# Predictions of Stock Vaules using LinkedIn Data

*Suren Rathnayake*

*29 July 2019*

### Aim

The main aim of this study is to preidct the stock prices using information available in LinkedIn.

### Data

The data used in this study consisted of the LinkedIn data from https://blog.thedataincubator.com/tag/data-sources/ and Stock Price data from the Yahoo Finance.

### Loading the LinkedIn data

```r
library(data.table)
library(tidyverse)
library(tidyr)
library(quantmod)

dlink <- fread("temp_datalab_records_linkedin_company.csv")
#head(dlink)
```

## Preprocessing the Data

```r
vars <- c("company_name", "industry")
dlink[, (vars) := lapply(.SD, tolower), .SDcols = vars]
dlink[, (vars) :=
    lapply(.SD, function (x) gsub("[[:punct:]]", "", x)), .SDcols = vars]

#length(unique(dlink$dataset_id))
# [1] 4610

#length(unique(dlink$company_name))
# [1] 4991
```

Format the numeric and date columns.

```r
cols_convert_to_num <- c("followers_count", "employees_on_platform")
dlink[,(cols_convert_to_num):= lapply(.SD, as.numeric),
                                        .SDcols = cols_convert_to_num]

cols_convert_to_date <- c("date_added",  "date_updated")
dlink[,(cols_convert_to_date) := lapply(.SD, function(x)
    as.Date(as.POSIXct(x, format = "%Y-%m-%d %H:%M:%S"), format='%Y-%m-%d')),
            .SDcols = cols_convert_to_date]
```

## Getting Stock Values Data

The `quantmod` provides a number functions to access and handle stock trading data. The following function

- downloads the data for a given company,
- align with the LinkedIn data based on the `added_date`, and
- extract interesting information for the particular company, as well as collation of those belonning to companies in the same industry.

```r
collate_data <- function(dlink, name, sym, src = "yahoo") {

    # LinkedIn Data
    linked_data <- dlink[company_name == name, .(date_added, company_name,
                                    followers_count, employees_on_platform)]

  # handle duplicates: multiple recodeds for same day
  vars <- c("employees_on_platform", "followers_count")
  linked_data[, (vars) := lapply(.SD, function(y) as.vector(mean(y))),
                by = date_added, .SDcols = vars]
  linked_data <- linked_data[!duplicated(date_added)]

  # get stock data for the duration of linkedin data
  date_range <- range(linked_data$date_added)
  getSymbols(sym, src = src, from = date_range[1], to = date_range[2])
  stock_data <- get(sym)

  # combine the data
  comm_dates <- linked_data$date_added[linked_data$date_added %in%
                                      as.Date(time(stock_data))]

    linked_data <- linked_data[date_added %in% comm_dates]
  stock_data <- stock_data[as.Date(time(stock_data)) %in% comm_dates]

  # "Open"     "High"     "Low"      "Close"    "Volume" "Adjusted"
  dt <- data.table(employees_on_platform = linked_data$employees_on_platform,
                                      followers_count = linked_data$followers_count,
                                      stock_close = as.numeric(stock_data[, 4]),
                                      date_added = comm_dates)

  # add the data for the same industry
  cindustry <- unique(dlink[company_name == name]$industry)
  # remove ""
  cindustry <- cindustry[!cindustry %in% ""]
  ccompanies <- unique(dlink[industry %in% cindustry]$company_name)

  ind_data <- dlink[company_name %in% ccompanies, .(date_added, company_name,
                                   followers_count, employees_on_platform)]

  # dplyer seems to be more conveniet here
  ind_data <- ind_data %>% group_by(date_added, company_name) %>%
       mutate(ind_followers_count = mean(followers_count),
              ind_employees_on_platform = mean(employees_on_platform)) %>%
       group_by(date_added) %>%
       summarise(ind_followers_count = sum(ind_followers_count),
              ind_employees_on_platform = sum(ind_employees_on_platform))
  setDT(ind_data)
  # merge the data sets
  ind_data <- ind_data[!duplicated(date_added)]
  setkey(dt, date_added)
  setkey(ind_data, date_added)
  dt <- merge(dt, ind_data, by = "date_added")
  dt
}
```

## Getting Stock Data

Getting stock data require knowing the symbol used by Yahoo for each company. The data doesn't always download. So, I have stored data for a number of companies.

```r
# Download the data
company <- c('ibm', 'mastercard', 'visa', 'oracle', 'accenture', 'cognizant',
                     'cdw', 'fiserv', 'gartner', 'netapp', 'sabre corporation',
                     'cerner corporation', 'ctripcom', 'wipro', 'xerox', 'iron mountain',
                     'avnet', 'costar group', 'arrow electronics',
                     'sify technologies limited', 'digital realty', 'ericsson', 'csc',
                     'genpact', 'pfsweb', 'pitney bowes', 'instructure',  'fis',
                     'general dynamics information technology', 'thomson reuters', 'verizon',
                     'amplify', 'verisign', 'cgi', 'motorola solutions', 'transunion',
                     'athenahealth', 'cyrusone', 'ncr corporation', 'goldman sachs',
                     'united technologies', 'novo nordisk', 'apple')

sym <- c("IBM", "MA", "V", "ORCL", "ACN", "CTSH", "CDW", "FISV", "IT",
                "NTAP", "SABR", "CERN", "CTRP", "WIT", "XRX", "IRM", "AVT", "CSGP", "ARW",
                "SIFY", "DLR", "ERIC", "CSC","G", "PFSW", "PBI", "INST", "FIS", "GD", "TRI",
                "VZ", "AMPY",  "VRSN", "GIB", "MSI", "TRU", "ATHN", "CONE", "NCR", "GS",
                "UTX", "NVO", "AAPL")
```

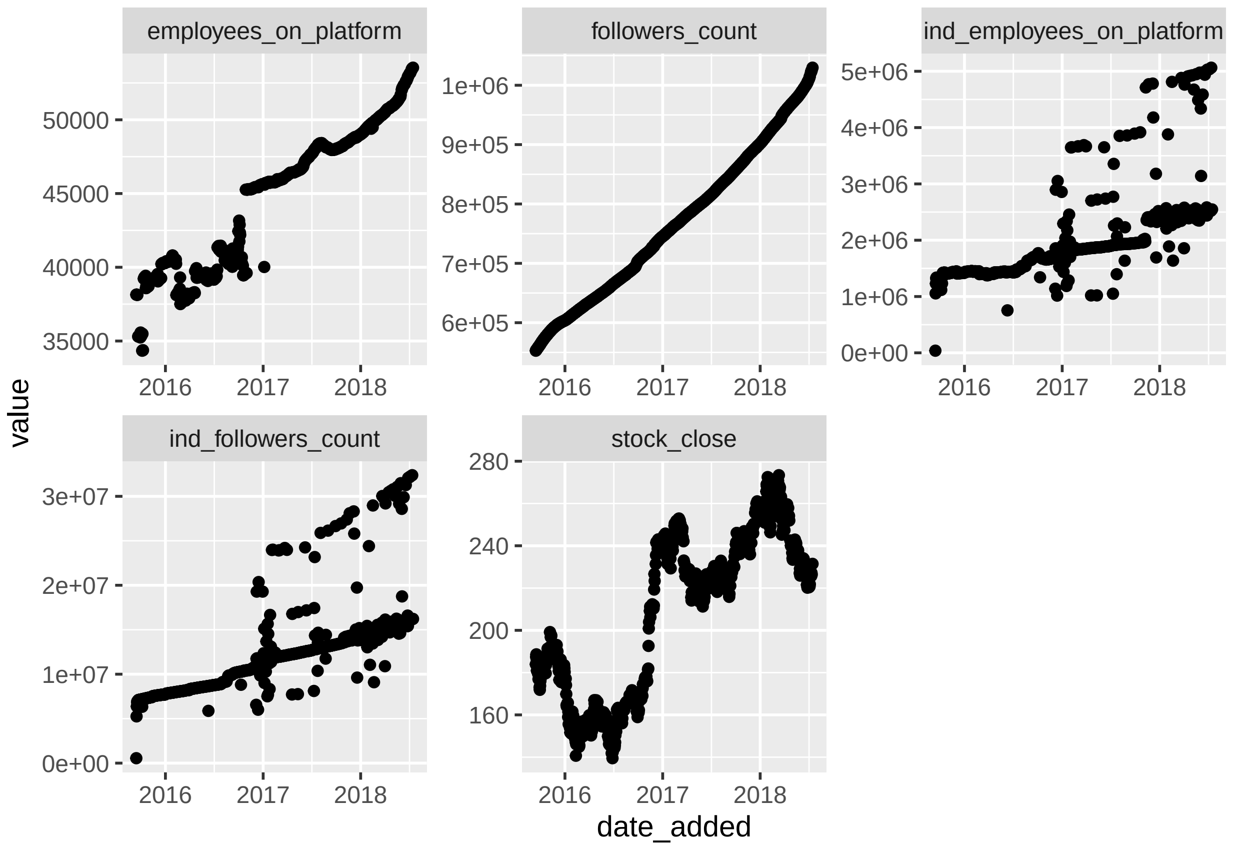
Download the data and save them along with corresponding LinkedIn data.

```r
for (j in 1:length(company)) {

    dt <- collate_data(dlink, company[j], sym[j])
    dt$company_name <- company[j]
    write.csv(file = paste0(company[j], ".csv"), dt, row.names = FALSE)
}
```

## Inspecting One Dataset (Goldman Sachs)

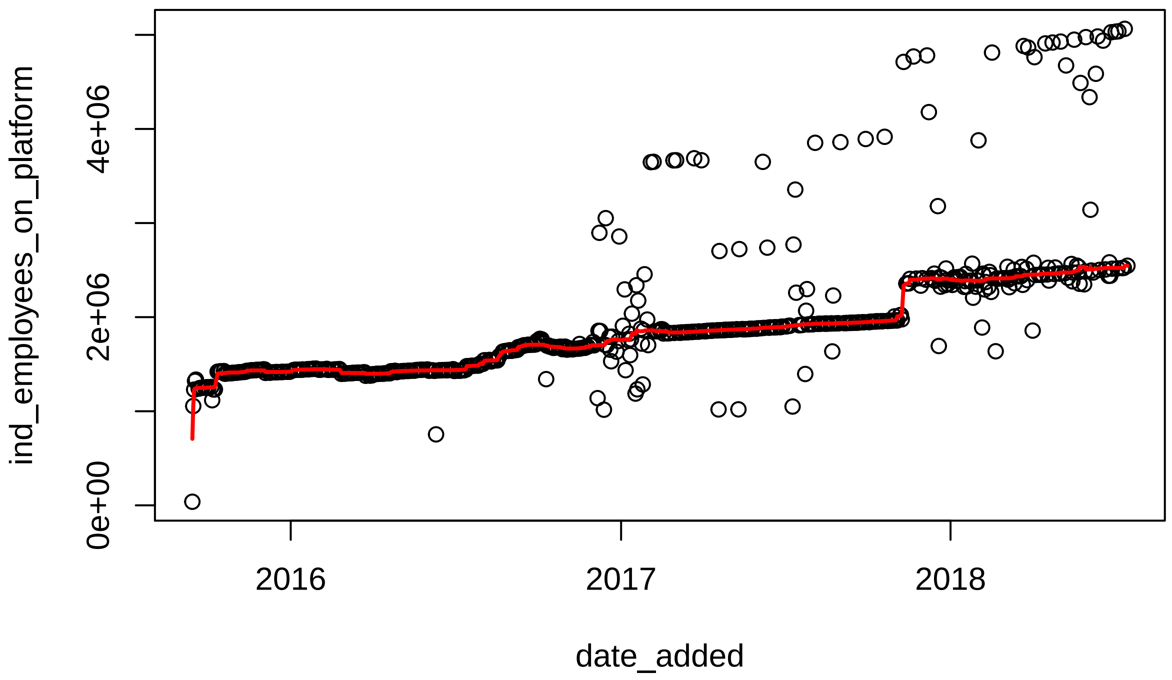
Here we look at the Goldman Sachs company.

```r
dt <- collate_data(dlink, name = "goldman sachs", sym = "GS")

tall_df <- dt %>% gather(type, value, -date_added)
ggplot(tall_df, aes(x = date_added, y = value)) + geom_point() +
  facet_wrap(. ~ type, scales="free")
```



Some of the variables in the data show unexplainable deviations. For example, it is difficult to believe some of the sudden changes in the total number employees for a perticular industry. Let's try a median filter.

```r
median_filter <- function(x, n = 21){runmed(x, n)}
# filter on total number of employee on comanies in same industry as Golsman Sachs
dt[, plot(ind_employees_on_platform ~ date_added)]
dt[, points(median_filter(ind_employees_on_platform) ~ date_added,
    type = "l", lwd = 2, col = "red")]
```

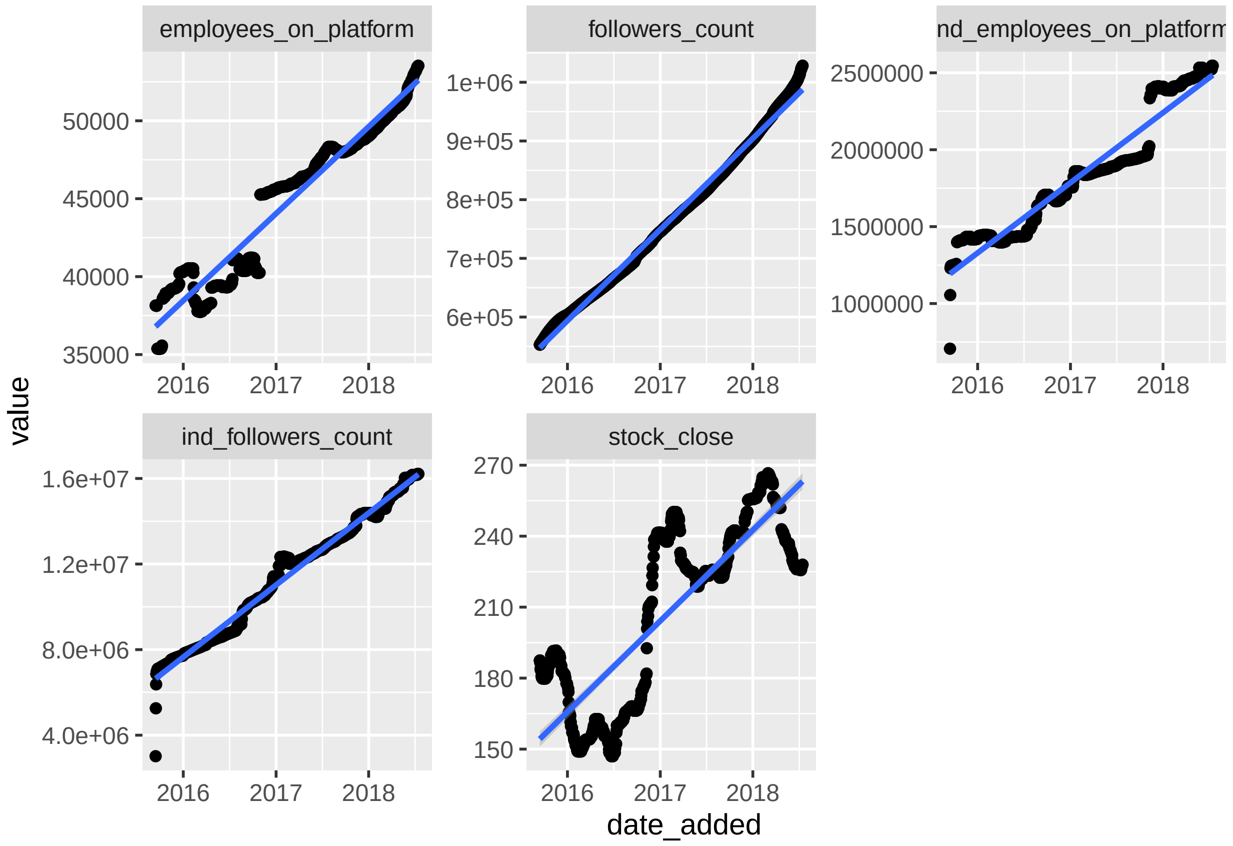


Seems to work fine. Apply the median filter to the data.

```
vars <- colnames(dt)[-1]
dt[, (vars) := lapply(.SD, function (y) as.vector(median_filter(y))),
                    .SDcols = vars]
```

Plot the extracted variables to visualize the data to see a relationship.

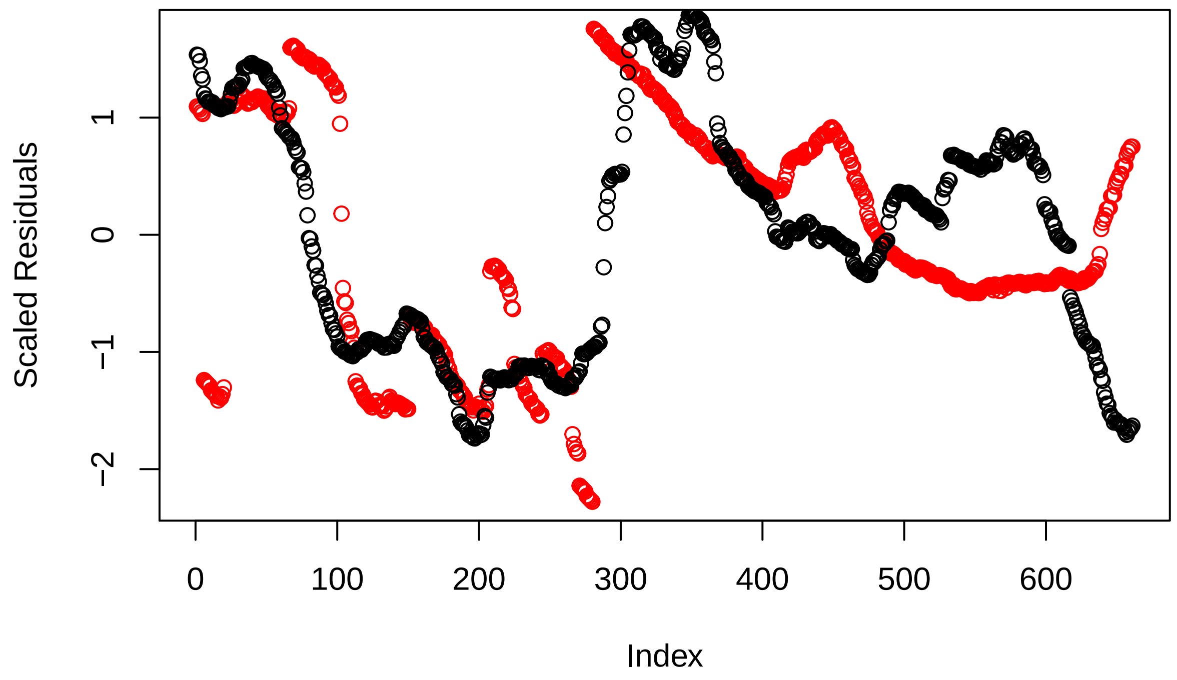
```
tall_df <- dt %>% gather(type, value, -date_added)
ggplot(tall_df, aes(x = date_added, y = value)) + geom_point() +
  geom_smooth(method = "lm") + facet_wrap(. ~ type, scales="free")
```



Since there appear to be a trend where everything increases with time, the stock closing value would show a positive relationship with any of the other four variables - even if they are meaningless.

Remove the trend and see if the changes in the increase in the number of employees with time can predict the changes in increase in the stock closing value with time.

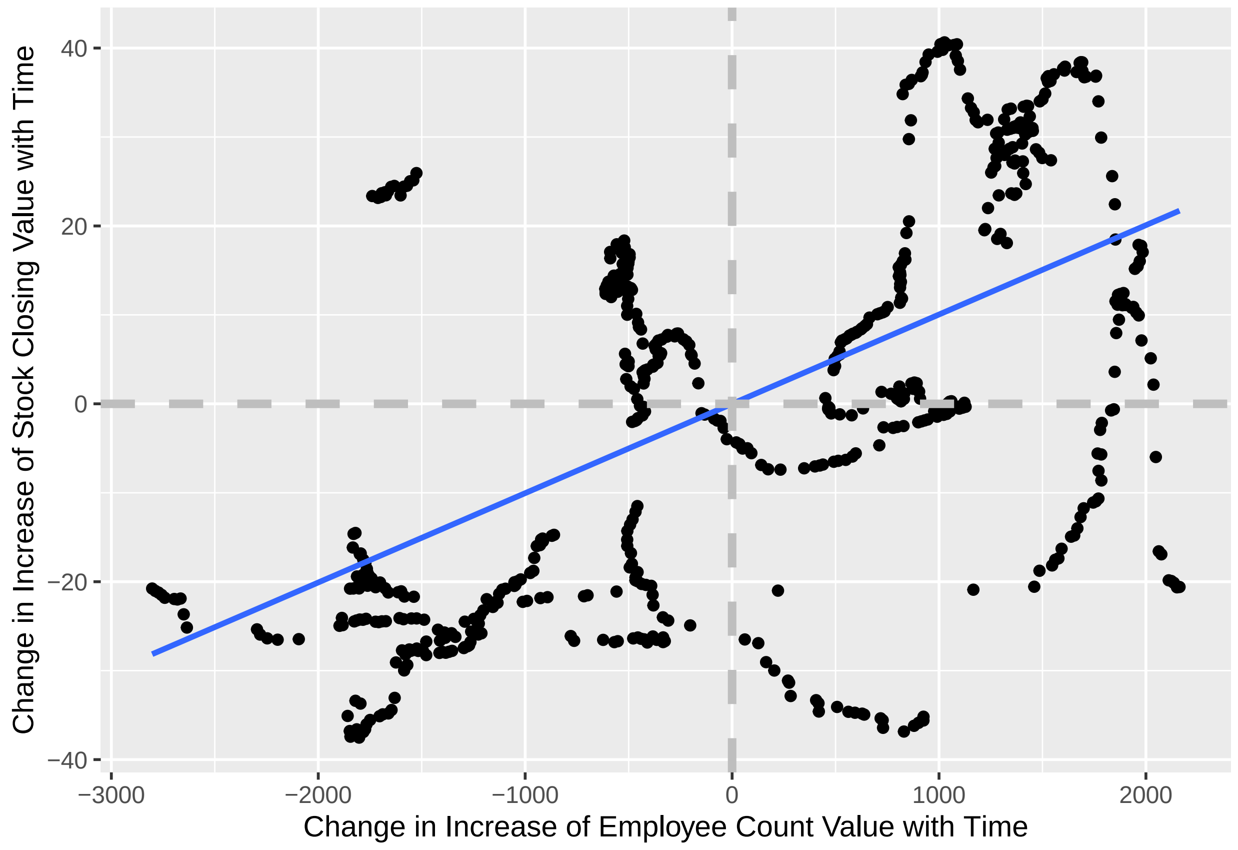
```
fit_close <- dt[, lm(stock_close ~ date_added)]
fit_emp <- dt[, lm(employees_on_platform ~ date_added)]

resid <- data.frame(close_resid = fit_close$residuals,
                                        emp_resid = fit_emp$residuals)
plot(scale(resid$emp_resid), col="red", ylab = "Scaled Residuals")
points(scale(resid$close_resid))
```



There appear to be a relationship! - even though there is some time lag is present in it.

Our aim is to see if increase (decrease) in the hiring practice would correspond to increase (decrease) in the stock prices.

```r
ggplot(resid, aes(x = emp_resid, y = close_resid)) + geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  geom_hline(yintercept=0, linetype="dashed", color = "grey", size = 1.5) +
  geom_vline(xintercept=0, linetype="dashed", color = "grey", size = 1.5)+
  ylab("Change in Increase of Stock Closing Value with Time") +
  xlab("Change in Increase of Employee Count Value with Time")
```



```
# proporation of points in +,+ or -, - quadrents
resid %>%
  summarise(correct_proportion = sum((close_resid < 0 & emp_resid < 0) |
                                     (close_resid > 0 & emp_resid > 0))/ n())
```

```
##   correct_proportion
## 1          0.6429652
```

In this case, around two thirds of the time, changes in trend of the employees numbers indicate the change in stock prices for the Goldman Sachs.

Proportion that would indicate change in the number of followers indicate the increase would correctly predict the change in increase of the closing value of the stock prices.

```
# proporation of points in +,+ or -, - quadrents
resid %>%
  summarise(prop = sum((close_resid < 0 & foll_resid < 0) | (close_resid > 0 & foll_resid > 0))/ n())
```

```
##        prop
## 1 0.4871407
```

I have considered various linear fits to the stock values. Although, the effect of linear fit appear to be statistically significant, there actually isn't meaningful relationship. –>

## Approach 1: Comparing Trend Remomved Data.

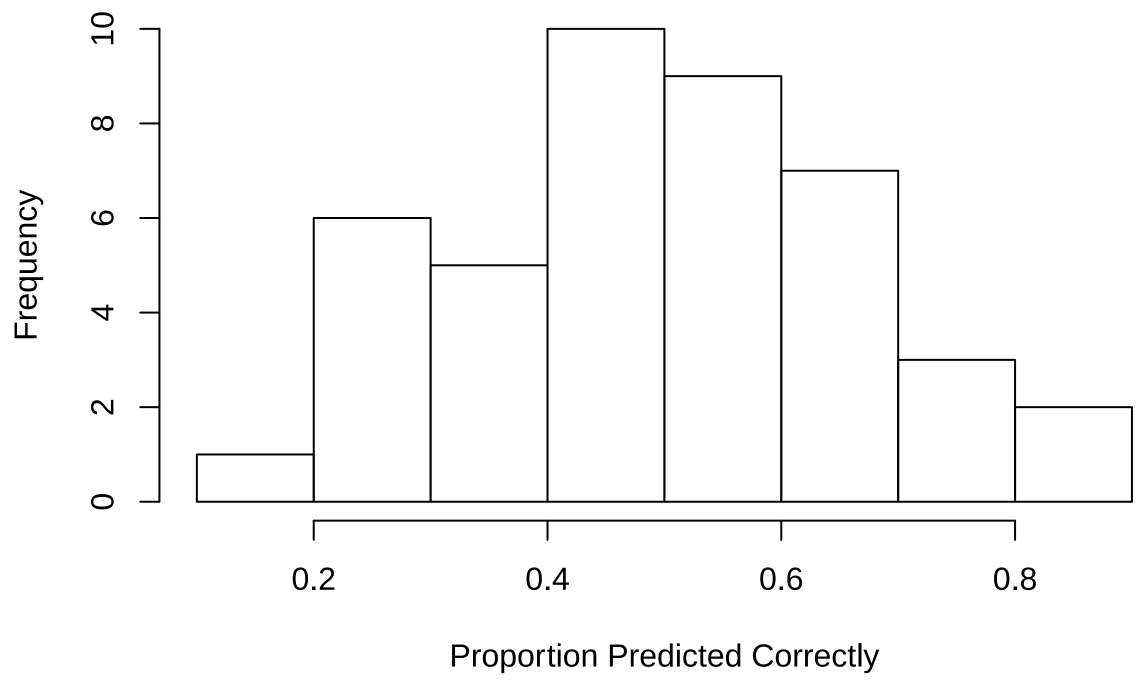
Here, we simply compare the trend removed stock closing values ($y$) with the trend removed number of employees ($x$) in LinkedIn.

We postulate that a sign of $x$ (+ or -) would corrosponds to sign of $y$. Now, we test this approach on the data we have downloaded. This contain data from 43 companies.

```r
prop_predict <- NULL
for (j in 1 : length(company)) {

    dt <-   read.csv(file = paste0(company[j], ".csv"), stringsAsFactors = FALSE)
    setDT(dt)
    #dt <- collate_data(dlink, company[j], sym[j])

    dt$date_added <- as.Date(dt$date_added)
    dt <- na.omit(dt)

  # appply the median filter
    vars <- colnames(dt)[!colnames(dt) %in% c("company_name", "date_added")]
    dt[, (vars) := lapply(.SD, function (y) as.vector(median_filter(y))),
                            .SDcols = vars]

  # remove uninteresting trend
  fit_close <- dt[, lm(stock_close ~ date_added)]
    fit_emp <- dt[, lm(employees_on_platform ~ date_added +
                                              ind_employees_on_platform)]
    fit_foll <- dt[, lm(followers_count ~ date_added + ind_followers_count)]

    resid <- data.frame(close_resid = fit_close$residuals,
                            emp_resid = fit_emp$residuals, fol_resid = fit_foll$residuals)

    fit_1 <- lm(close_resid ~ emp_resid + fol_resid, resid)

    fit_close_d <- dt[, lm(stock_close ~ date_added)]
    fit_emp_d <- dt[, lm(employees_on_platform ~ date_added)]
    fit_foll_d <- dt[, lm(followers_count ~ date_added )]

    resid2 <- data.frame(close_resid = fit_close_d$residuals,
                            emp_resid = fit_emp_d$residuals, fol_resid = fit_foll_d$residuals)

    fit_2 <- lm(close_resid ~ emp_resid + fol_resid, resid2)

    if (summary(fit_2)$adj.r.squared > summary(fit_1)$adj.r.squared)
        resid <- resid2

    prop_predict[j] <- resid %>%
    summarise(prop = sum((close_resid < 0 & emp_resid < 0) |
        (close_resid > 0 & emp_resid > 0))/ n())
}
prop_predict <- unlist(prop_predict)
```

Histogram showing the proportion that the change in hiring trend correctly predicts similar change in trend for the stock prices.

```r
hist(prop_predict, xlab = "Proportion Predicted Correctly",
     main = NULL)
```



In most cases, we bcan predict change in the stock values over 40% of times y looking at changes in hiring practices. However, this isn't that great.

---

## Approach 2: Including Time Lagged Informtion

Now we try to improve the preidction by including time lagged information from the de-trended employee and followers numbers, as well as the stock values.

We use logistic regression to predict if there is a positive change in the stock prices based on the past data including those from the LinkedIn data as well as past behaviour of the stock prices.

We can express this by

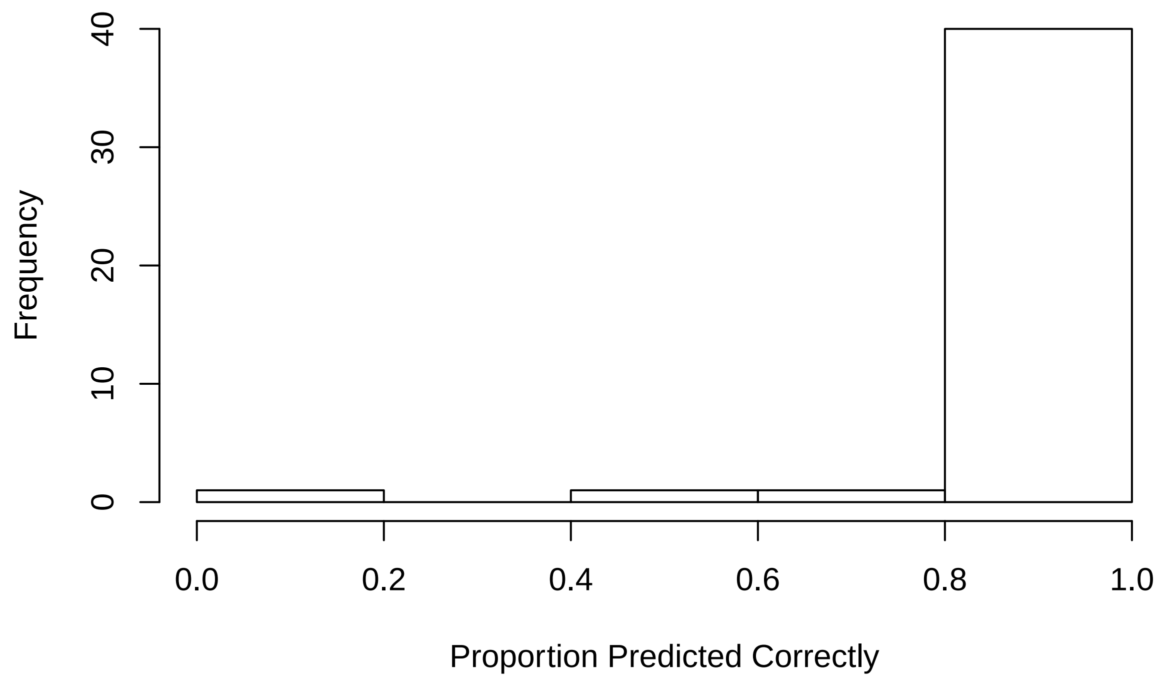
$$z_t \sim y_{t-1} + x_t + x_{t-1} + u_t + u_{t-1}$$

where $y_t$ is the change in the trend of the stock value at time $t$, $z_t = y_t > 0$ is a categorical varible expressing if the stock values has increased or not, $x$ and $u$ are de-trended the number of employees, and the number of followers in the LinkedIn.

For each company, we use the first (in time) 75% of the data to fit the model, and use the remainder to test the model.

```r
prop_predict <- NULL
for (j in 1:length(company)) {

    dt <-   read.csv(file = paste0(company[j], ".csv"), stringsAsFactors = FALSE)
    setDT(dt)
    #dt <- collate_data(dlink, company[j], sym[j])
    dt$date_added <- as.Date(dt$date_added)
    dt <- na.omit(dt)

  # appply the median filter
    vars <- colnames(dt)[!colnames(dt) %in% c("company_name", "date_added")]
    dt[, (vars) := lapply(.SD, function (y) as.vector(median_filter(y))),
                          .SDcols = vars]

   avg_dt <- dt
  # remove uninteresting trend
  fit_close <- avg_dt[, lm(stock_close ~ date_added)]
    fit_emp <- avg_dt[, lm(employees_on_platform ~ date_added +
                                          ind_employees_on_platform)]
    fit_foll <- avg_dt[, lm(followers_count ~ date_added + ind_followers_count)]

    resid <- data.frame(close_resid = fit_close$residuals,
                          emp_resid = fit_emp$residuals, fol_resid = fit_foll$residuals)
  setDT(resid)
  n_lag <- 1
  resid[, paste0("close_resid", 1:n_lag) := shift(close_resid, n=1:n_lag, type = "lag")]
  resid[, paste0("emp_resid", 1:n_lag) := shift(emp_resid, n=1:n_lag, type = "lag")]
  resid[, paste0("fol_resid", 1:n_lag) := shift(fol_resid, n=1:n_lag, type = "lag")]

  resid[, close_resid := as.factor(close_resid > 0)]
  n <- nrow(resid)
  n_train <- round(n * 0.75)

  mylogit <- glm(close_resid ~ ., data = resid[1:n_train], family = "binomial")

  out  <- predict(mylogit, newdata=resid[-c(1:n_train)], type = "response") > 0.5

    prop_predict[j] <- mean(resid[-c(1:n_train)]$close_resid == out)
}
prop_predict <- unlist(prop_predict)
```

```
hist(prop_predict, xlab = "Proportion Predicted Correctly",
     main = NULL)
```



## Conclustion

Based on the outcomes of this study, we note that the changes in stock prices can be predicted to a high accuracy using time lagged information of the stock prices as well as the LinkedIn data.