(withAge, xlab="Age, years", ylab="Count", main="Age Distribution of
Users Reporting")
lines(h=mean(withAge))
mean(withAge)
median(withAge)
mode(withAge)


#### SEX INFORMATION
# replace blank with NA
user$Sex[user$Sex==""] <- NA
# Create a new vector with just the values input and plot histogram
withSex <- na.omit(user$Sex)
haveSex <- length(withSex)/length(user$Sex)
print(haveSex)
print("report Sex")
# Pie Chart from data frame with Appended Sample Sizes
mytable <- table(withSex)
lbls <- paste(names(mytable), "\n", mytable, sep="")
pie(mytable, labels = lbls,
       main="User Sex, of Those Reporting")

#### User hardware information
haveHardware <-
length(user$Hardware[user$Hardware=="Y"])/length(user$Hardware)
print(haveHardware)
print("Have Hardware")
mytable <- table(user$Hardware)
lbls <- paste(names(mytable), "\n", mytable, sep="")
pie(mytable, labels = lbls,
       main="User Hardware?")
```

```
# Location Information
user$Location[user$Location==""]<-NA
ActualLocations <- na.omit(user$Location)
haveLocation <- length(ActualLocations)/length(user$Location)
print(haveLocation)
print("Have Location")
ActualLocations <- sort(ActualLocations)
mytable <- table(ActualLocations)

# Action information
bins=seq(0,300,by=10)
hist(user$LifeActions[user$LifeActions<300], breaks=bins, freq=FALSE,
xlab="Lifetime # of Actions", ylab="Density of Users", main="Distribution of
Lifetime User Actions")
```

#Q2

```
#QUESTION 2 (Find relationships)
#Go to the web page for Federal Reserve data
at:http://research.stlouisfed.org/
#Download exchange rate data for 6 countries of your choice versus the US
dollar. Summarize this data and plot the series. Provide a brief description.
library("quantmod")
library("tseries")
library("car")
library("stats")

date_range <- "2000-01-01::2013-10-15"
# Canada, China, India, Japan  to $1 US
getSymbols(c("DEXCAUS", "DEXCHUS", "DEXINUS", "DEXJPUS", "DEXUSEU",
"DEXUSUK"), src='FRED')

# convert xts from getSymbols to numeric
canadaUS <- as.numeric(DEXCAUS[date_range])
chinaUS <- as.numeric(DEXCHUS[date_range])
indiaUS <- as.numeric(DEXINUS[date_range])
japanUS <- as.numeric(DEXJPUS[date_range])
USEU <- as.numeric(DEXUSEU[date_range])
USUK <- as.numeric(DEXUSUK[date_range])
EUUS <- 1/USEU
UKUS <- 1/EUUS

# create data frame
fxrates <- data.frame(canadaUS, chinaUS, indiaUS, japanUS, EUUS, UKUS)
fxrates <- na.omit(fxrates)

par(mfrow=c(3,2))
plot(fxrates$canadaUS, ylab="Candadian Dollar per $1 US", type="l")
plot(fxrates$chinaUS, ylab="China per $1 US", type="l")
plot(fxrates$indiaUS, ylab="India Dollar per $1 US", type="l")
plot(fxrates$japanUS, ylab="Japan Yen per $1 US", type="l")
plot(fxrates$EUUS, ylab="Euro per $1 US", type="l")
plot(fxrates$UKUS, ylab="UK per $1 US", type="l")
```

```
#Present the correlation table of exchange rates.
cor(fxrates)

#Present the correlation table of changes in exchange rates.
# create matrix of changes (percent changes)
n <- dim(fxrates)[1]
fxrates.change <- (fxrates[2:n,]-fxrates[1:(n-1),])/fxrates[1:(n-1),]
cor(fxrates.change)

#Pick your favorite pair of exchange rates and say whether the correlation of
exchange rate changes in statistically significant or not. How would you
establish this?
#Regress one series of exchange rate changes on another, and describe the
output (R-square, t-statistics, f-statistic, etc.) Is there any economic
conclusion that you can infer from the regression?
res <- lm(fxrates.change$canadaUS ~ fxrates.change$EUUS)
summary(res)

# Using the data series you have, can you build a model that uses lagged
values of changes in exchange rates to predict future exchange rate changes
with a high level of accuracy? (This requires some experimentation and
playing with the data. Ideally program R to run all possible cases.)
# this is one rate against all the others, current rate
n <- length(fxrates.change$canadaUS)
res <- lm(fxrates.change$canadaUS ~ fxrates.change$chinaUS +
fxrates.change$indiaUS + fxrates.change$japanUS +
      fxrates.change$EUUS + fxrates.change$UKUS)
summary(res)

# this is one rate against all the others, ;agged once
res <- lm(fxrates.change$canadaUS[2:n] ~ fxrates.change$chinaUS[1:(n-1)] +
fxrates.change$indiaUS[1:(n-1)] + fxrates.change$japanUS[1:(n-1)] +
      fxrates.change$EUUS[1:(n-1)] + fxrates.change$UKUS[1:(n-1)])
summary(res)



# Use a DurbinWatson to test for auto-correlation of Canada:US exchange rate
res <- lm(fxrates.change[2:n,1] ~ fxrates.change[1:(n-1),1])
summary(res)
durbin.watson(res,max.lag=10)


#Run a vector autoregression (VAR) on the data of exchange rate changes that
you have. What inferences can you make from the results?
var6 = ar(fxrates.change,aic=TRUE,order=6)
var6$order
var6$ar


#Q3

#QUESTION 2 (Find relationships)
#Go to the web page for Federal Reserve data
at:http://research.stlouisfed.org/
```

```
#Download exchange rate data for 6 countries of your choice versus the US
dollar. Summarize this data and plot the series. Provide a brief description.
library("quantmod")
library("tseries")
library("car")
library("stats")


date_range <- "2000-01-01::2013-10-15"
# get symbols
getSymbols(c("AAPL","^GSPC"))

# convert xts from getSymbols to numeric
aapl <- as.numeric(AAPL[date_range])
gspc <- as.numeric(GSPC[date_range])

# create data frame
stocks <- data.frame(aapl,gspc)
stocks <- na.omit(stocks)

#Present the correlation table of changes in exchange rates.
# create matrix of changes (percent changes)
n <- dim(stocks)[1]
rets <- (stocks[2:n,]-stocks[1:(n-1),])/stocks[1:(n-1),]
logRets <- log(stocks[2:n,]/stocks[1:(n-1),])

# linear regression of returns
res <- lm(logRets$aapl ~ logRets$gspc)
summary(res)


# Using the data series you have, can you build a model that uses lagged
values of changes in exchange rates to predict future exchange rate changes
with a high level of accuracy? (This requires some experimentation and
playing with the data. Ideally program R to run all possible cases.)

res <- lm(logRets$aapl[5:n] ~ logRets$aapl[4:(n-1)] + logRets$aapl[3:(n-2)] +
logRets$aapl[2:(n-3)] + logRets$aapl[1:(n-4)])
summary(res)

res <- lm(logRets$gspc[5:n] ~ logRets$gspc[4:(n-1)] + logRets$gspc[3:(n-2)] +
logRets$gspc[2:(n-3)] + logRets$gspc[1:(n-4)])
summary(res)

# Use a DurbinWatson to test for auto-correlation
dwres <- durbin.watson(logRets$aapl,max.lag=10)


n <- length(logRets$aapl)
res <- lm(logRets$aapl[2:n] ~ logRets$aapl[1:(n-1)])
durbin.watson(res,max.lag=10)
```

#Q4

```
library("plyr")
library("twitteR")
library("ROAuth")
library("RCurl")
library("tm")

## resoution for SSL verification issue - disable it
opts <- list(
  capath = system.file("CurlSSL", "cacert.pem", package = "RCurl"),
  ssl.verifypeer = FALSE);
options(RCurlOptions = opts)


### SIMPLE HARVARD POS NEGATIVE DICTIONARY
# Read in HI Dictionary from TXT  #every word is tagged by psychometric tags
HIDict = readLines("e:/Dropbox/FNCE 696/Data/inqdict.txt")
# create a dictionary of Pos-tagged words
dict_pos <- HIDict[grep("Pos",HIDict)]    #return index for every line that
contains "Pos"
poswords <- NULL
for (s in dict_pos) {
      s <- strsplit(s,"#")[[1]][1]        #split at hash to remove instance
ID of same words
      poswords <- c(poswords, strsplit(s," ")[[1]][1])      #returns list,
take first element of that array
}
poswords <- tolower(poswords)
# create a dictionary of Neg-tagged words
dict_neg <- HIDict[grep("Neg",HIDict)]    #return index for every line that
contains "Pos"
negwords=NULL
for (s in dict_neg) {
      s <- strsplit(s,"#")[[1]][1]        #split at hash to remove instance
ID of same words
      negwords <- c(negwords, strsplit(s," ")[[1]][1])      #returns list,
take first element of that array
}
negwords <- tolower(negwords)
######################


#### AUTHORIZATION CODE
download.file(url="http://curl.haxx.se/ca/cacert.pem", destfile="cacert.pem")
cKey <- REMOVED
cSecret <- REMOVED
reqURL <- "https://api.twitter.com/oauth/request_token"
authURL <- "https://api.twitter.com/oauth/authorize"
accURL <- "https://api.twitter.com/oauth/access_token"
cred <- OAuthFactory$new(consumerKey=cKey,
consumerSecret=cSecret,requestURL=reqURL,
      accessURL=accURL,authURL=authURL)
cred$handshake(cainfo="cacert.pem")
registerTwitterOAuth(cred)
save(list="cred", file="twitteR_credentials")
load("twitteR_credentials")
registerTwitterOAuth(cred)



#### some examples of twitter searches, retrieving m tweets
```

```
m <- 1000

# search for m tweets based on string
#searchTwitter("#beer", n=m)
tweets <- searchTwitter("#GAP", n=m)

## Search between two dates
#searchTwitter("charlie sheen", since="2011-03-01", until="2011-03-02")

## geocoded results
#tweets <- searchTwitter("big data", geocode="37.3628,-121.9292,50mi", n=m)

# get m tweets from user
#user <- "WSJ"
#userInfo <- getUser(user,cainfo="cacert.pem") #Works correctly
#tweets <- userTimeline(user, n=m,cainfo="cacert.pem")

# create a data frame
tweets_df = twListToDF(tweets)

n <- length(tweets_df$created)
y <- seq(1,n)
y <- y[n:1]
plot(y ~ tweets_df$created, main="Tweets over Time, #shutdown", xlab="Date
and Time",ylab="Cumulative Tweets")

# what is the time window for which we got tweets?
timeWindow <-
difftime(max(tweets_df$created),min(tweets_df$created),units="hours")
# what is the average frequency of tweets per hour
tweetFreq <- length(y)/as.numeric(timeWindow)


# pull out only text from twitter data frame
tweets <- lapply(tweets, function(t)t$getText())


Score <- NULL

# score tweets for sentiment
for (i in 1:m) {
      text <- tweets[i]
      txtCLEAN <- tolower(text)
      txtCLEAN <- removePunctuation(txtCLEAN)
      txtCLEAN <- strsplit(txtCLEAN, " ")
      txtCLEAN <- unlist(txtCLEAN)

## POSITIVE WORD COUNT
posmatch <- match(txtCLEAN, poswords)     #take two vectors/arrays and give
back matches
numPosMatch <- length(posmatch[which(posmatch>0)])

#### NEGATIVE WORD COUNT
negmatch <- match(txtCLEAN, negwords)
numNegMatch <- length(negmatch[which(negmatch>0)])

Score[i] <- (numPosMatch - numNegMatch)/(numPosMatch + numNegMatch)
#print(Score[i])
}

# Remove NaN results
Score <- na.omit(Score)
# plot distribution of polarity score
hist(Score, xlab="Positive/Negative Polarity, Normalized", ylab="Count of
Tweets", main="Twitter Polarity Distribution")
mean(Score)

m <- length(Score)
Score <- Score[m:1]

netScore <- NULL

for (n in 1:length(Score)) {
netScore[n] <- sum(Score[1:n])
}
plot(netScore, type="l", main="Trend of Cumulative Sentiment over Tweets",
xlab="Number of Tweets", ylab="Cumulative Sentiment Score")w
```