SAEID REZAEI

CAPSTONE 2017

STOCK MARKET PREDICTION USING TIME SERIES METHOD

DATE

11-DEC-2017

INSTRUCTOR

CENI BABAOGLU

Contents

Predict Stock Market (PSM)

1. Introduction

A Stock market, equity market or share market is the aggregation of buyers and sellers of stocks, which represents ownership claims of business, Stock market and share price changes based on economy, international reputation, war, and so on. Therefore, Investors are willing to know the future and predict stock market

In this project, I'm going to analyze data related to 30 famous stock market and predict their price and up and down based on price history using analytical model called time series. In addition I will visualize my data in different ways to help portfolio managers convince investor in the better way.

2. Literature Review

Stock Market prediction has always had a certain appeal for researchers. While numerous scientific attempts have been made, no method has been discovered to accurately predict stock price movement. The difficulty of prediction lies in the complexities of modeling market dynamics. Even with a lack of consistent prediction methods, there have been some mild successes. Stock Market research encapsulates two elemental trading philosophies; Fundamental and Technical approaches. In Fundamental analysis, Stock Market price movements are believed to derive from a security's relative data. Fundamentalists use numeric information such as earnings, ratios, and management effectiveness to determine future forecasts. In Technical analysis, it is believed that market timing is key. Technicians utilize charts and modeling techniques to identify trends in price and volume. These later individuals rely on historical data in order to predict future outcomes. Most existing literature on financial text mining relies on identifying a predefined set of keywords and machine learning techniques. These methods typically assign weights to keywords in proportion to the movement of a share price. These types of analyses have shown a definite, but weak ability to forecast the direction of share prices.

3. Dataset

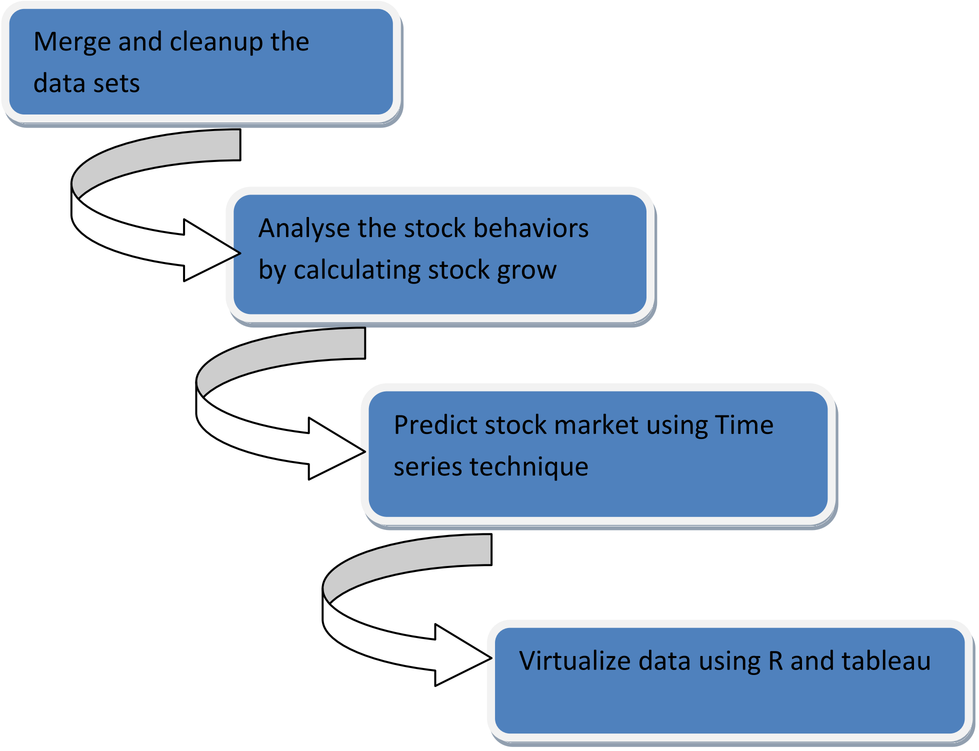
In this project I'm going to use stock price history from Yahoo finance and S&P 500. I would use around 500 stocks which have at least 30 stock traded for at least 10 years, so I'm sure that 10 years of price history is available. My data set contains Date, symbol, open price, low price, high price, close price, and volume and contains around 1,200,000 rows in CSV format

https://quantquote.com/historical-stock-data

http://www.nasdaq.com/symbol

- Date: Represents the date of the market price, format is DATE, YYYYMMDD.
- Symbol: each company or stock in the market is known with symbol. This is a unique value that identifies equity in the market, its CHAR variable that could be a combination of alphabet and number.
- Open price: this is a numeric entity that shows price of each unit of stock when market opens, this price is usually is closed to previous day closing price.
- Low price: this item shows lowest price of each unit in a day, this items changes frequency while market is open, however my data set record the lowest one. This is a numeric attribute with 3 decimal.
- High price: this item is opposite of low price, hence shows highest unit price in one day. This is also numeric attribute with 3 decimal.
- Close price: this item records the latest share price during a day and do not use in financial calculation such as rate of returns (ROR). Same as above, this is a numeric with 3 decimal.
- Volume: This attribute shows the number of shares in total that has been traded within a day; Volume is an important element for investor to analyze the stock market.

4. Approach


Merge and cleanup the data sets
Analyse the stock behaviors by calculating stock grow
Predict stock market using Time series technique
Virtualize data using R and tableau

Step 1: <Merge and cleanup data set>

I've downloaded around 500 individual data file (in CSV format) for different stock as of 10 years ago, I wrote a script in Perl to read each file and write into one consolidated file. In addition, I used the previous price for missing price value, if price is missed on 1st day I evaluated as zero. In this project I've also got symbol price on line with using "quantmod" package in R. I realized it's much more effective for my software to read data online, similarly source of data could be either GOOGLE or YAHOO. There were certain dates that price/data either was missing, In this case my Perl program as well as R function could handle the missing price by putting closed price for missing price and Zero for initial price (1st day)

Step 2: <Analyze the stock behavior by calculating stock growth >

I have followed several methods to analyze stock market and Index. I've reviewed the trend of each index such as IBEX, SP&500 individually for the period of 10 years. I've also done basket analyzes by putting several stock in same category or industry and compared their behavior and price changes

Step 3: <Predict stock market using time series>

Since stock market price and Index factors are changing almost every day, I used the time series method in most of my analyzes to compare the price, forecasting, and prediction.

Step 4: <Virtualize data using R and Tableau >

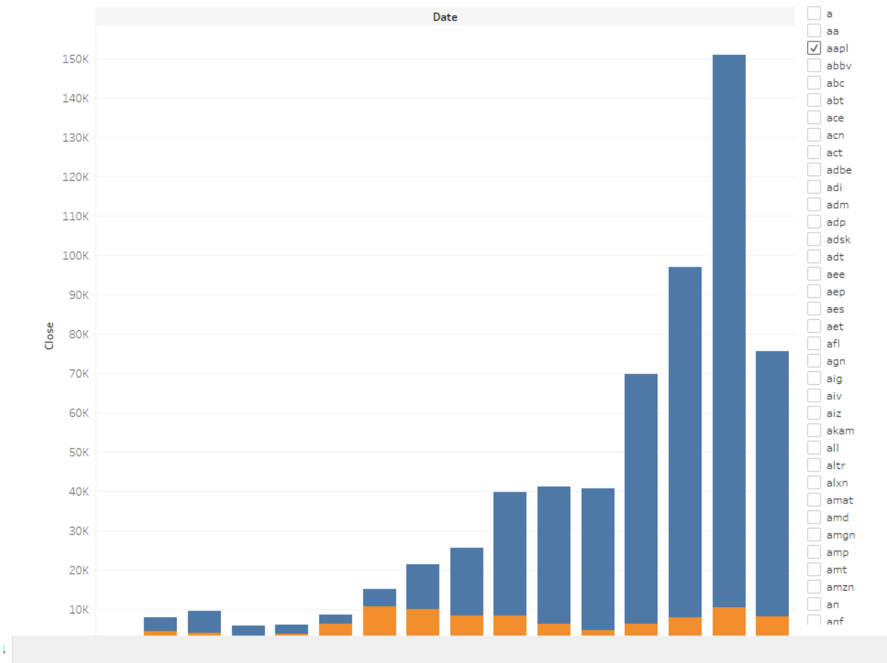
Since R is very powerful software for chart and graphs, I used mainly R to build my chart however I used Rserver to connect R to Tableau.

Step 5: <Conclusion and result >

Result would be clean and processed data in target files, several models and charts in R that shows stock market evaluation and prediction. Visual chart in Tableau that shows:

Sample:

Analyse Stock Market and Index



5. Results / Code

Stock Market Analyses - Terend

Saeid Rezaei

January 7, 2018

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
#############################################################################
##########################################
# Script name: SMP.SaeidRezaei.R
# Porpouse   : This script is developed to analyse the stock market for
certain security and
#              provide prediction based on stock price (using time series
method)
# Data source: Data source could be off-line (marketPriceHistory.csv) or
online SP&500
# R Package usagae:
# quantmod
# ggplot2
# forecast
# plotly
# ggfortify
# tseries
# gridExtra
# docstring
# here
#############################################################################
##########################################
#Developer         Date                 Version                    Reason
#Saeid Rezaei     2017-12-20             0                       Initial Version
#############################################################################
##########################################

# Start program
print ("Start program - Forcaste Stock Marekt")

## [1] "Start program - Forcaste Stock Marekt"
```

Analyse Stock Market and Index

Description: In this project, my input data is came from either Yahoo or Google finance. At the first attempt I found the index and several price histories from Yahoo finance and S&P 500 website, all files were individual, therefore I developed a script with PERL to read files, transform data (missing data based on closer record) and merge them all in one file. Then I realized R has a function called “Quantum” that could give me this opportunity to get the index and stock price on-line from Google or Yahoo. Missing price is also handling by R.

```
print ("STEP 1: Merging data into one file and value missing records")

## [1] "STEP 1: Merging data into one file and value missing records"

# If you are using off line market price you would need to execute
# DataClening.pl (Perl) script to merge files and value the secirities
# with missing price, The method is to value the missing price by looking
into
# Previous price, if this is first row price would be Zero (0)
# Note: I'm running from my local drive. You would need to specify the path
# if you are running from other location
# Recomandation setup:
# Create subfolder in your local (C) drive call it CHM136
# Create another sub-directory under CHM136 call id StockPriceHist
# Copy all downloaded price .csv files there

system("perl C:/CHM136/DataCleaning.pl")

print ("STEP 2: Analyse data and train data")

## [1] "STEP 2: Analyse data and train data"

print ("STEP 2.1: Install and Load R Packages")

## [1] "STEP 2.1: Install and Load R Packages"

#install.packages('quantmod')
#install.packages('binhf')
library(quantmod)

## Warning: package 'quantmod' was built under R version 3.3.3

## Loading required package: xts

## Warning: package 'xts' was built under R version 3.3.3

## Loading required package: zoo

## Warning: package 'zoo' was built under R version 3.3.3
```

```
##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: TTR

## Warning: package 'TTR' was built under R version 3.3.3

## Version 0.4-0 included new data defaults. See ?getSymbols.
```

```
# Load data into Var.
# Load data from local .csv file into var.
#marketPriceHisotry <- read.csv(
"C:/CHM136/StockPriceHist/output/secPriceHistory.csv")
#attach(marketPriceHisotry)

# Since Downloading data is not up-t-date, I used R PACKAGE CALLED quantmod
to get realtime stock price
# I'll use that source in my project going forward

print ("STEP 2.2: Get stock price from Yahoo and analyse data")
```

```
## [1] "STEP 2.2: Get stock price from Yahoo and analyse data"
```

```
getSymbols('SPY', src='yahoo')
```

```
## 'getSymbols' currently uses auto.assign=TRUE by default, but will
## use auto.assign=FALSE in 0.5-0. You will still be able to use
## 'loadSymbols' to automatically load data. getOption("getSymbols.env")
## and getOption("getSymbols.auto.assign") will still be checked for
## alternate defaults.
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.warning4.0"=FALSE). See ?getSymbols for details.

##
## WARNING: There have been significant changes to Yahoo Finance data.
## Please see the Warning section of '?getSymbols.yahoo' for details.
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.yahoo.warning"=FALSE).

## [1] "SPY"
```

```
getSymbols('^GSPC', src='yahoo')
```

```
## [1] "GSPC"
```

```
getSymbols('^IBEX', src='yahoo')
```

```
## Warning: ^IBEX contains missing values. Some functions will not work if
## objects contain missing values in the middle of the series. Consider using
## na.omit(), na.approx(), na.fill(), etc to remove or replace them.
```

```
## [1] "IBEX"
```

```
getSymbols(c('QQQ'), src='google')
```

```
## [1] "QQQ"
```

```
head(GSPC)
```

```
##            GSPC.Open GSPC.High GSPC.Low GSPC.Close GSPC.Volume
## 2007-01-03   1418.03   1429.42  1407.86    1416.60  3429160000
## 2007-01-04   1416.60   1421.84  1408.43    1418.34  3004460000
## 2007-01-05   1418.34   1418.34  1405.75    1409.71  2919400000
## 2007-01-08   1409.26   1414.98  1403.97    1412.84  2763340000
## 2007-01-09   1412.84   1415.61  1405.42    1412.11  3038380000
## 2007-01-10   1408.70   1415.99  1405.32    1414.85  2764660000
##            GSPC.Adjusted
## 2007-01-03       1416.60
## 2007-01-04       1418.34
## 2007-01-05       1409.71
## 2007-01-08       1412.84
## 2007-01-09       1412.11
## 2007-01-10       1414.85
```

```
tail(GSPC)
```

```
##            GSPC.Open GSPC.High GSPC.Low GSPC.Close GSPC.Volume
## 2017-12-28   2686.10   2687.66  2682.69    2687.54  2153330000
## 2017-12-29   2689.15   2692.12  2673.61    2673.61  2443490000
## 2018-01-02   2683.73   2695.89  2682.36    2695.81  3357250000
## 2018-01-03   2697.85   2714.37  2697.77    2713.06  3538660000
## 2018-01-04   2719.31   2729.29  2719.07    2723.99  3695260000
## 2018-01-05   2731.33   2743.45  2727.92    2743.15  3236620000
##            GSPC.Adjusted
## 2017-12-28       2687.54
## 2017-12-29       2673.61
## 2018-01-02       2695.81
## 2018-01-03       2713.06
## 2018-01-04       2723.99
## 2018-01-05       2743.15
```

```
head(SPY)
```

```
##            SPY.Open SPY.High SPY.Low SPY.Close SPY.Volume SPY.Adjusted
## 2007-01-03   142.25   142.86  140.57    141.37   94807600     113.1958
## 2007-01-04   141.23   142.05  140.61    141.67   69620600     113.4360
## 2007-01-05   141.33   141.40  140.38    140.54   76645300     112.5312
## 2007-01-08   140.82   141.41  140.25    141.19   71655000     113.0516
## 2007-01-09   141.31   141.60  140.40    141.07   75680100     112.9556
## 2007-01-10   140.58   141.57  140.30    141.54   72428000     113.3318
```

```
tail(SPY)
```

```
##            SPY.Open SPY.High SPY.Low SPY.Close SPY.Volume SPY.Adjusted
## 2017-12-28   267.89   267.92  267.45    267.87   45116100       267.87
## 2017-12-29   268.53   268.55  266.64    266.86   96007400       266.86
## 2018-01-02   267.84   268.81  267.40    268.77   86655700       268.77
## 2018-01-03   268.96   270.64  268.96    270.47   90070400       270.47
## 2018-01-04   271.20   272.16  270.54    271.61   80636400       271.61
## 2018-01-05   272.51   273.56  271.95    273.42   72820100       273.42
```

Description: On above steps, I loaded price history for certain symbol & index into R variable and tried to look into data structure.

Below I tried to catch the null value and price them either with 0 or close price.

```
# Remove the null values
QQQ <- QQQ[!(rowSums(is.na(QQQ))),]
SPY <- SPY[!(rowSums(is.na(SPY))),]

GSPC <- GSPC[!(rowSums(is.na(GSPC))),]
IBEX <- IBEX[!(rowSums(is.na(IBEX))),]



# GSPC and SPY are Time series data, Let's find the class
class(GSPC)
```

```
## [1] "xts" "zoo"
```

```
# Create a vector and put more than one symbol into that
# This VAR will being used to compare more than one symbol
# and analyse the market
basketSymbols <-(c('YELP','AAPL','AMZN'))
getSymbols(basketSymbols, src='yahoo')
```

```
## [1] "YELP" "AAPL" "AMZN"
```

```
# Analyse the Data
summary(YELP)
```

```
##      Index              YELP.Open       YELP.High        YELP.Low
##  Min.   :2012-03-02   Min.   :15.11   Min.   : 15.26   Min.   :14.10
##  1st Qu.:2013-08-19   1st Qu.:25.01   1st Qu.: 25.68   1st Qu.:24.51
##  Median :2015-02-04   Median :35.61   Median : 36.07   Median :35.02
##  Mean   :2015-02-03   Mean   :40.47   Mean   : 41.34   Mean   :39.58
##  3rd Qu.:2016-07-21   3rd Qu.:51.33   3rd Qu.: 52.61   3rd Qu.:50.42  
##  Max.   :2018-01-05   Max.   :99.80   Max.   :101.75   Max.   :97.25  
##    YELP.Close     YELP.Volume       YELP.Adjusted  
##  Min.   :15.22   Min.   :  226800   Min.   :15.22  
##  1st Qu.:25.09   1st Qu.: 1273550   1st Qu.:25.09  
##  Median :35.43   Median : 2002650   Median :35.43  
##  Mean   :40.45   Mean   : 2683634   Mean   :40.45  
##  3rd Qu.:51.45   3rd Qu.: 3044850   3rd Qu.:51.45  
##  Max.   :98.04   Max.   :47155000   Max.   :98.04  
```

```
summary(AAPL)
```

```
##      Index              AAPL.Open        AAPL.High         AAPL.Low     
##  Min.   :2007-01-03   Min.   : 11.34   Min.   : 11.71   Min.   : 11.17  
##  1st Qu.:2009-10-02   1st Qu.: 27.25   1st Qu.: 27.55   1st Qu.: 26.96  
##  Median :2012-07-03   Median : 65.08   Median : 65.41   Median : 64.21  
##  Mean   :2012-07-04   Mean   : 69.75   Mean   : 70.38   Mean   : 69.07  
##  3rd Qu.:2015-04-08   3rd Qu.:103.10   3rd Qu.:105.05   3rd Qu.:102.72  
##  Max.   :2018-01-05   Max.   :175.11   Max.   :177.20   Max.   :174.86  
##    AAPL.Close      AAPL.Volume        AAPL.Adjusted   
##  Min.   : 11.17   Min.   : 11475900   Min.   : 10.01  
##  1st Qu.: 27.21   1st Qu.: 49739400   1st Qu.: 24.39  
##  Median : 64.76   Median : 97645800   Median : 58.87  
##  Mean   : 69.75   Mean   :123330073   Mean   : 65.74  
##  3rd Qu.:103.30   3rd Qu.:164516100   3rd Qu.: 99.24  
##  Max.   :176.42   Max.   :843242400   Max.   :176.42  
```

```
summary(AMZN)
```

```
##      Index              AMZN.Open         AMZN.High      
##  Min.   :2007-01-03   Min.   :  35.29   Min.   :  37.07  
##  1st Qu.:2009-10-02   1st Qu.:  95.00   1st Qu.:  95.90  
##  Median :2012-07-03   Median : 227.57   Median : 230.97  
##  Mean   :2012-07-04   Mean   : 324.42   Mean   : 327.69  
##  3rd Qu.:2015-04-08   3rd Qu.: 398.94   3rd Qu.: 403.49  
##  Max.   :2018-01-05   Max.   :1217.51   Max.   :1229.14  
##     AMZN.Low         AMZN.Close       AMZN.Volume        AMZN.Adjusted  
##  Min.   :  34.68   Min.   :  35.03   Min.   :   984400   Min.   :  35.03  
##  1st Qu.:  93.11   1st Qu.:  94.45   1st Qu.:  3034000   1st Qu.:  94.45  
##  Median : 225.29   Median : 228.29   Median :  4550700   Median : 228.29  
##  Mean   : 320.87   Mean   : 324.48   Mean   :  5815254   Mean   : 324.48  
##  3rd Qu.: 394.29   3rd Qu.: 398.79   3rd Qu.:  7077400   3rd Qu.: 398.79  
##  Max.   :1210.00   Max.   :1229.14   Max.   :104329200   Max.   :1229.14  
```

```
# Merge all there symbol data into one data frame
basket <- data.frame(as.xts(merge(YELP,AAPL,AMZN)))
# N/A respresents when Symbol does not have have price
head(basket)
```

```
##            YELP.Open YELP.High YELP.Low YELP.Close YELP.Volume
## 2007-01-03        NA        NA       NA         NA          NA
## 2007-01-04        NA        NA       NA         NA          NA
## 2007-01-05        NA        NA       NA         NA          NA
## 2007-01-08        NA        NA       NA         NA          NA
## 2007-01-09        NA        NA       NA         NA          NA
## 2007-01-10        NA        NA       NA         NA          NA
##            YELP.Adjusted AAPL.Open AAPL.High AAPL.Low AAPL.Close
## 2007-01-03            NA  12.32714  12.36857 11.70000   11.97143
## 2007-01-04            NA  12.00714  12.27857 11.97429   12.23714
## 2007-01-05            NA  12.25286  12.31428 12.05714   12.15000
## 2007-01-08            NA  12.28000  12.36143 12.18286   12.21000
## 2007-01-09            NA  12.35000  13.28286 12.16429   13.22429
## 2007-01-10            NA  13.53571  13.97143 13.35000   13.85714
##            AAPL.Volume AAPL.Adjusted AMZN.Open AMZN.High AMZN.Low
## 2007-01-03   309579900      10.73159     38.68     39.06    38.05
## 2007-01-04   211815100      10.96978     38.59     39.14    38.26
## 2007-01-05   208685400      10.89166     38.72     38.79    37.60
## 2007-01-08   199276700      10.94545     38.22     38.31    37.17
## 2007-01-09   837324600      11.85469     37.60     38.06    37.34
## 2007-01-10   738220000      12.42201     37.49     37.70    37.07
##            AMZN.Close AMZN.Volume AMZN.Adjusted
## 2007-01-03      38.70    12405100         38.70
## 2007-01-04      38.90     6318400         38.90
## 2007-01-05      38.37     6619700         38.37
## 2007-01-08      37.50     6783000         37.50
## 2007-01-09      37.78     5703000         37.78
## 2007-01-10      37.15     6527500         37.15
```

```
tail(basket)
```

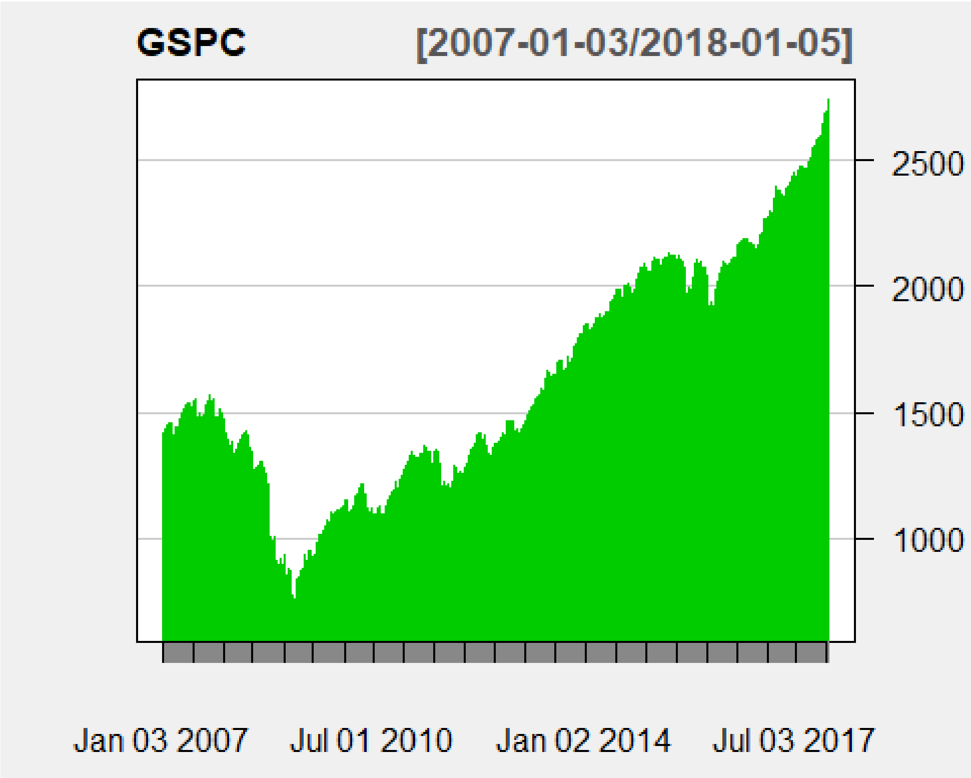
```
##            YELP.Open YELP.High YELP.Low YELP.Close YELP.Volume
## 2017-12-28     42.44     42.74    42.01      42.23      472900
## 2017-12-29     42.23     42.43    41.90      41.96      688800
## 2018-01-02     42.06     43.47    42.06      43.24     1355300
## 2018-01-03     43.35     43.59    42.86      43.12     1347700
## 2018-01-04     43.36     43.53    42.52      42.82      933000
## 2018-01-05     42.95     43.34    42.74      43.17      903600
##            YELP.Adjusted AAPL.Open AAPL.High AAPL.Low AAPL.Close
## 2017-12-28         42.23    171.00    171.85   170.48     171.08
## 2017-12-29         41.96    170.52    170.59   169.22     169.23
## 2018-01-02         43.24    170.16    172.30   169.26     172.26
## 2018-01-03         43.12    172.53    174.55   171.96     172.23
## 2018-01-04         42.82    172.54    173.47   172.08     173.03
## 2018-01-05         43.17    173.44    175.37   173.05     175.00
##            AAPL.Volume AAPL.Adjusted AMZN.Open AMZN.High AMZN.Low
## 2017-12-28    16480200        171.08   1189.00   1190.10  1184.38
## 2017-12-29    25999900        169.23   1182.35   1184.00  1167.50
## 2018-01-02    25555900        172.26   1172.00   1190.00  1170.51
## 2018-01-03    29517900        172.23   1188.30   1205.49  1188.30
## 2018-01-04    22434600        173.03   1205.00   1215.87  1204.66
## 2018-01-05    23329000        175.00   1217.51   1229.14  1210.00
##            AMZN.Close AMZN.Volume AMZN.Adjusted
## 2017-12-28    1186.10     1841700       1186.10
## 2017-12-29    1169.47     2688400       1169.47
## 2018-01-02    1189.01     2694500       1189.01
## 2018-01-03    1204.20     3108800       1204.20
## 2018-01-04    1209.59     3022100       1209.59
## 2018-01-05    1229.14     3452800       1229.14
```

```
# Draw few charts to do basid analyses
print ("STEP 2.3: Draw few charts and analyse them")
```

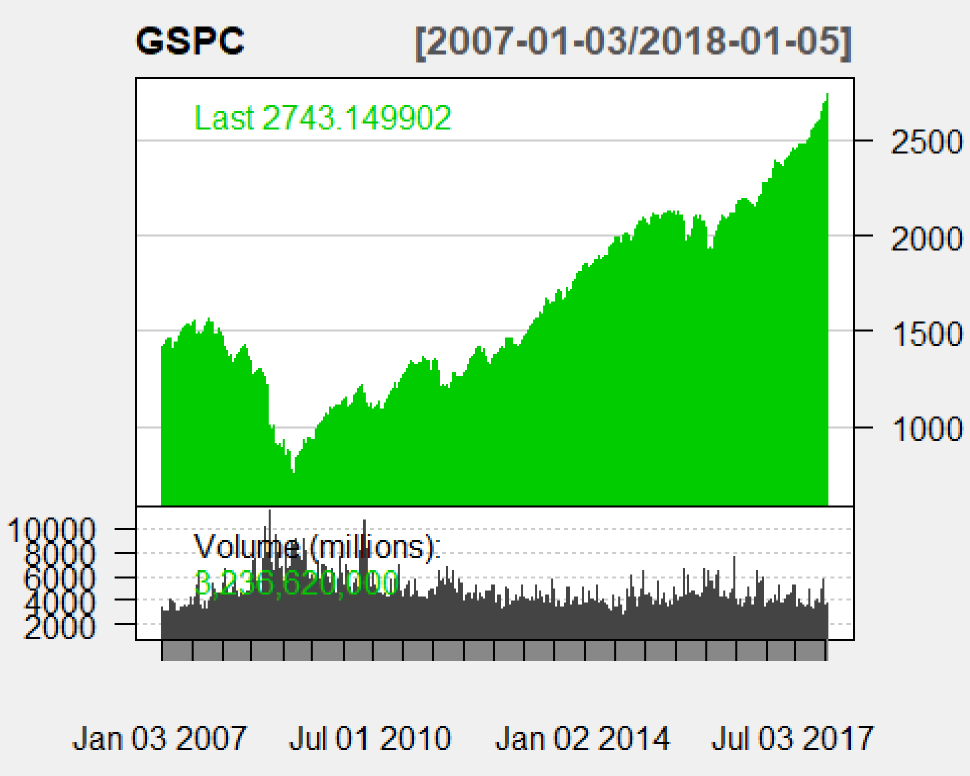
```
## [1] "STEP 2.3: Draw few charts and analyse them"
```

```
lineChart(GSPC,line.type = 'h',theme = 'white',TA=NULL)
```

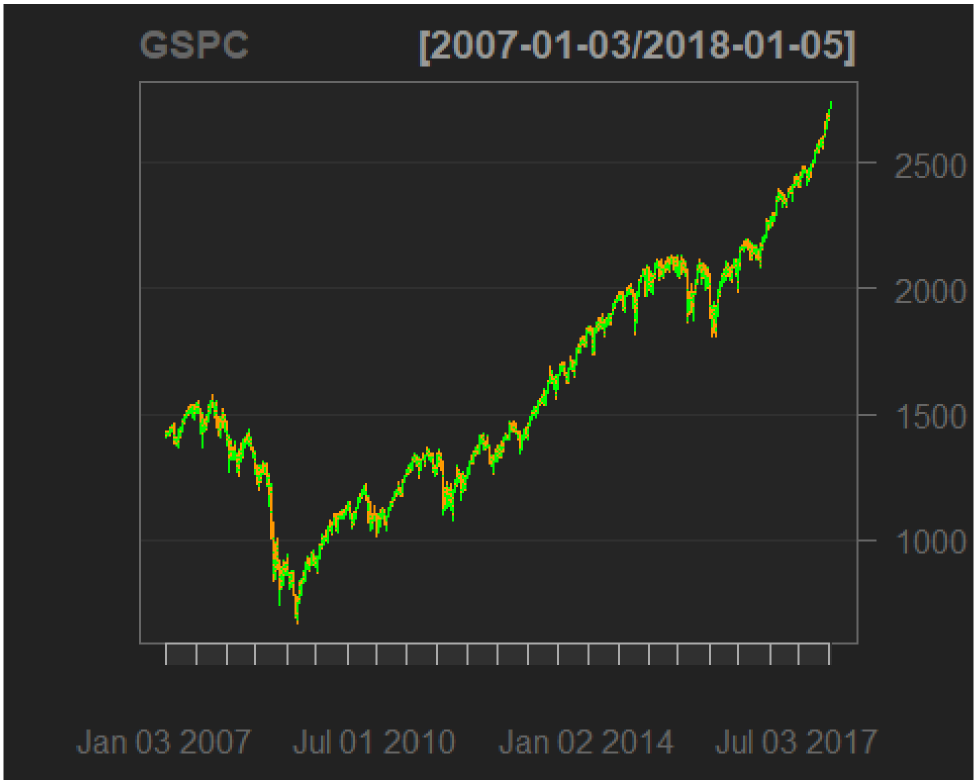
```
Description:

Below chart is showing S&P Index (GSPC) for the period of 10 years in
quarter. As you see below, the Index price was low between 2008 – 2010 Q1 due
to financial crisis starts on 2008. On the other chart I tried to find the
relation between Index price and Index Volume. I realized the volume on Index
was “high” for the same period when Index price was low due to crisis. It
seems investors tried to sell more often that other times
```



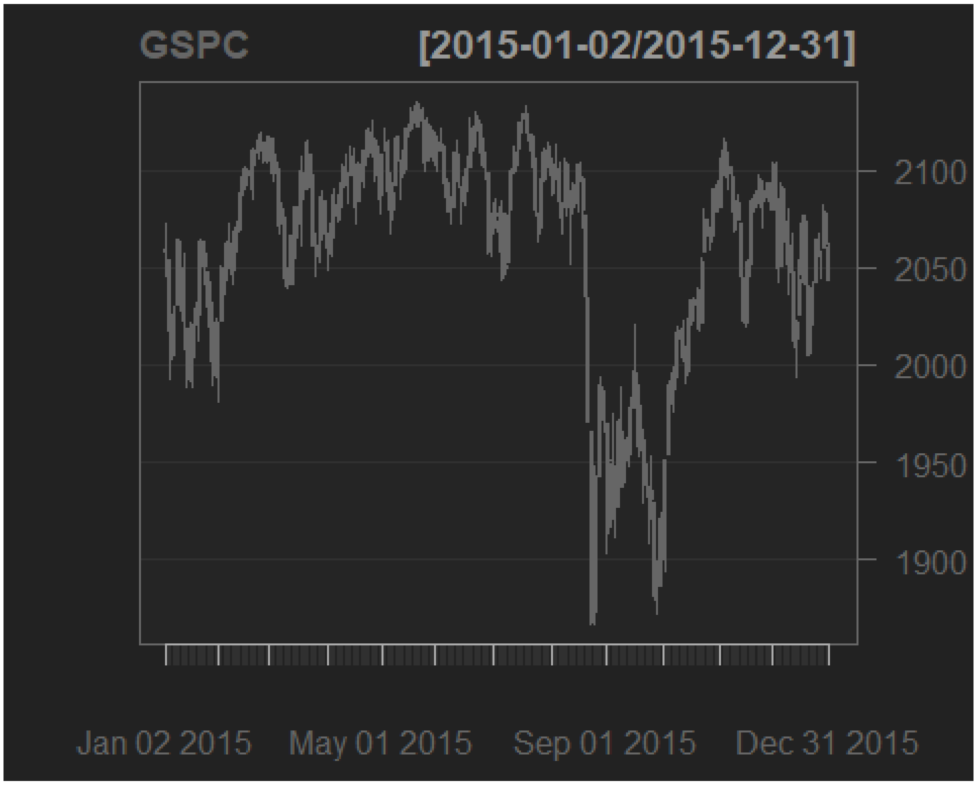
```
# put the volumn
lineChart(GSPC,line.type = 'h',theme = 'white')
```



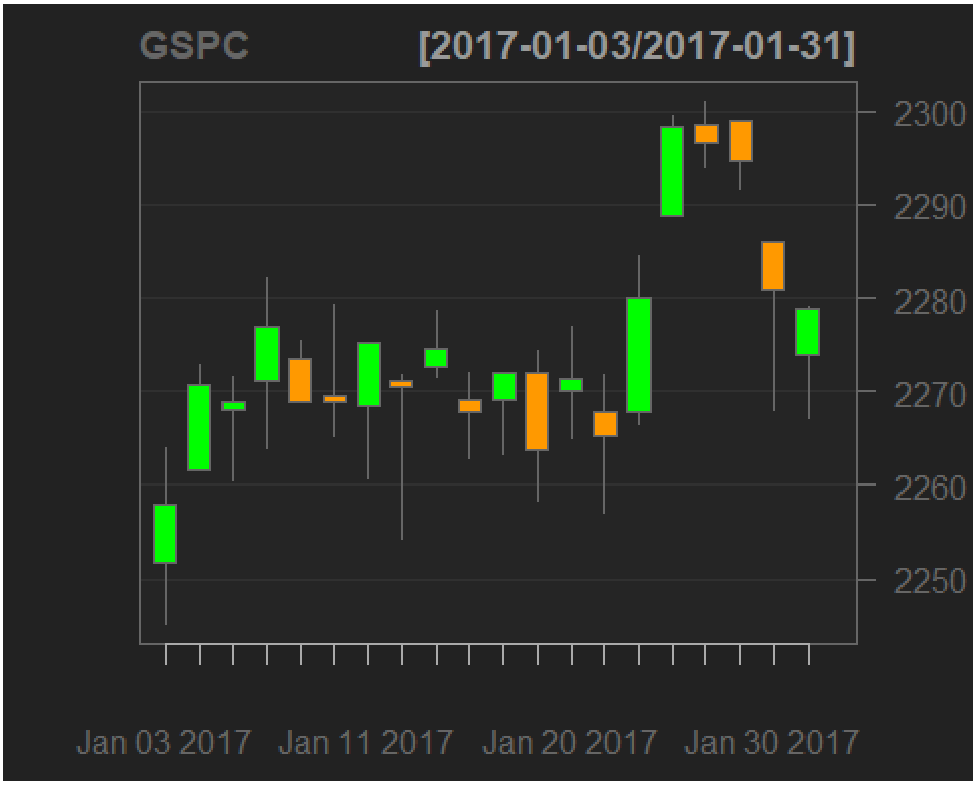
```
barChart(GSPC,bar.type = 'hcl',TA=NULL)
```



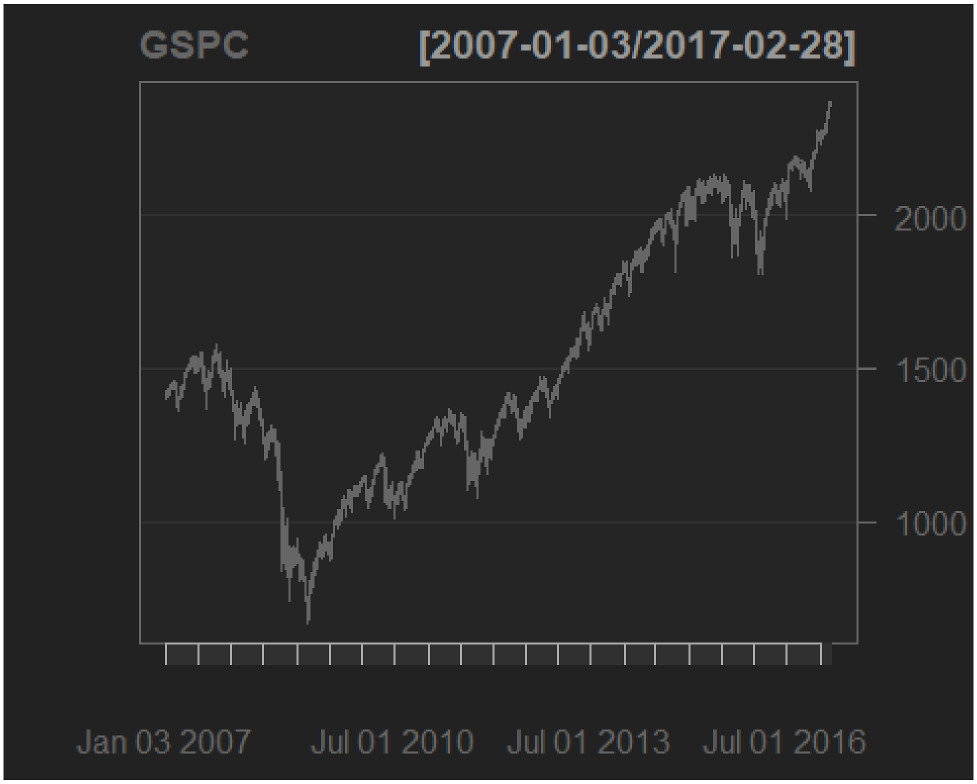
```
candleChart(GSPC,TA=NULL,subset = '2015')
```



```
# Fucase on Jan 2017
candleChart(GSPC,TA=NULL,subset = '2017-01')
```

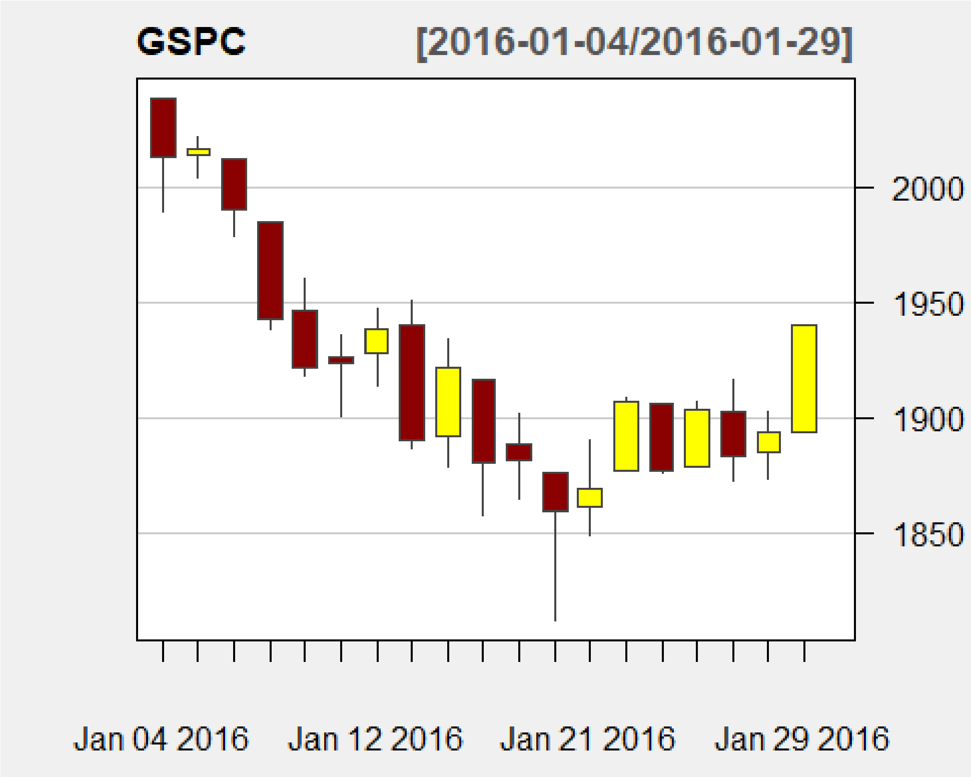


```
# Review the price changes from Feb 2017 and backward to 1st day
candleChart(GSPC,TA=NULL,subset = '::2017-02')
```

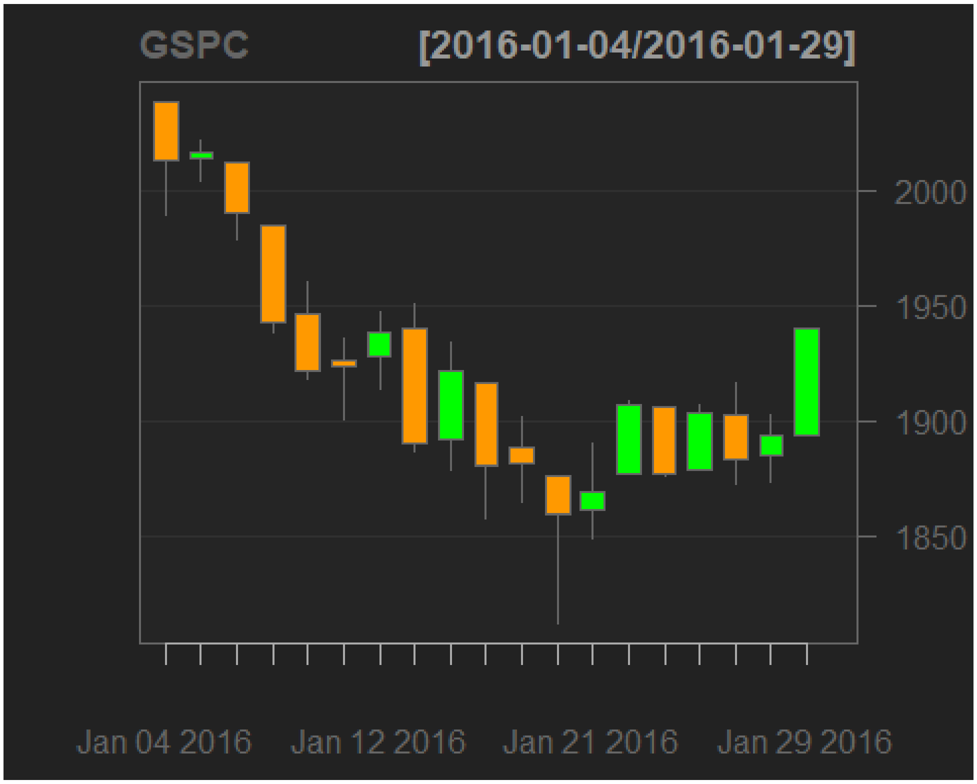


```
candleChart(GSPC,theme =
chartTheme('white',up.col='yellow',dn.col='darkred'),
            TA=NULL,subset = '2016-01')
```

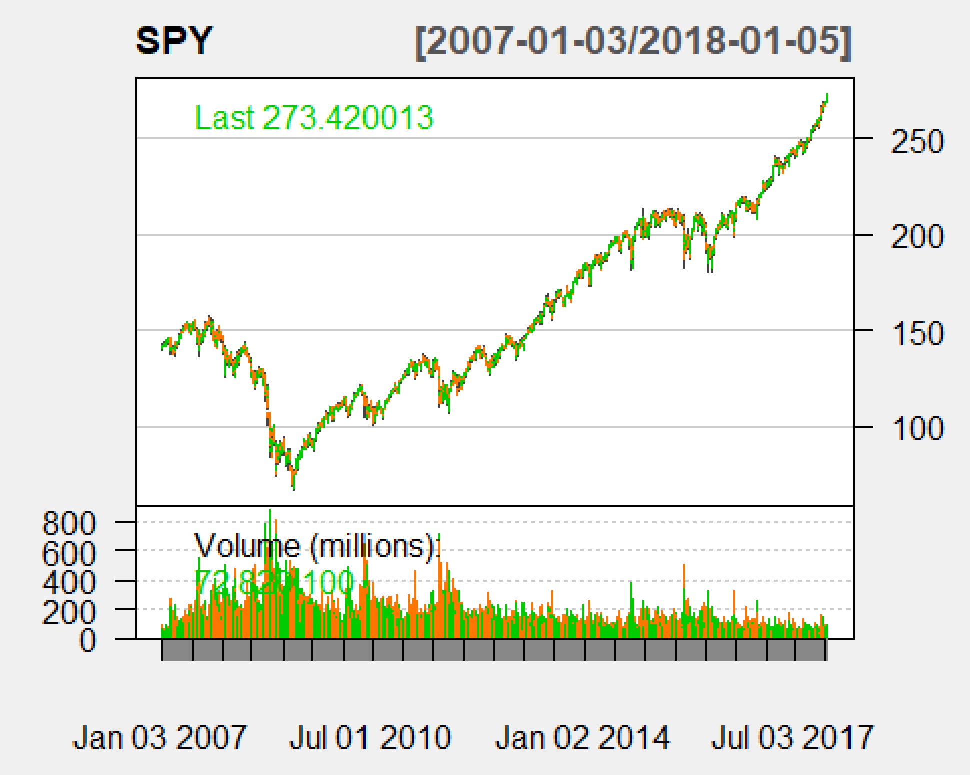
Description: Below chart is showing outliers for Index price for only one month. I tried to look closer into Index price changes within a month, red box is showing grow and yellow box is showing drop on Index price. Last week of Jan is showing two consecutive week Index price drop.



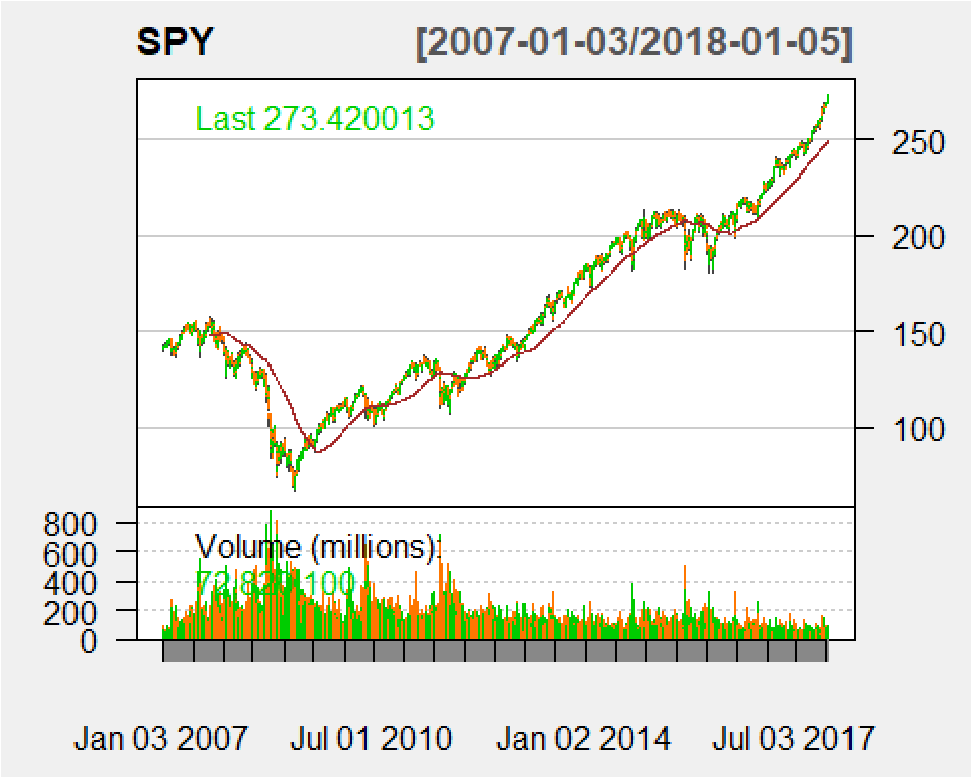
```
chartSeries(GSPC,type =c("candlesticks"),TA=NULL,subset = '2016-01')
```



```
chartSeries(SPY, theme='white')

I have done same analyses for other stock SPY.
```

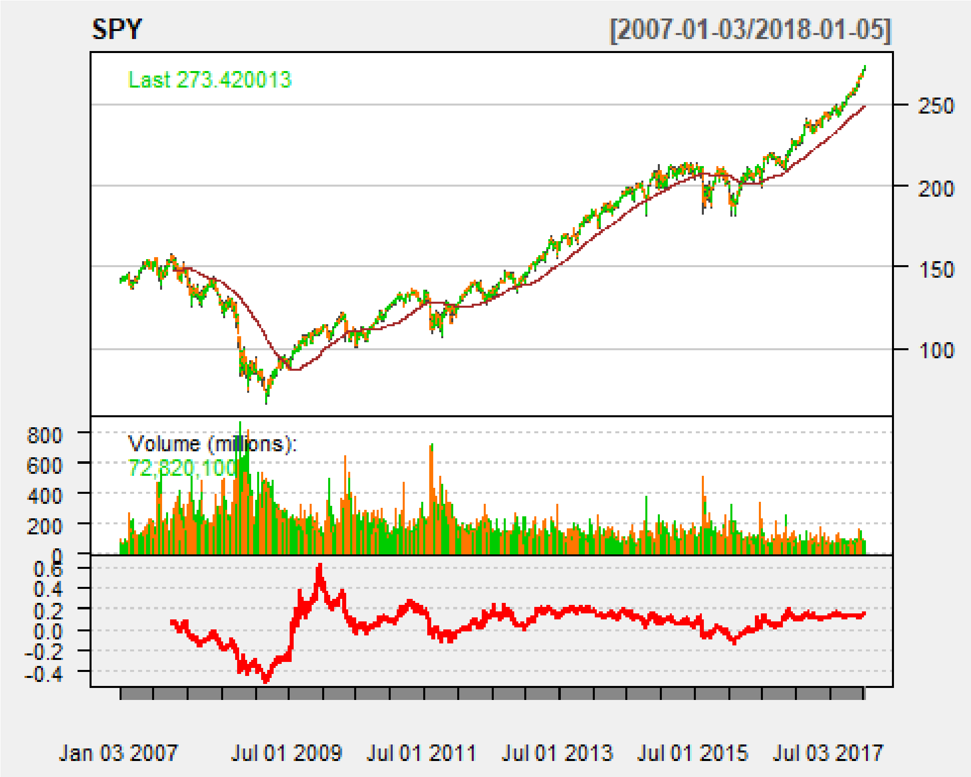


```
# Let's find the Symple moving avarage for period of 200
#{{\mathit  {momentum}} \over N+1}={\mathit  {SMA}}_{{\mathit  {today}}}-
{\mathit  {SMA}}_{{\mathit  {yesterday}}}


addSMA(n=200)
```

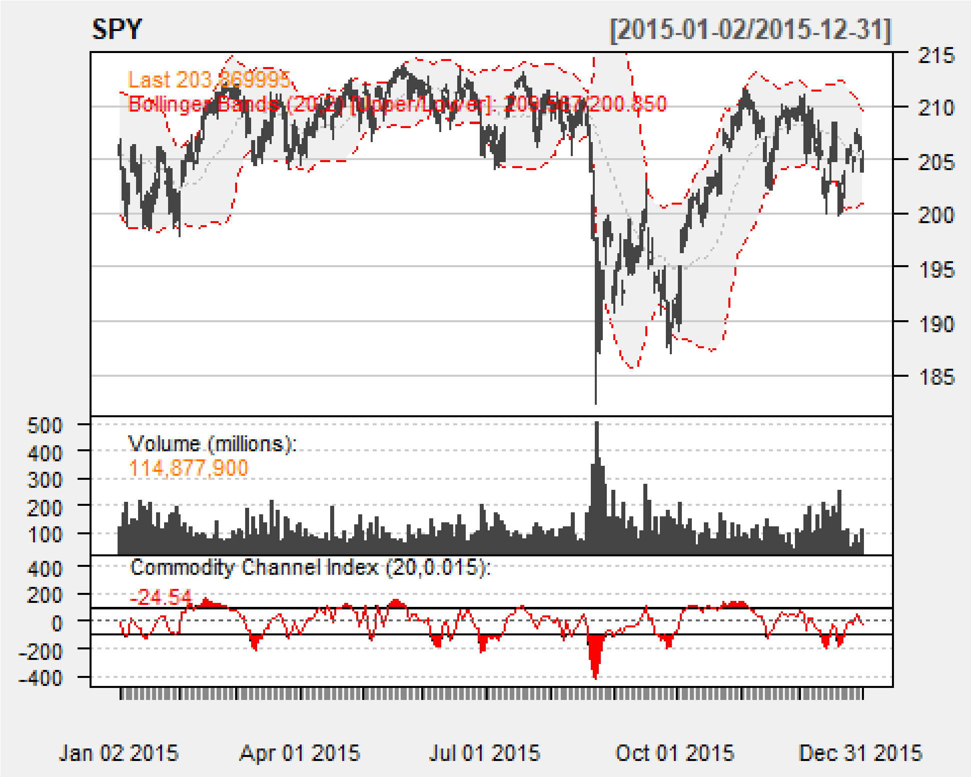


```
#Find the 10 period days of rate of change

addROC(n=200)
```

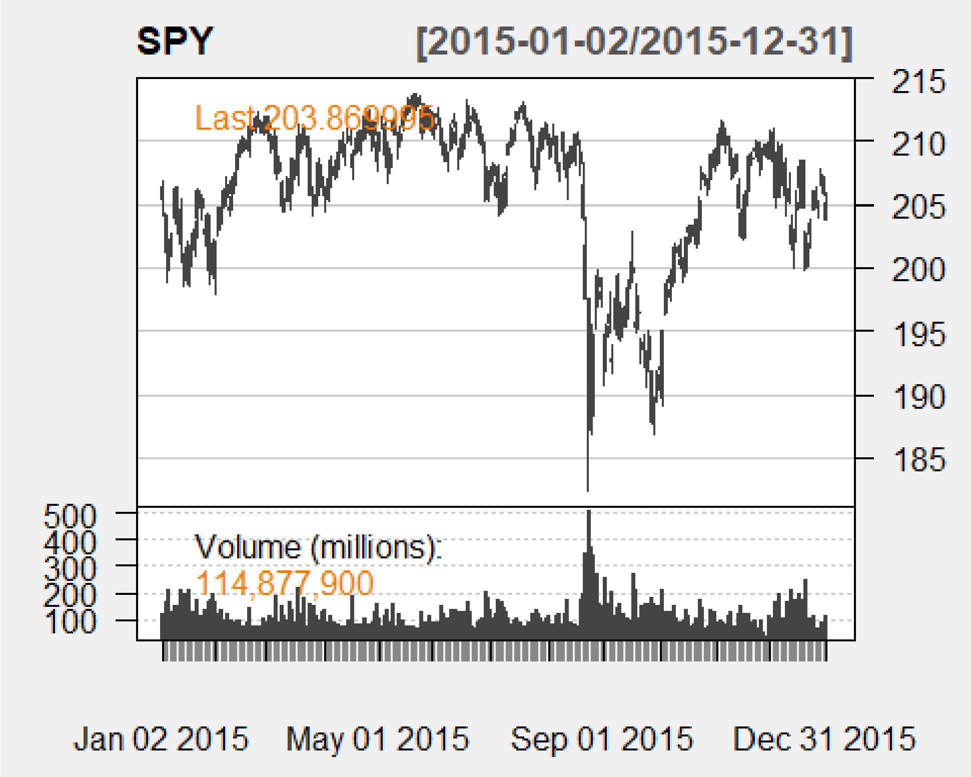
Analyse Stock Market and Index



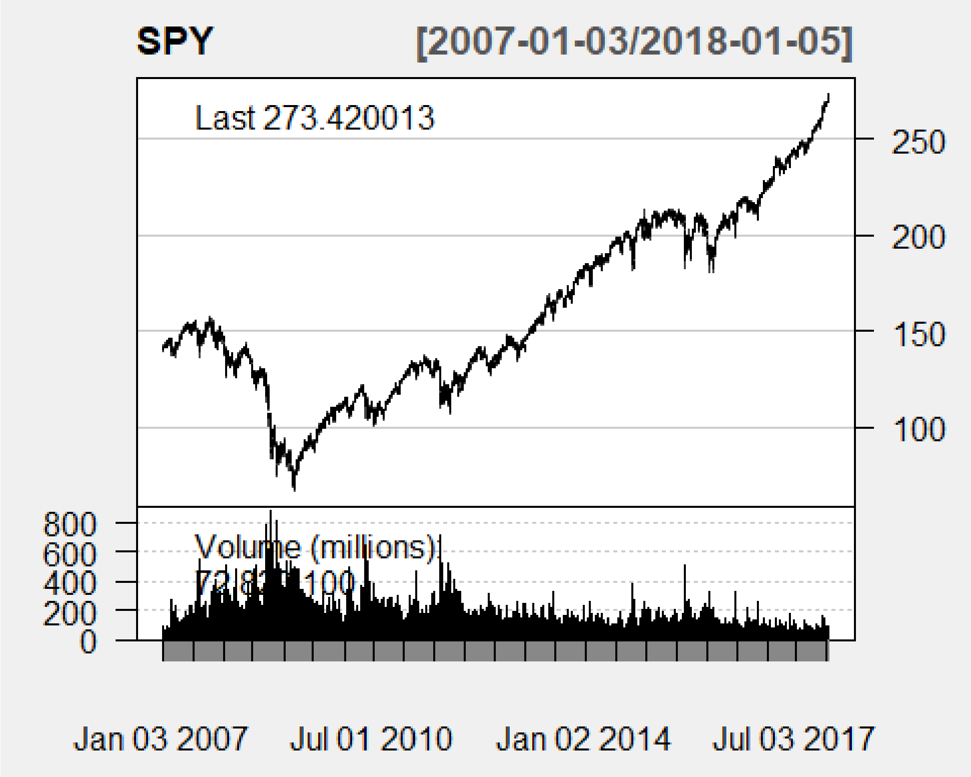
```
chartSeries(SPY, theme="white",
            TA="addVo();addBBands();addCCI()", subset='2015')

For the Year of 2015, I compared the SPY index price vs. volume and commodity
channel index, I found direct relation between Index price and commodity
channel
```



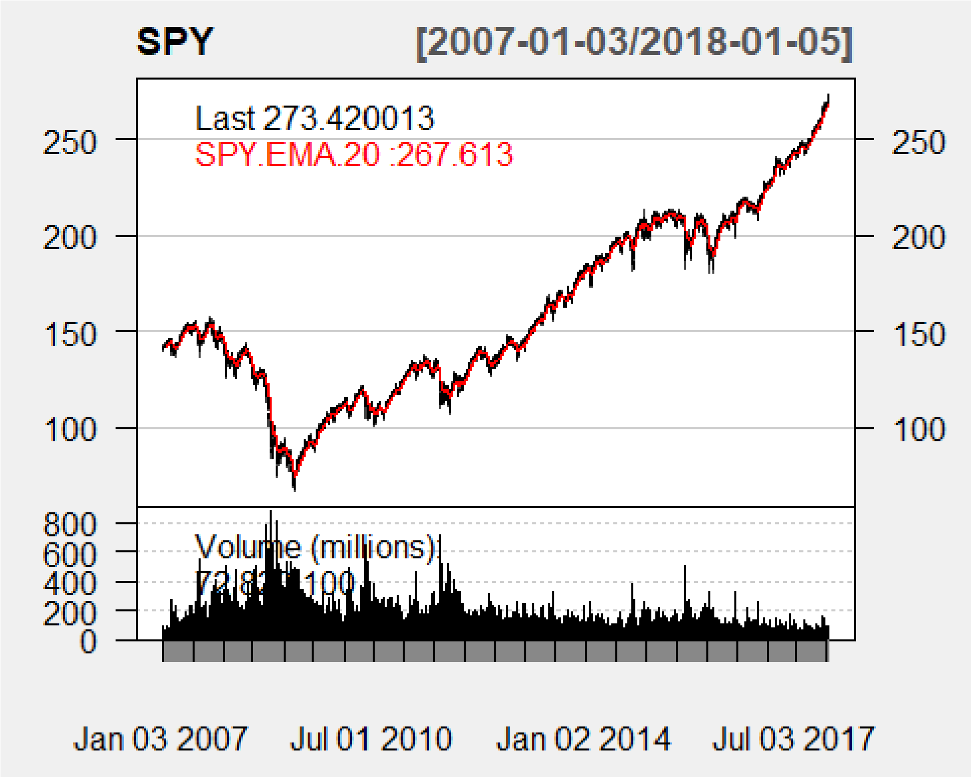
```
chartSeries(SPY, theme="white", subset='2015')
```



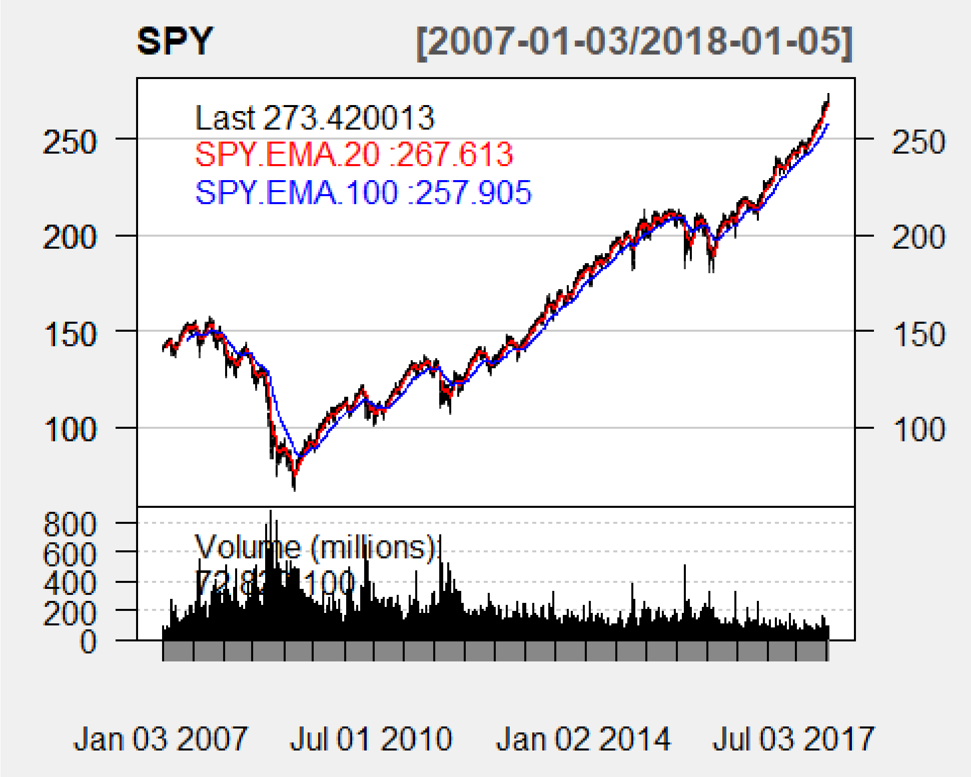
```
chartSeries(SPY, theme=chartTheme('white'), up.col="black",
            dn.col="black")
```



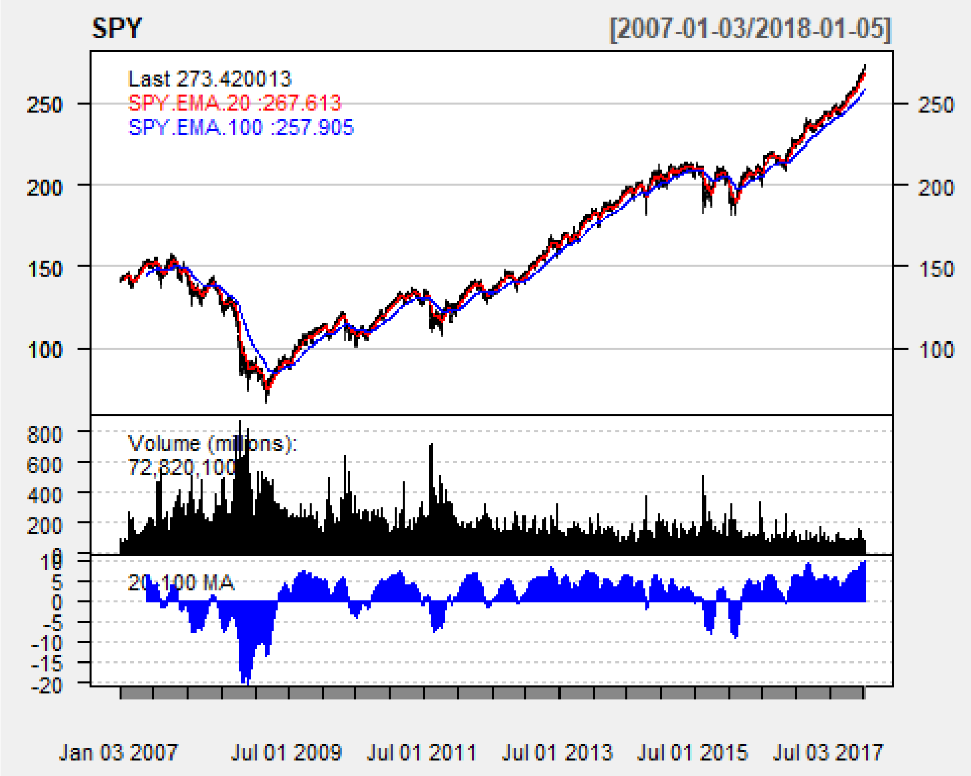
```
SPY.EMA.20<- EMA(SPY$SPY.Close, n=20)
SPY.EMA.100<- EMA(SPY$SPY.Close, n=100)
addTA(SPY.EMA.20, on=1, col = "red")
```



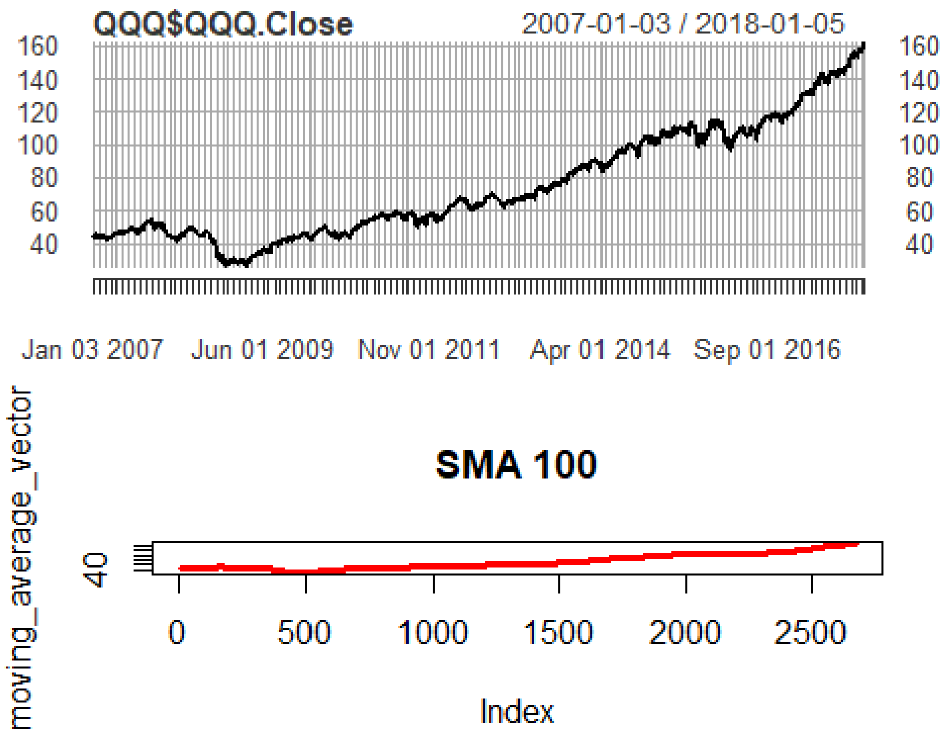
```
addTA(SPY.EMA.100, on=1, col = "blue")
```



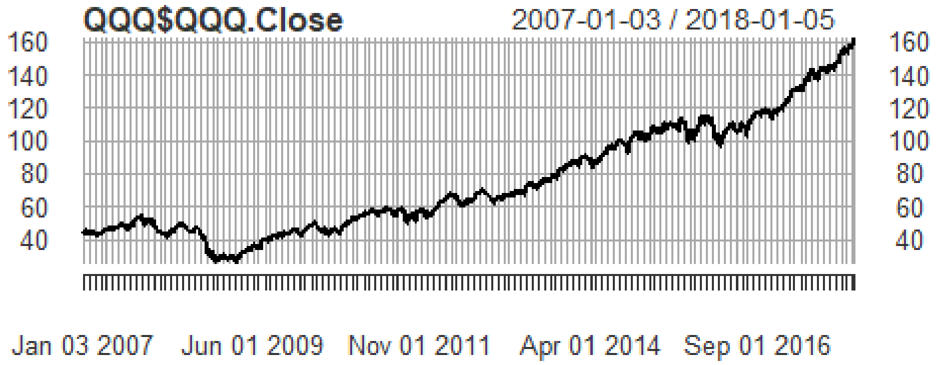
```
addTA(SPY.EMA.20 - SPY.EMA.100,col='blue', type='h',legend="20-100 MA")
```

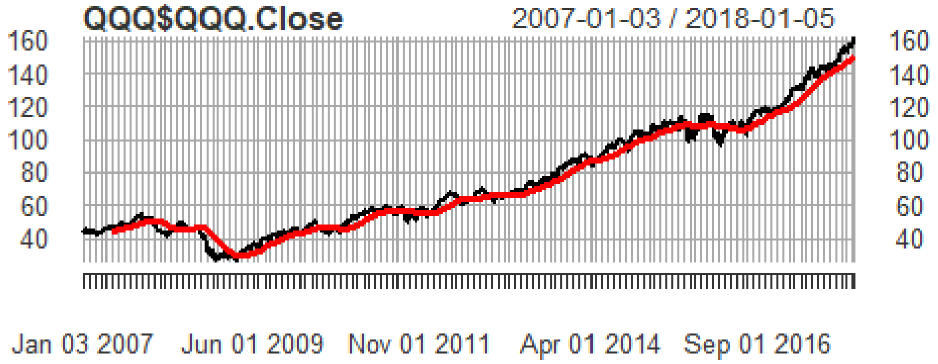


```
# get more inside about Moving Average price
# In the below lines I'm going to explain the SMA
# function that I have used above
print ("STEP 2.4:Creating Moving Average")

## [1] "STEP 2.4:Creating Moving Average"

getSymbols(c('QQQ'), src='google')

## [1] "QQQ"

#I’ll focus on the Close of the bar (where it closed for the day). Let’s take
a quick peek at what we have:
plot(QQQ$QQQ.Close)

#I’ll create a simple function to break down the data and average every price
point by x amount of points prior to it.
#In this case I’ll use a 100 day smoothing period.

period <- 100
price_vector <- QQQ$QQQ.Close
moving_average_vector <- c()
for (ind in seq((period+1),(length(price_vector))) ){
       moving_average_vector <- c(moving_average_vector,
mean(price_vector[(ind-period):ind]))
}

par(mfrow=c(2,1))
plot(QQQ$QQQ.Close)
plot(moving_average_vector, type='l', col='red', lwd=3, main = paste('SMA',
period))
```

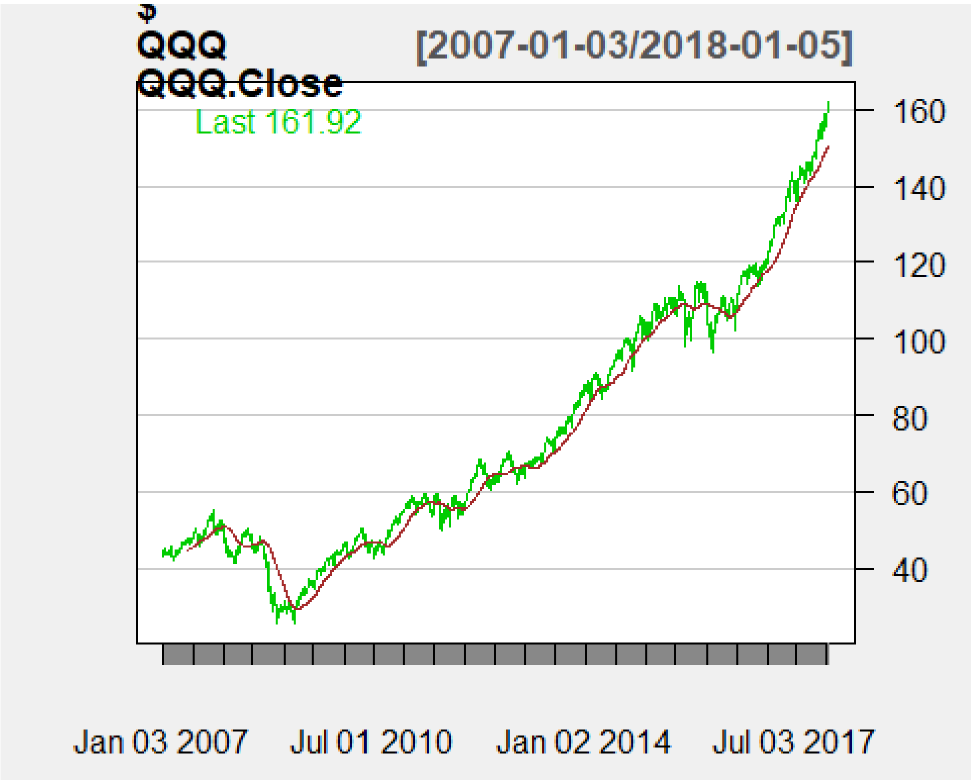


```
#The first plot is the raw QQQ daily closing prices and the second plot, is
our smoothed version. Keep in mind that the first 100 days of price data
#can't be used as that is the minimum data we need to create a 100 period
average.
#The issue we have is our new SMA vector contains 2065 entries, while our the
QQQ market download, has 2165 entries.
#This should be easy to understand as it takes 100 entries to calculate an
SMA.
#This is going to make it difficult to overlay our SMA onto the raw market
data.
#One way around this is to buffer our SMA with 100 NA's.

period <- 100
price_vector <- QQQ$QQQ.Close
moving_average_vector <- c(rep(NA, period))
# moving_average_vector <- c(rep(as.numeric(QQQ$QQQ.Close[period]), period))
for (ind in seq((period+1),(length(price_vector))) ){
      moving_average_vector <- c(moving_average_vector,
mean(price_vector[(ind-period):ind]))
}

# pass it back to our time series object
QQQ$QQQ.Close.SMA <- moving_average_vector

plot(QQQ$QQQ.Close)
lines(QQQ$QQQ.Close.SMA, type='l', col='red', lwd=3)
```

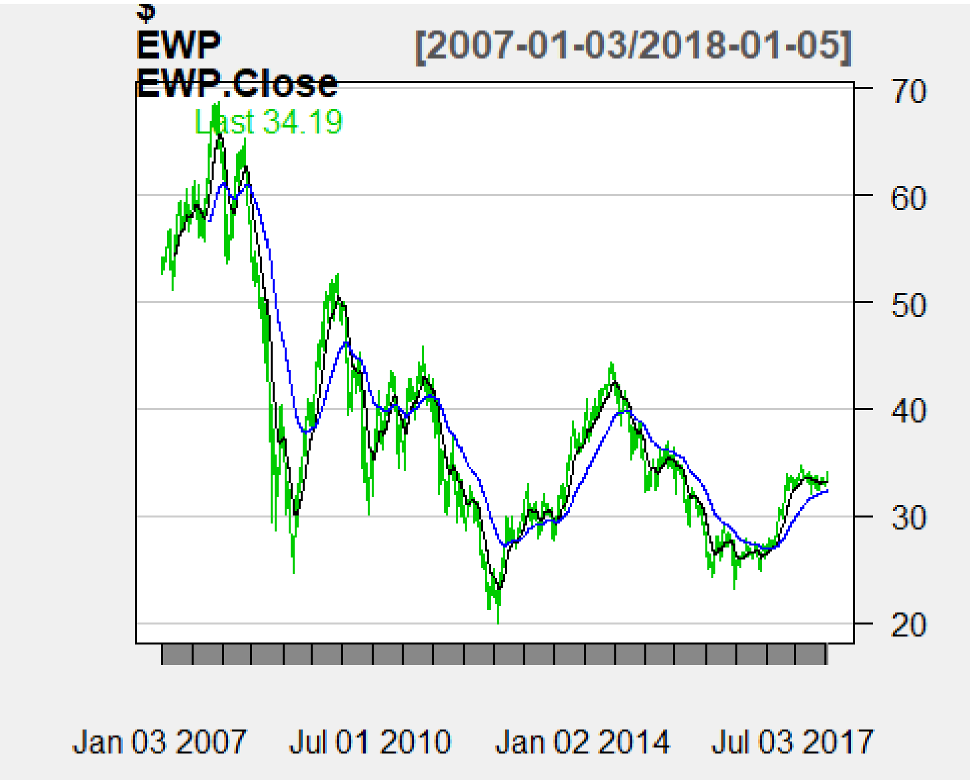




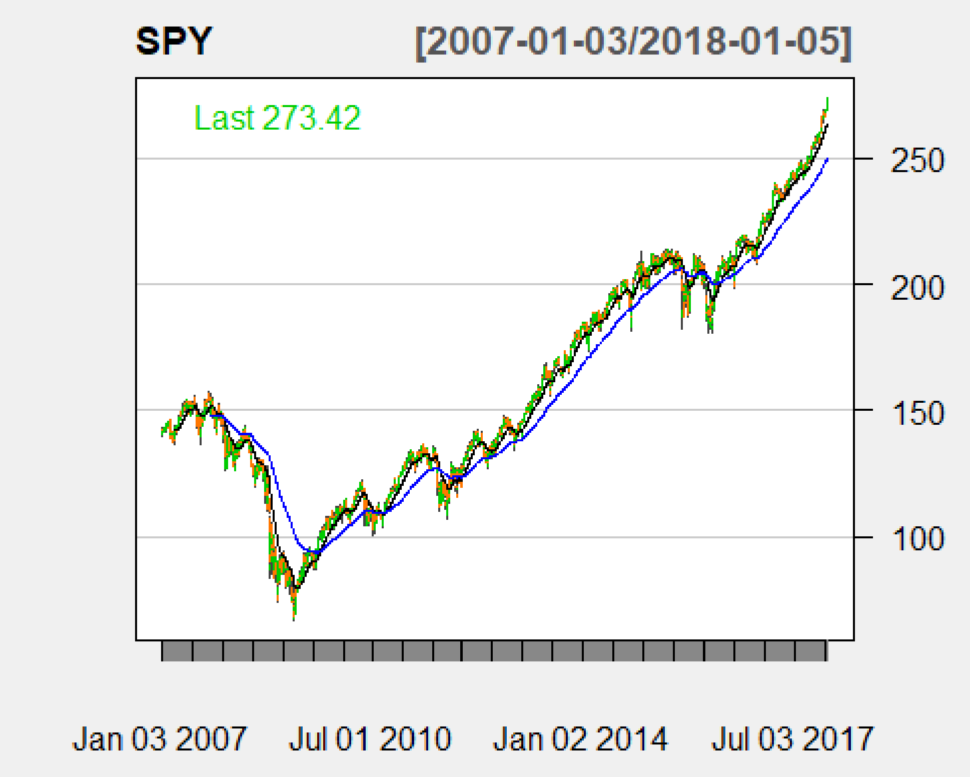
```
# All above action could be simplified by using TTA package same as below:
chartSeries(QQQ$QQQ.Close, theme="white", TA="addSMA(100)")
```



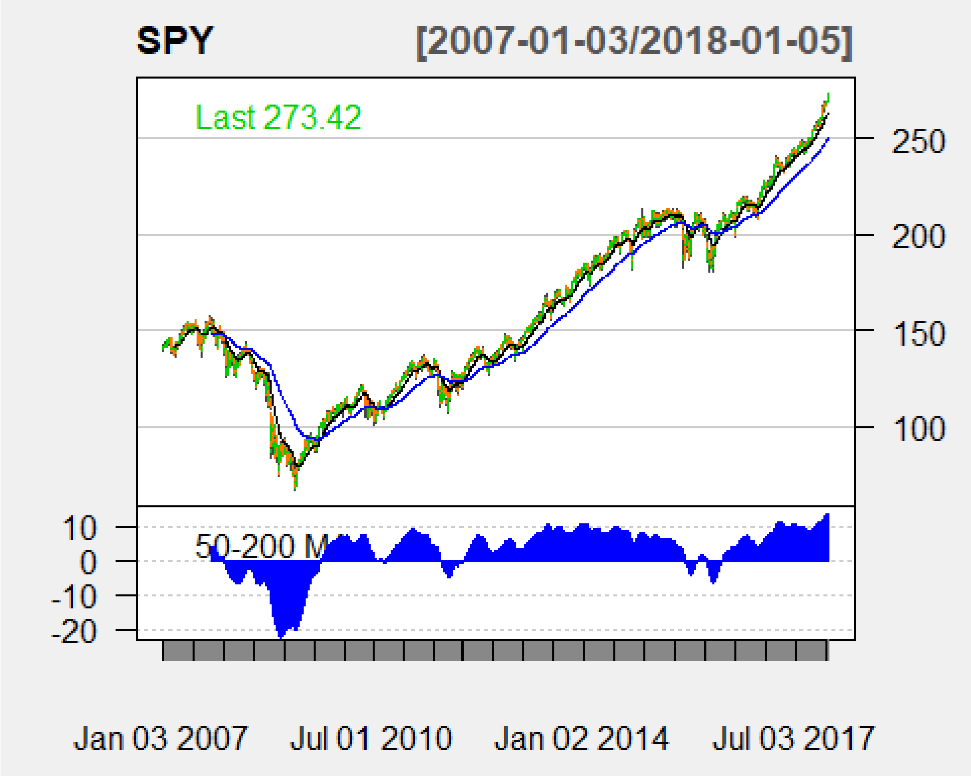
```
# Following the trend with multiple moving avarge
# Looking at multiple moving averages, the 10, 50 & 200 MAs * Detrending
market action
getSymbols(c('EWP', 'SPY'), src='google')

## [1] "EWP" "SPY"

#Let’s chart the data using a 50 and 200-period moving average.
#These are common periods often used as benchmarks to indicate a
strengthening or weakening stock.
chartSeries(EWP$EWP.Close, theme="white", TA="addEMA(50,
col='black');addEMA(200, col='blue')")
```



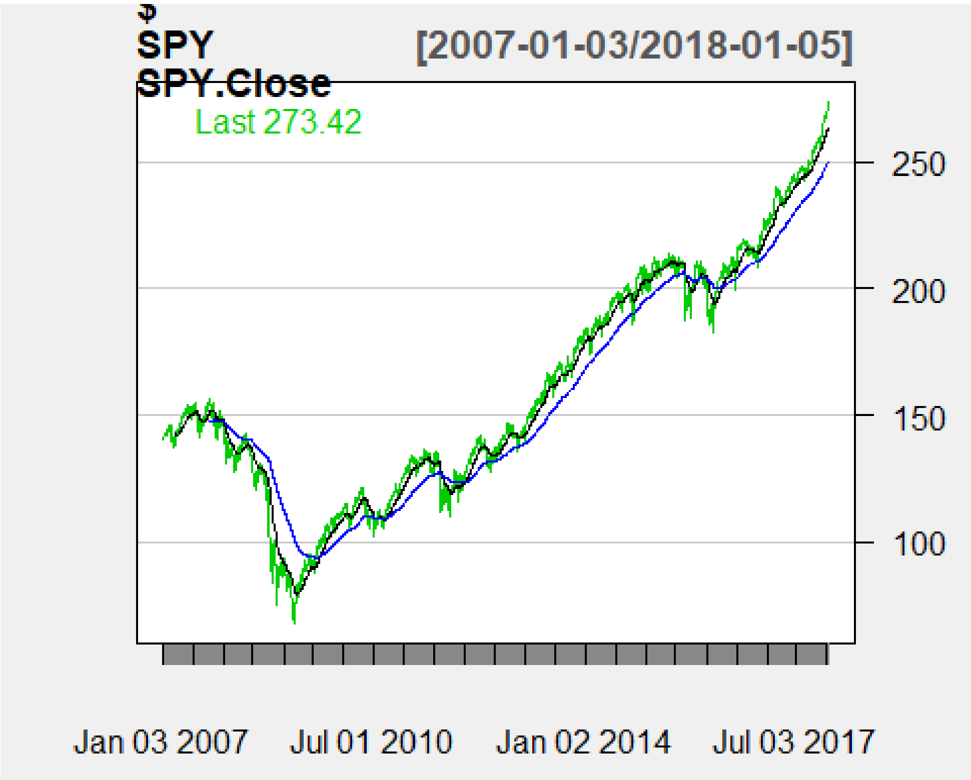
```
chartSeries(SPY, theme="white", TA="addEMA(50, col='black');addEMA(200,
col='blue')")
```



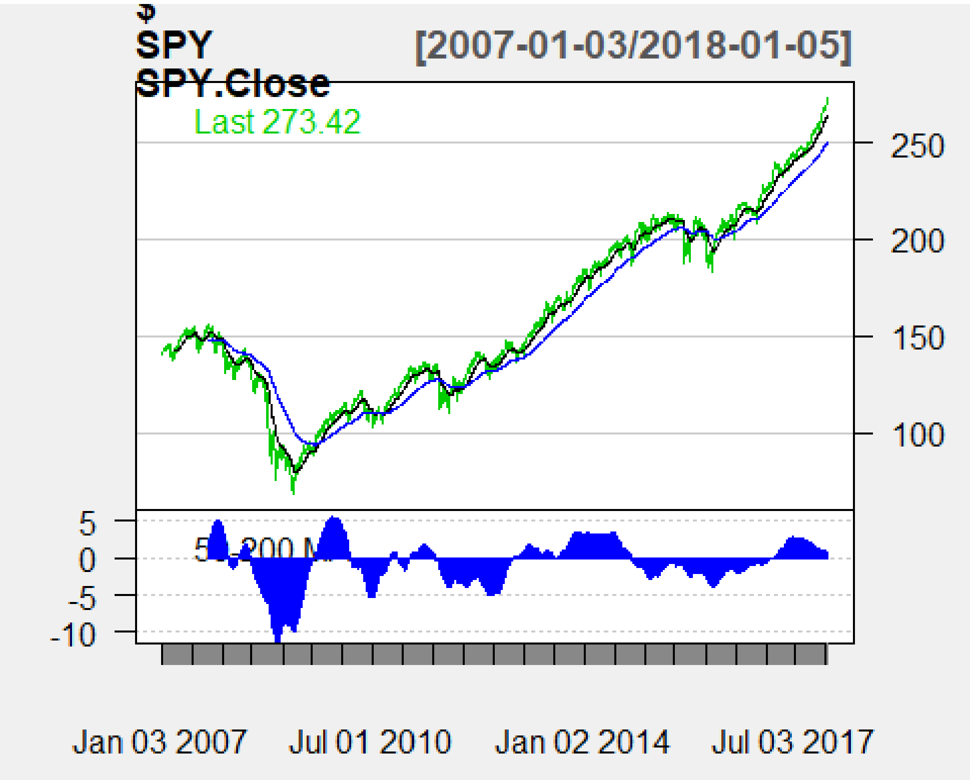
```
#Having two moving averages of different periods removes a lot of the noise.
#When the fast moving average is above the slow one, the market is moving
upwards,
#and when the fast is below the slow, it is going down. Some traders will
look at the
#crossing of these moving averages to take a directional position


SPY.EMA.50<- EMA(SPY$SPY.Close, n=50, )
SPY.EMA.200<- EMA(SPY$SPY.Close, n=200, )
#SPY.EMA.50 fast change
#SPY.EMA.200 slow change
addTA(SPY.EMA.50 - SPY.EMA.200,col='blue', type='h',legend="50-200 MA")
```



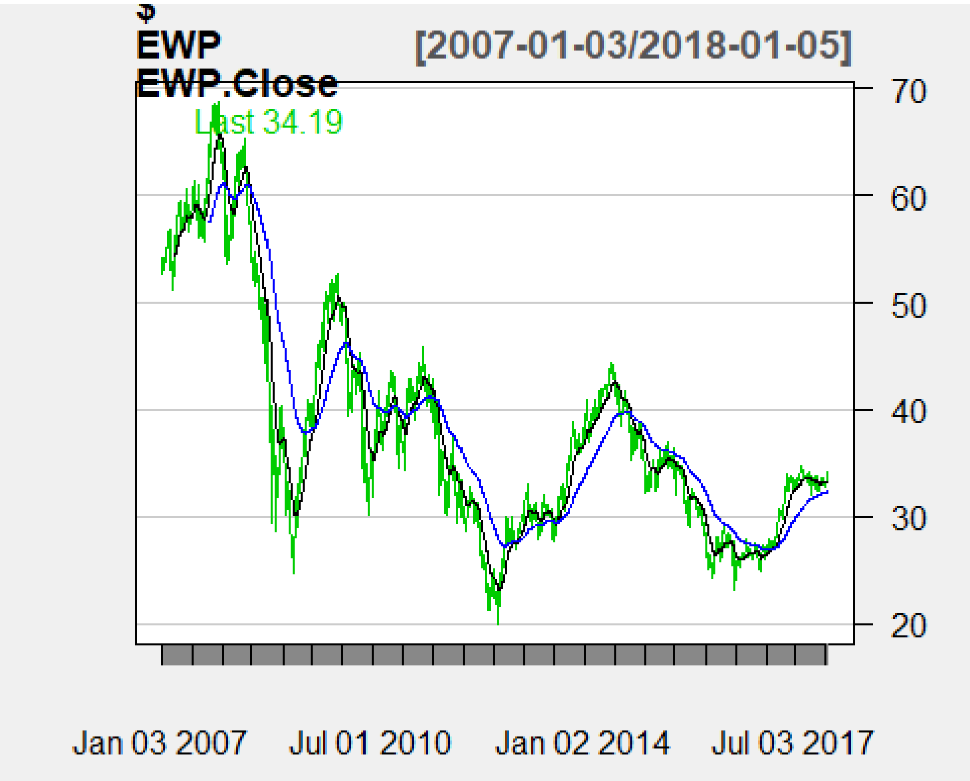
```
chartSeries(SPY$SPY.Close, theme="white", TA="addEMA(50,
col='black');addEMA(200, col='blue')")
```



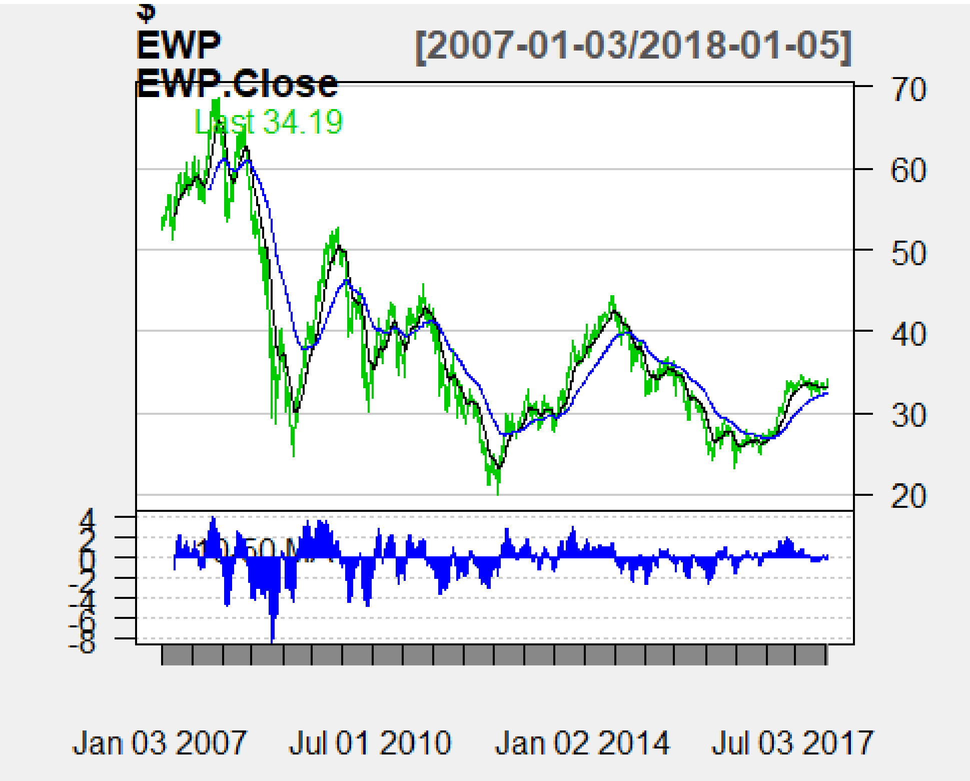
```
EWP.EMA.50 <- EMA(EWP$EWP.Close, n=50, )
EWP.EMA.200 <- EMA(EWP$EWP.Close, n=200, )
addTA(EWP.EMA.50 - EWP.EMA.200, col='blue', type='h',legend="50-200 MA")
```



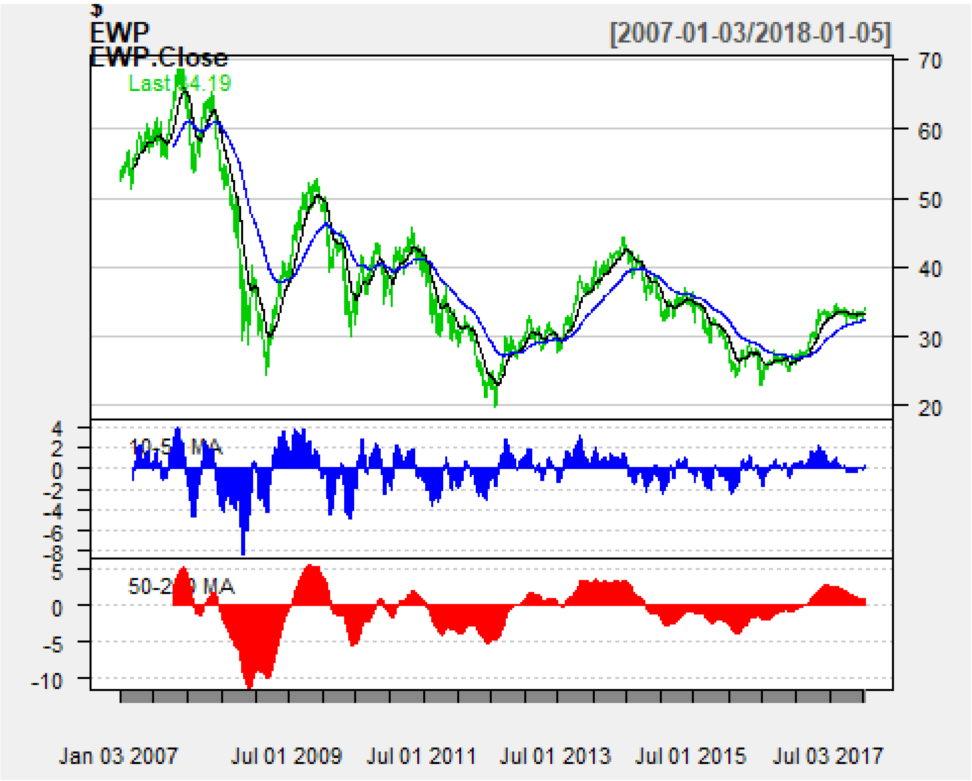
```
chartSeries(EWP$EWP.Close, theme="white", TA="addEMA(50,
col='black');addEMA(200, col='blue')")
```



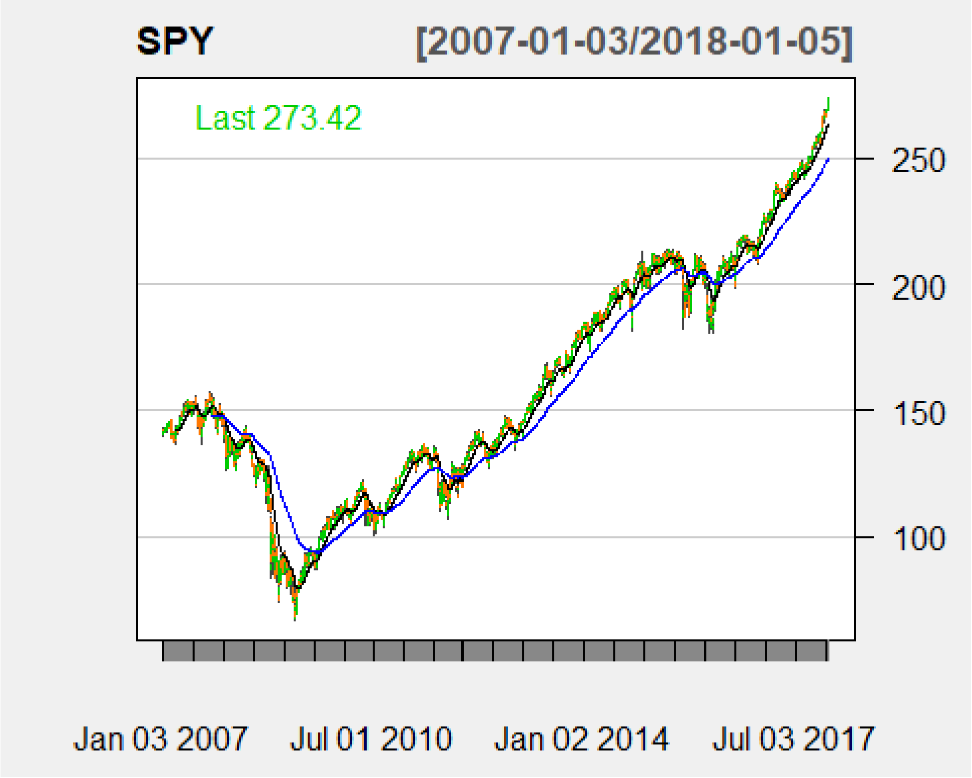
```
# everyting below Zero - You should not be long - and keep the Index , Holding
# everything above Zero - You should not be short - and sell the Index , Holding
# Let's look into three avarage moving , I'm adding 10 period

EWP.EMA.10 <- EMA(EWP$EWP.Close, n=10, )
EWP.EMA.50 <- EMA(EWP$EWP.Close, n=50, )
EWP.EMA.200 <- EMA(EWP$EWP.Close, n=200, )
Fast.Diff <- EWP.EMA.10 - EWP.EMA.50
Slow.Diff <- EWP.EMA.50 - EWP.EMA.200
addTA(Fast.Diff, col='blue', type='h',legend="10-50 MA")
```



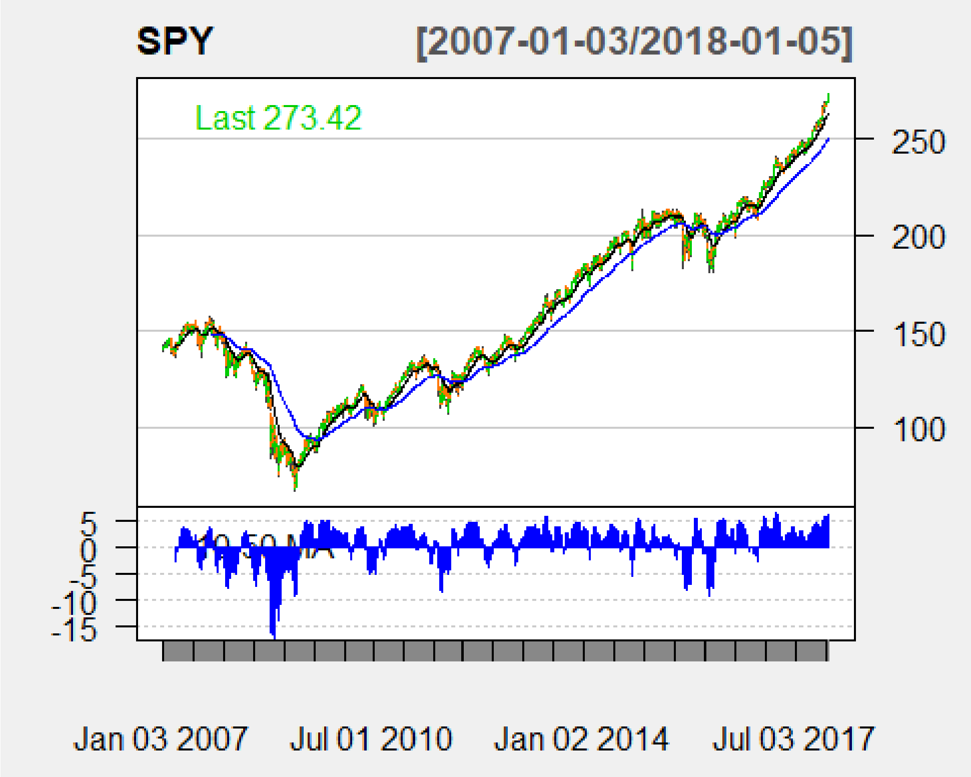
```
addTA(Slow.Diff, col='red', type='h',legend="50-200 MA")
```



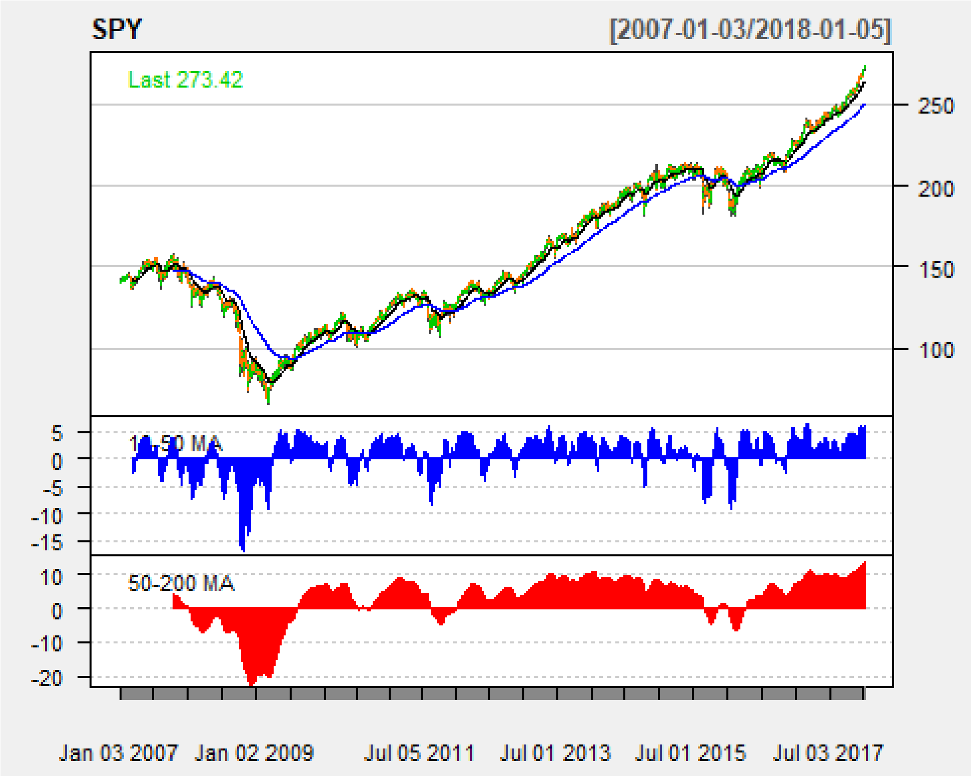
```
chartSeries(SPY, theme="white", TA="addEMA(50, col='black');addEMA(200,
col='blue')")
```



```
SPY.EMA.10 <- EMA(SPY$SPY.Close, n=10, )
SPY.EMA.50 <- EMA(SPY$SPY.Close, n=50, )
SPY.EMA.200 <- EMA(SPY$SPY.Close, n=200, )
Fast.Diff <- SPY.EMA.10 - SPY.EMA.50
Slow.Diff <- SPY.EMA.50 - SPY.EMA.200
addTA(Fast.Diff, col='blue', type='h',legend="10-50 MA")
```



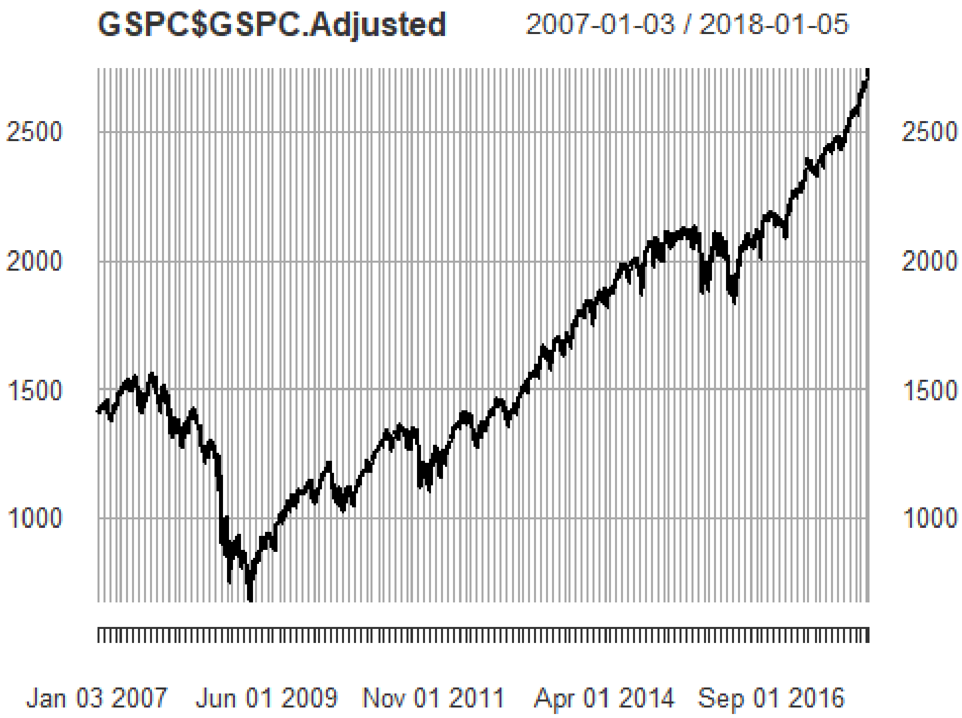
```
addTA(Slow.Diff, col='red', type='h',legend="50-200 MA")
```



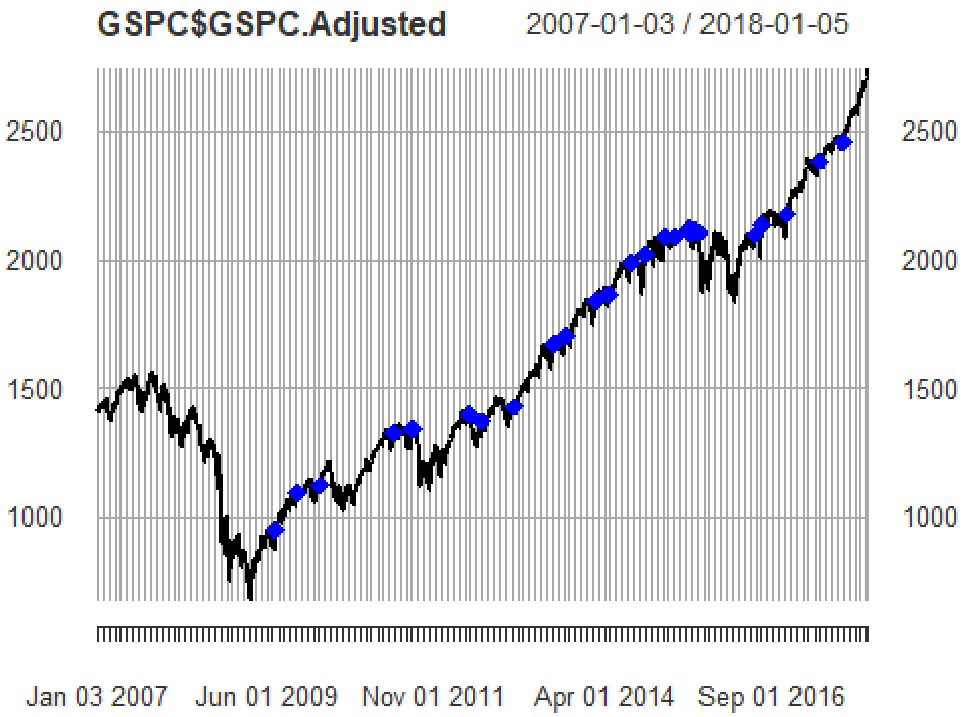
```
#Trading With The Trend

#You can only enter in the direction of the red Slow.Diff indicator,
#if its above zero you can take long signals, if its below zero,
#you can take short signals. The Fast.Diff indicator dictates the entries.
#When the blue line goes from negative to positive, its a long trade (and the
slower red Slow.Diff indicator is above zero).
#Same thing for shorts. This is also referred to as a moving average
crossover trading system.

#To run this system, we need to build rules to hunt them down.

#The rules are:

#    if no position: red > 0 and blue-1 < 0 and blue > 0 go long
#    if long: blue < 0 exit long

#    if no position: red < 0 and blue-1 > 0 and blue < 0 go short
#    if short: blue > 0 exit short
# New chalange would to find the blue -1 means, meaning lag of blue, Pre.
price .
print ("STEP 2.5:Trading With The Trend")

## [1] "STEP 2.5:Trading With The Trend"

library(binhf)
```

```
## Warning: package 'binhf' was built under R version 3.3.3

## Loading required package: wavethresh

## Warning: package 'wavethresh' was built under R version 3.3.3

## Loading required package: MASS

## WaveThresh: R wavelet software, release 4.6.8, installed

## Copyright Guy Nason and others 1993-2016

## Note: nlevels has been renamed to nlevelsWT

## Loading required package: adlift

## Warning: package 'adlift' was built under R version 3.3.3

## Loading required package: EbayesThresh

## Warning: package 'EbayesThresh' was built under R version 3.3.3

## 
##  **********************************************
##  adlift: a package to perform wavelet lifting schemes
## 
##  --- Written by Matt Nunes and Marina Knight ---
##    Current package version:  1.3-3  ( 2017-09-13 )
## 
##              -+ packaged by MAN +-
##  **********************************************
## 
##  adlift 1.3-3 loaded

## 
## Attaching package: 'adlift'

## The following object is masked from 'package:EbayesThresh':
## 
##     postmean.cauchy

## 
##  **********************************************
##  binhf: Haar-Fisz functions for binomial data
## 
##  --- Written by Matt Nunes ---
##    Current package version:  1.0-1  ( 24/04/2014 )
## 
## 
##  **********************************************
## 
##  binhf 1.0-1 loaded
```

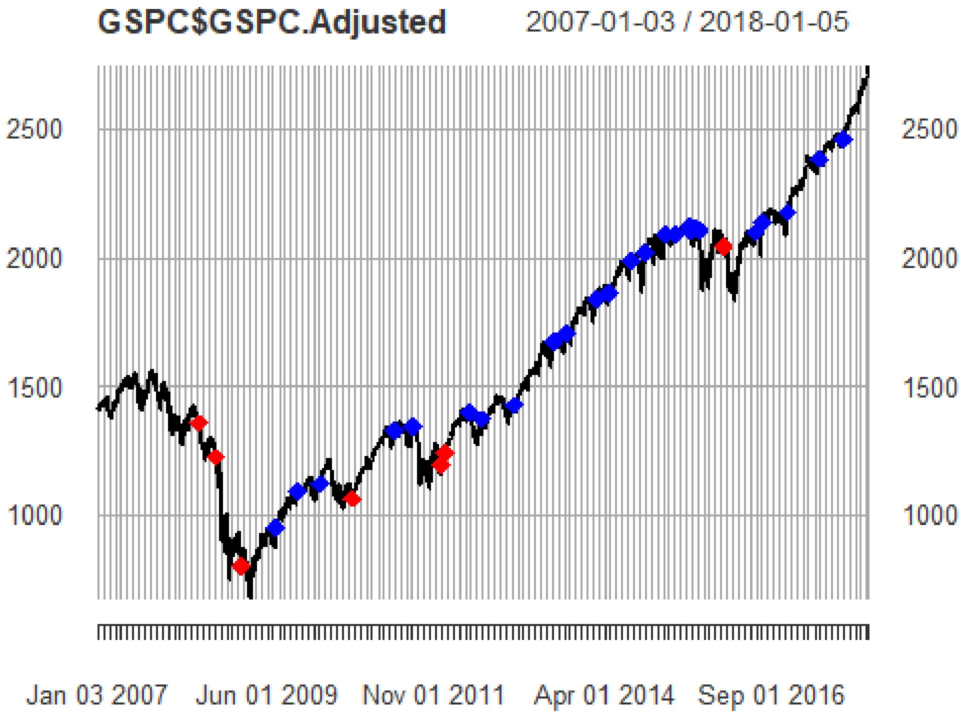
```
##
## Attaching package: 'binhf'

## The following objects are masked from 'package:EbayesThresh':
##
##     negloglik.laplace, wandafromx

## The following object is masked from 'package:wavethresh':
##
##     madmad

## The following object is masked from 'package:base':
##
##     norm

tail(as.numeric(Fast.Diff))

## [1] 5.322173 5.087720 5.139512 5.386004 5.703934 6.171299

# return prev. data
tail(shift(v=as.numeric(Fast.Diff), places=1, dir="right"))

## [1] 5.388930 5.322173 5.087720 5.139512 5.386004 5.703934

#This allows us to compare the values of two different rows on the same row.
#We still have our indicator value of today, but we now can compare it with
yesterday’s value on the same row.
#Sure, we could have just easily created a loop and run through each value
but by doing it this way we stick to vector comparison in its simplest form.

#Now, let’s translate our trend trading system pseudo code into R code:
#Note: Closing price won't give us best price since compay pays dividend /
interest and this price is not accure ah the end of the
# month, Hence I have used Adjusted price.

GSPC.SMA.10 <- SMA(GSPC$GSPC.Adjusted, n=10, )
GSPC.SMA.50 <- SMA(GSPC$GSPC.Adjusted, n=50, )
GSPC.SMA.200 <- SMA(GSPC$GSPC.Adjusted, n=200, )
Fast.Diff <- GSPC.SMA.10 - GSPC.SMA.50
Slow.Diff <- GSPC.SMA.50 - GSPC.SMA.200

# look for long entries
Long_Trades <- ifelse(
Slow.Diff  > 0 &
Fast.Diff  > 0 &
shift(v=as.numeric(Fast.Diff), places=1, dir="right") < 0,
GSPC$GSPC.Adjusted, NA)

# look for long exits (same thing but inverse signts)
Short_Trades <- ifelse(
Slow.Diff  < 0 &
Fast.Diff  < 0 &
shift(v=as.numeric(Fast.Diff), places=1, dir="right") > 0,
GSPC$GSPC.Adjusted, NA)
plot(GSPC$GSPC.Adjusted)
```



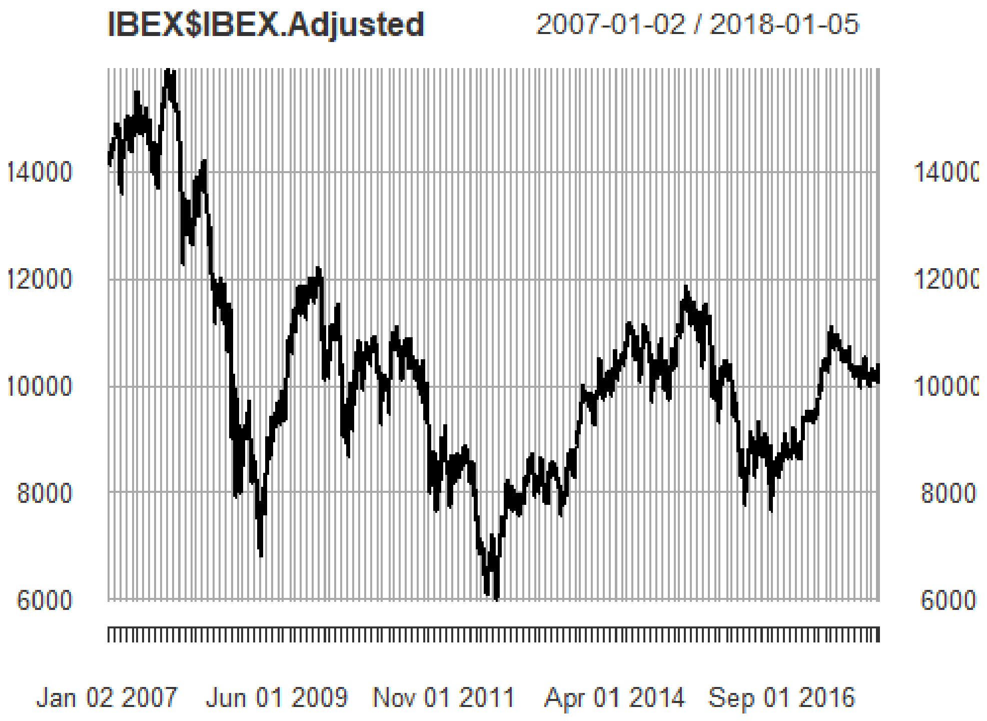
```
## Warning in plot.xts(EWP): only the univariate series will be plotted
points(Long_Trades, col='blue', cex=1.5, pch=18)
```



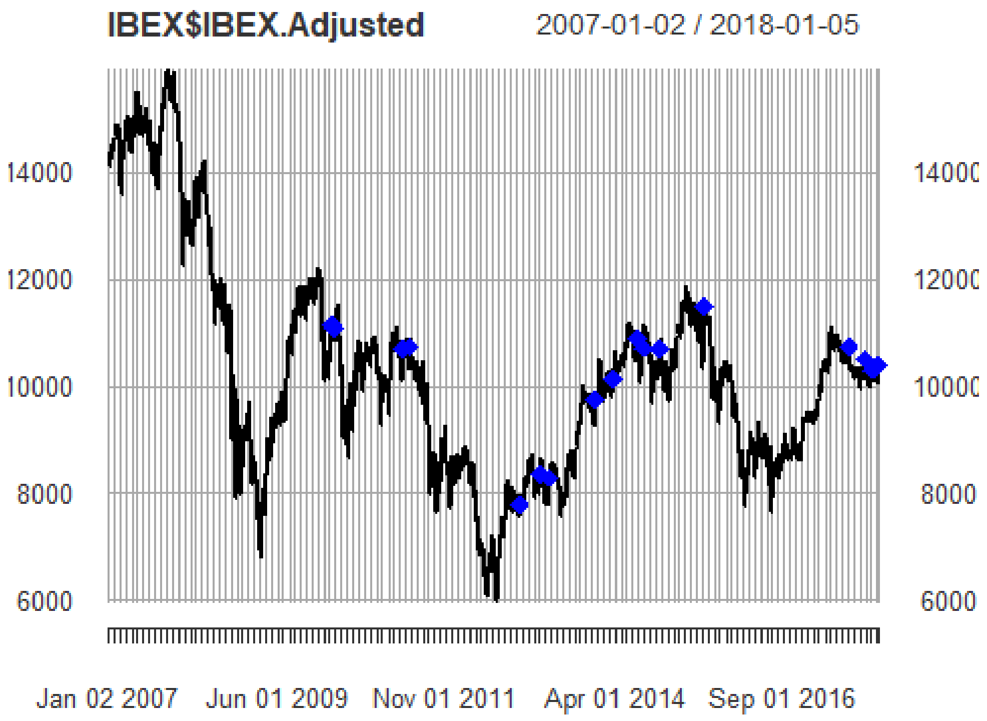
```
points(Short_Trades, col='red', cex=1.5, pch=18)
```



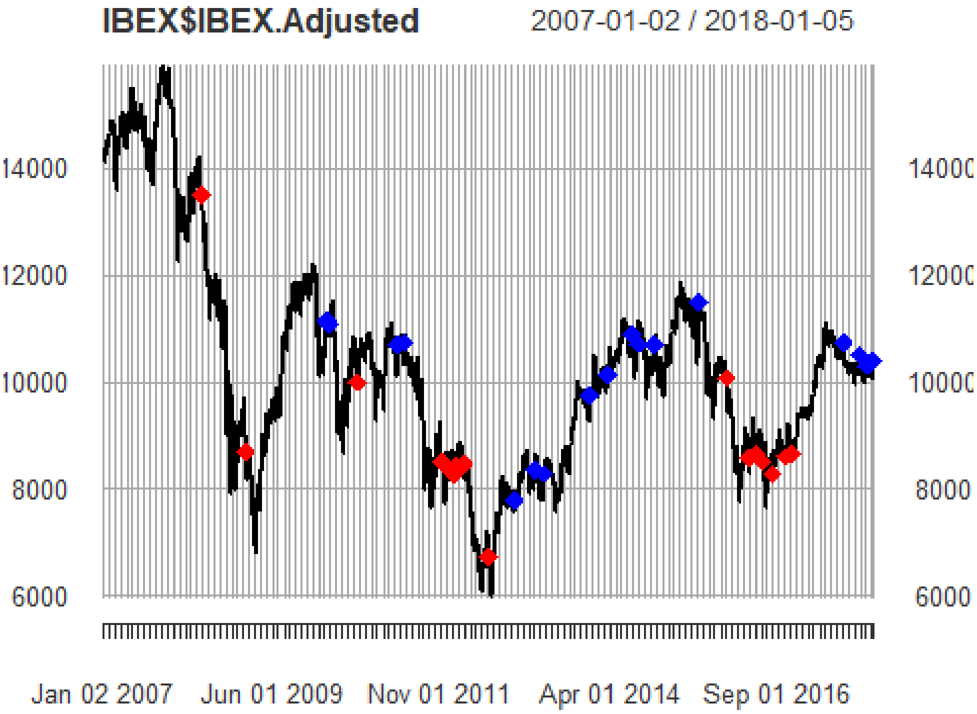
```
#Mixture of entry points and that is usually how it works on a trading,
bouncing trend.
#Though we aren’t going to design full trending systems here, a stop-loss
exit order is key to any directional
#trading so you don’t lose everything! Let’s see what it does on trending
market:



IBEX.EMA.10 <- EMA(IBEX$IBEX.Adjusted, n=10 )
IBEX.EMA.50 <- EMA(IBEX$IBEX.Adjusted, n=50, )
IBEX.EMA.200 <- EMA(IBEX$IBEX.Adjusted, n=200, )
Fast.Diff <- IBEX.EMA.10 - IBEX.EMA.50
Slow.Diff <- IBEX.EMA.50 - IBEX.EMA.200

# look for long entries
Long_Trades <- ifelse(
  Slow.Diff  > 0 &
    Fast.Diff  > 0 &
    shift(v=as.numeric(Fast.Diff), places=1, dir="right") < 0,
IBEX$IBEX.Adjusted, NA)

# look for long exits (same thing but inverse signts)
Short_Trades <- ifelse(
  Slow.Diff  < 0 &
    Fast.Diff  < 0 &
    shift(v=as.numeric(Fast.Diff), places=1, dir="right") > 0,
IBEX$IBEX.Adjusted, NA)

plot(IBEX$IBEX.Adjusted)
```



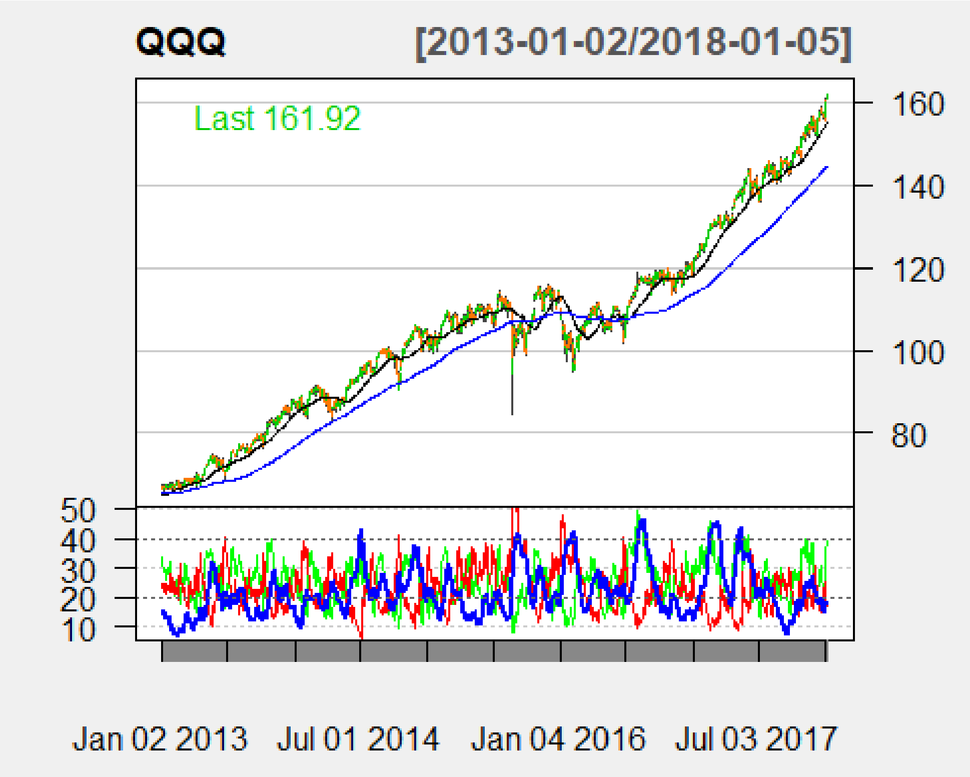
```
points(Long_Trades, col='blue', cex=1.5, pch=18)
```



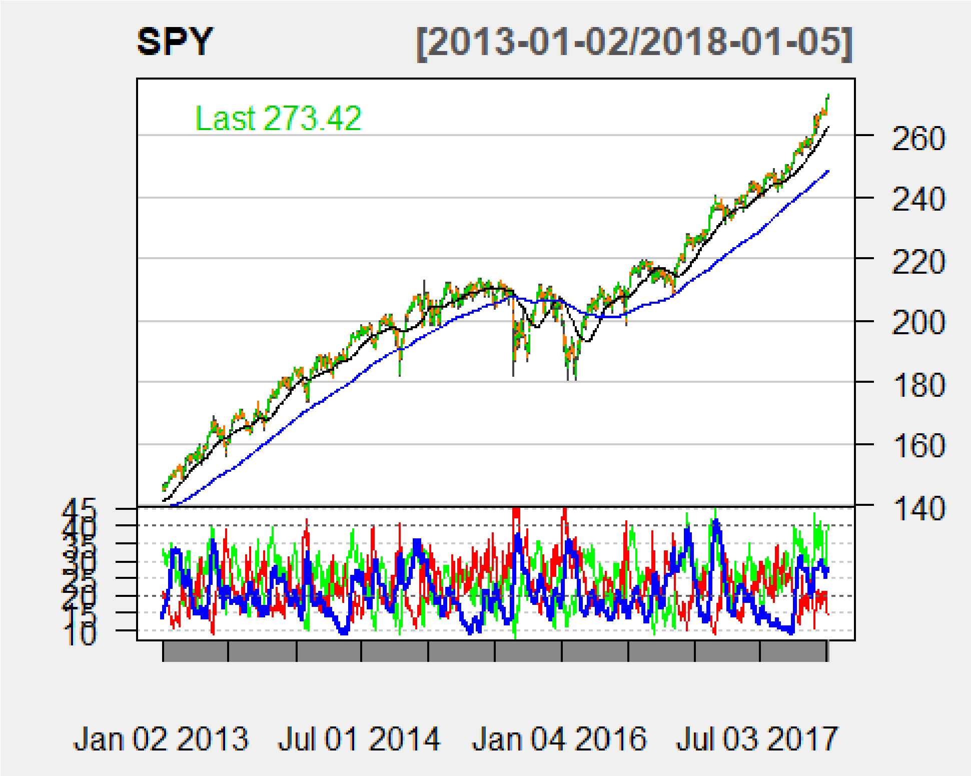
```
points(Short_Trades, col='red', cex=1.5, pch=18)
```



```
print ("STEP 2.6:Volume-based indicators")

## [1] "STEP 2.6:Volume-based indicators"

library(quantmod)
getSymbols(c('QQQ', 'SPY'), src='google')

## [1] "QQQ" "SPY"

# remove any NAs
QQQ <- QQQ[!(rowSums(is.na(QQQ))),]
SPY <- SPY[!(rowSums(is.na(SPY))),]

library(TTR)

#The ADX is Welles Wilder’s Directional Movement Indicator. It is used by
lots of people to determine if the market is trending or range bound.
# Refrence: https://en.wikipedia.org/wiki/Average_directional_movement_index
chartSeries(QQQ, theme="white", TA="addSMA(50, col='black');addSMA(200,
col='blue');addADX(n = 14, maType='EMA', wilder=TRUE)", subset='2013::')
```

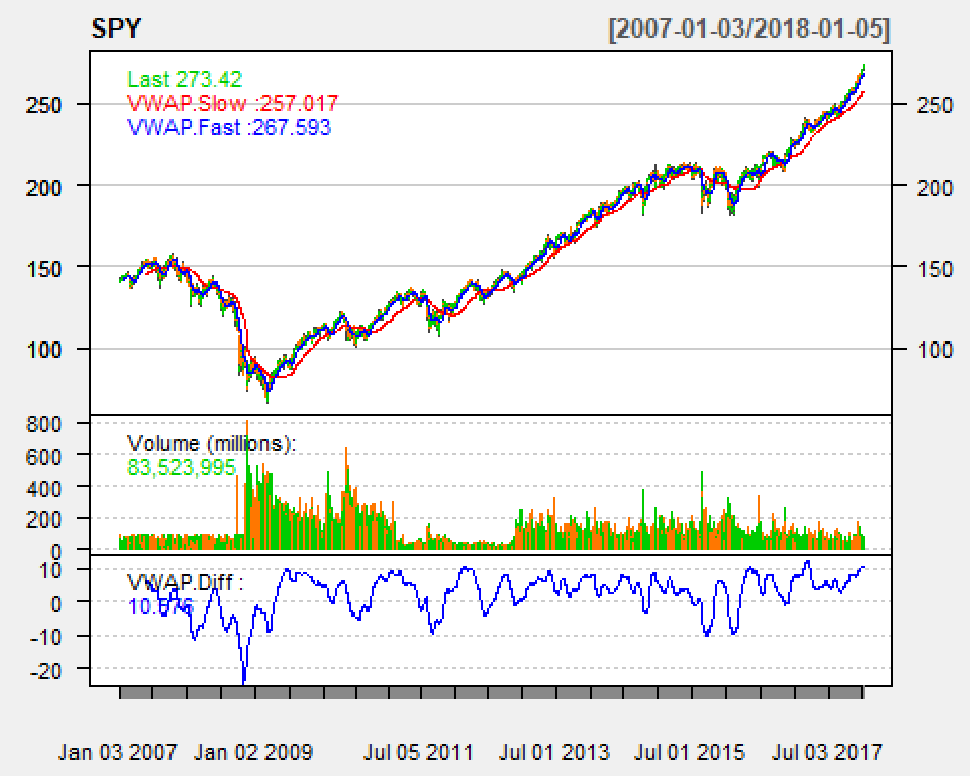


```
# Look into price as of 2013 and onward
chartSeries(SPY, theme="white", TA="addSMA(50, col='black');addSMA(200,
col='blue');addADX(n = 14, maType='EMA', wilder=TRUE)", subset='2013::')
```

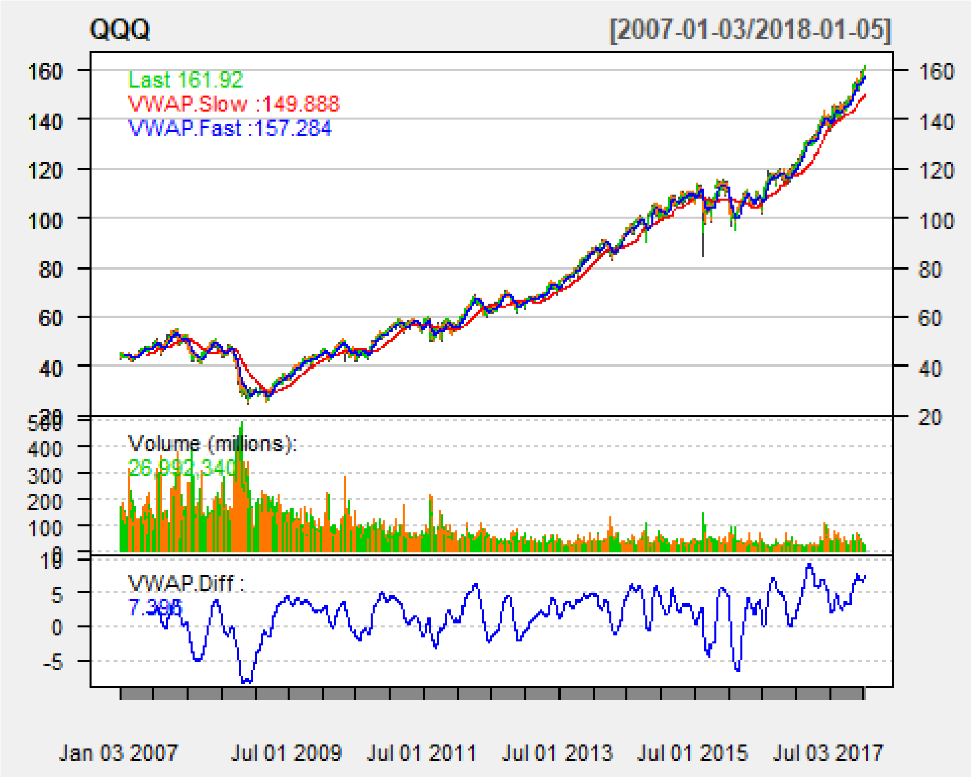


```
#In a nutshell, Welles recommends using the ADX with a 14-day period. When
the main blue line is above 20, it is considered a strong,
#trending market, when it is below, it is considered a weak one.
#Volume

#As this is an introductory course, we’re mostly using the closing price but
it is important to note that there are a lot of other market variables
available.
#You can design systems with the open price, the high or low, the difference
between the open and close, etc. And there is also the volume.

#This an important indicator. A falling stack on rising volume or a rising
stock on falling volume may mean the move is about to
#reverse. Whatever the reason for abnormal volume, it should be a warning to
keep a vigilant eye on the stock.

#There are plenty of indicators that include the volume price such as the
Volume-weighted average price (VWAP).
#The VWAP is a guide more than a trading indicator as to where the market is
trading compared to the volume adjusted price.
#It divides dollars traded by volume (see above link for more details).

VWAP.Slow <- VWAP(price=SPY$SPY.Close, volume=SPY$SPY.Volume, n=100)
VWAP.Fast <- VWAP(price=SPY$SPY.Close, volume=SPY$SPY.Volume, n=20)
VWAP.Diff <- VWAP.Fast- VWAP.Slow
```

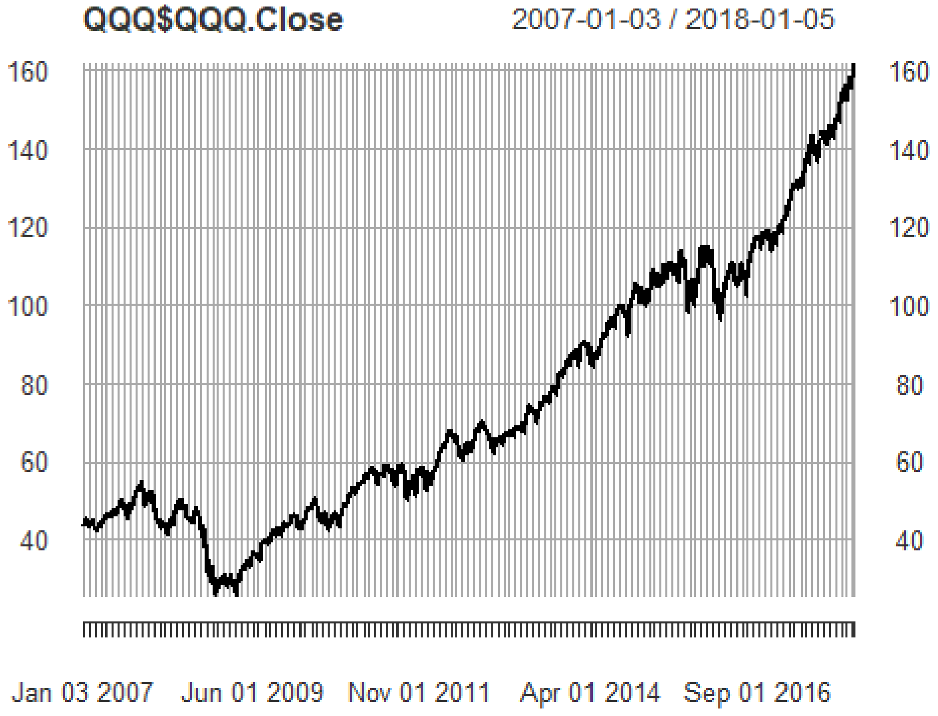
```
chartSeries(SPY, theme="white", TA="addVo();addTA(VWAP.Slow, on=1,
col='red');addTA(VWAP.Fast, on=1, col='blue');addTA(VWAP.Diff, col='blue')")
```



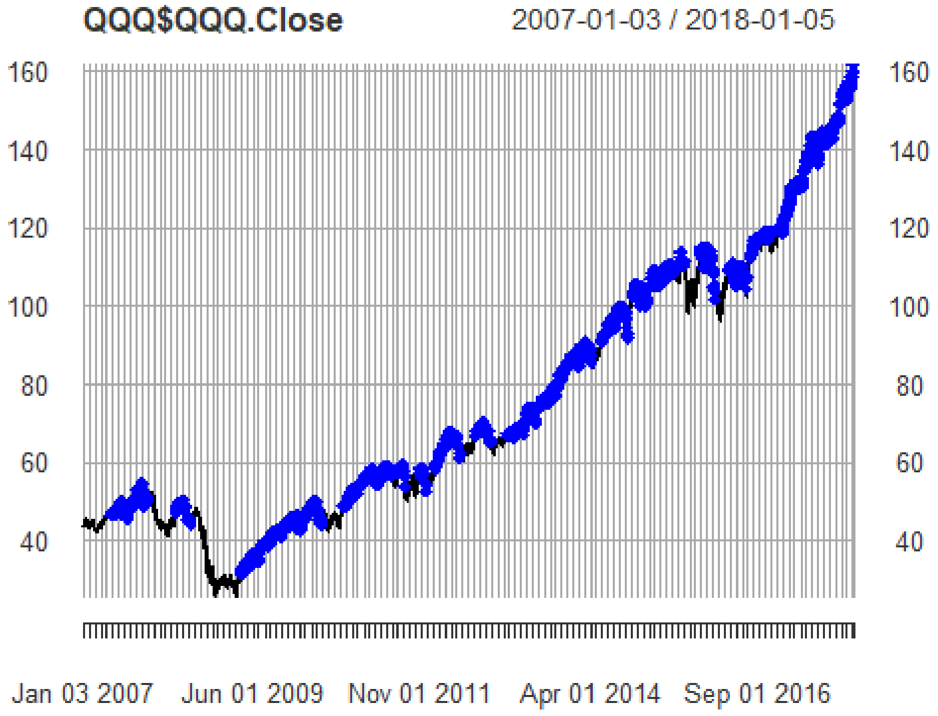
```
# QQQ
VWAP.Slow <- VWAP(price=QQQ$QQQ.Close, volume=QQQ$QQQ.Volume, n=100)
VWAP.Fast <- VWAP(price=QQQ$QQQ.Close, volume=QQQ$QQQ.Volume, n=20)
VWAP.Diff <- VWAP.Fast- VWAP.Slow

chartSeries(QQQ, theme="white", TA="addVo();addTA(VWAP.Slow, on=1,
col='red');addTA(VWAP.Fast, on=1, col='blue');addTA(VWAP.Diff, col='blue')")
```



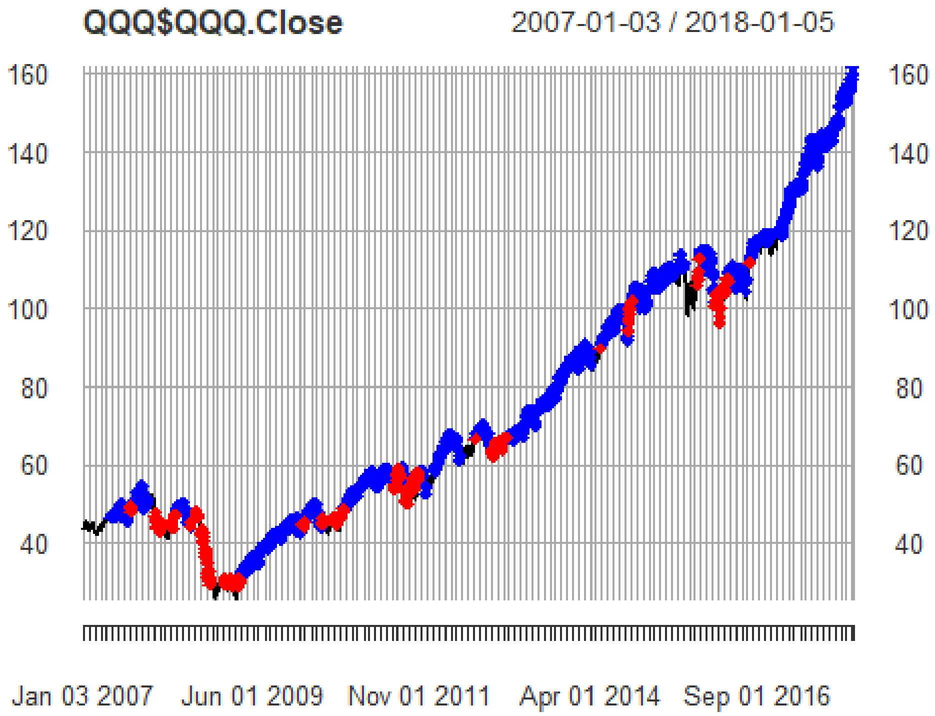
```
ADX.20 <- ADX(QQQ,n=14)

# look for long entries
Long_Trades <- ifelse(
  ADX.20$ADX > 20 &
    VWAP.Diff> 0, QQQ$QQQ.Close, NA)

# look for long entries
Short_Trades <- ifelse(
  ADX.20$ADX > 20 &
    VWAP.Diff < 0, QQQ$QQQ.Close, NA)

plot(QQQ$QQQ.Close)
```



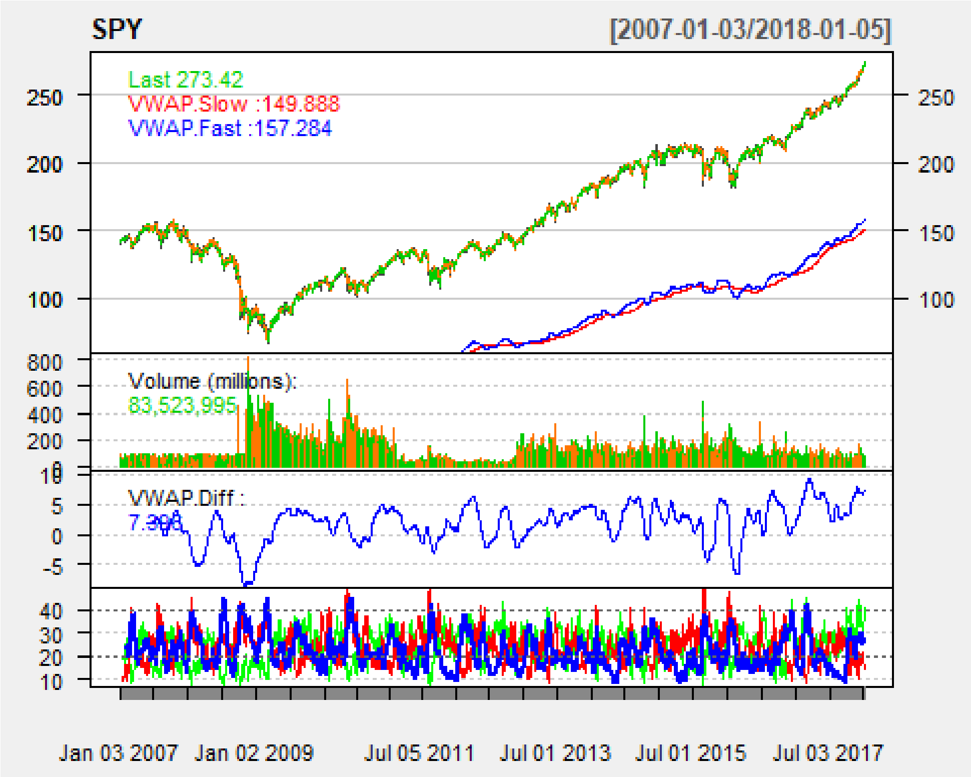
```
points(Long_Trades, col='blue', cex=1, pch=18)
```



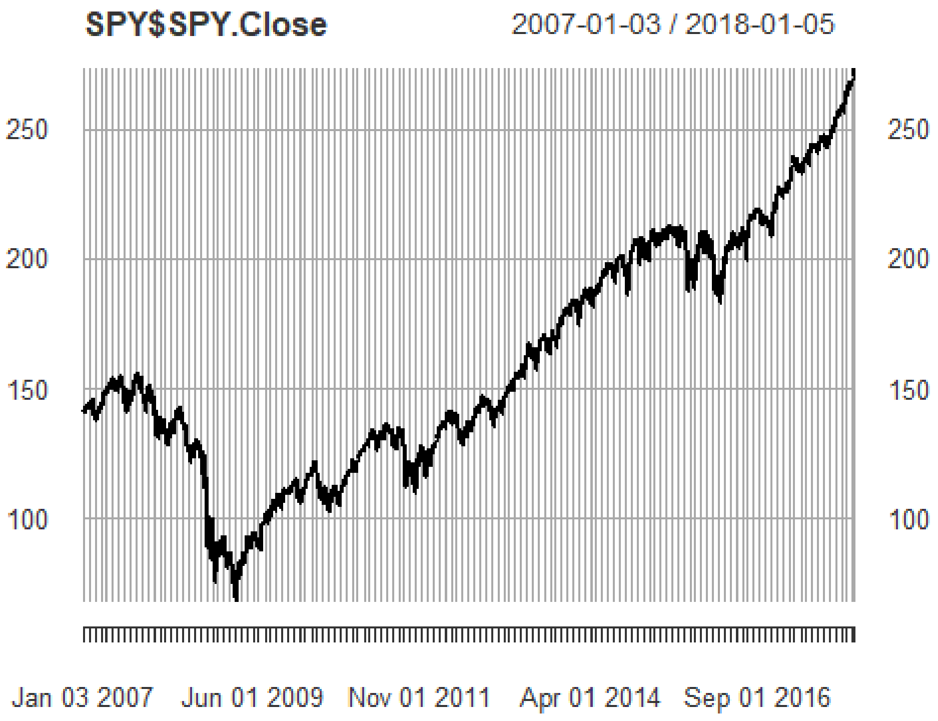
```
points(Short_Trades, col='red', cex=1, pch=18)
```



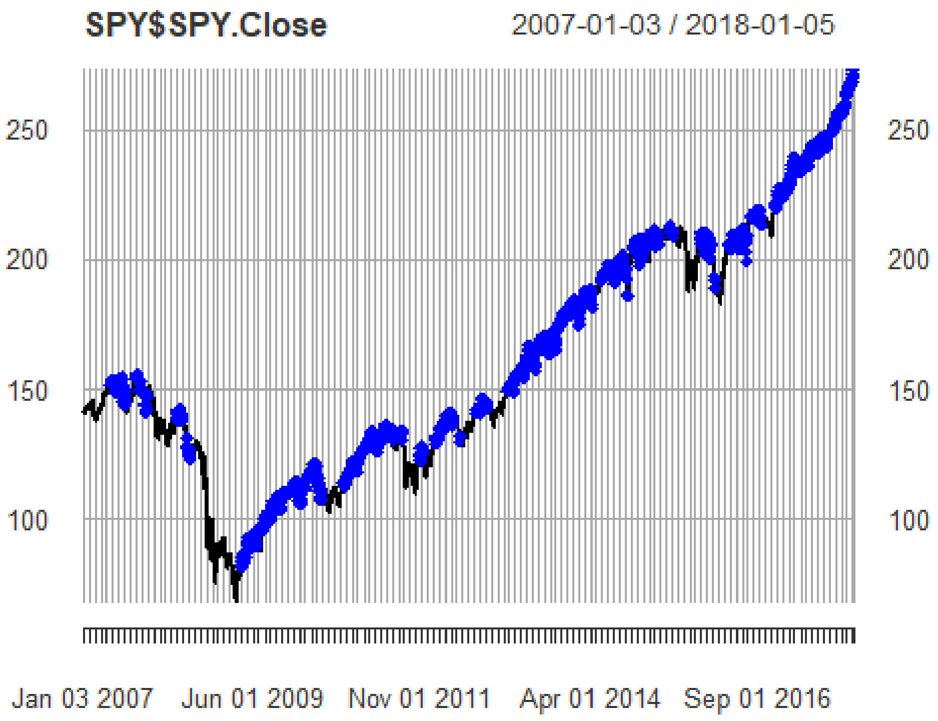
```
chartSeries(SPY, theme="white", TA="addVo();addTA(VWAP.Slow, on=1,
col='red');addTA(VWAP.Fast, on=1, col='blue');addTA(VWAP.Diff, col='blue');
            addADX(n = 14, maType='EMA', wilder=TRUE)")
```



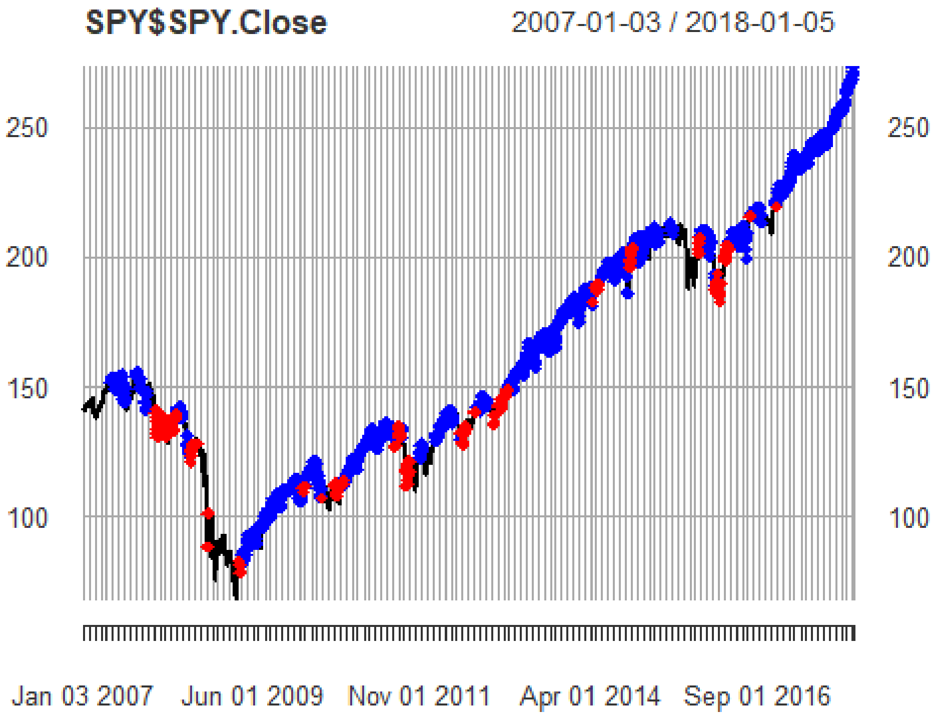
```
ADX.20 <- ADX(SPY,n=14)

# look for long entries
Long_Trades <- ifelse(
        ADX.20$ADX > 20 &
        VWAP.Diff> 0, SPY$SPY.Close, NA)

# look for long entries
Short_Trades <- ifelse(
        ADX.20$ADX > 20 &
        VWAP.Diff < 0, SPY$SPY.Close, NA)

plot(SPY$SPY.Close)
```



```
## Warning in plot.xts(SPY): only the univariate series will be plotted
points(Long_Trades, col='blue', cex=1, pch=18)
```

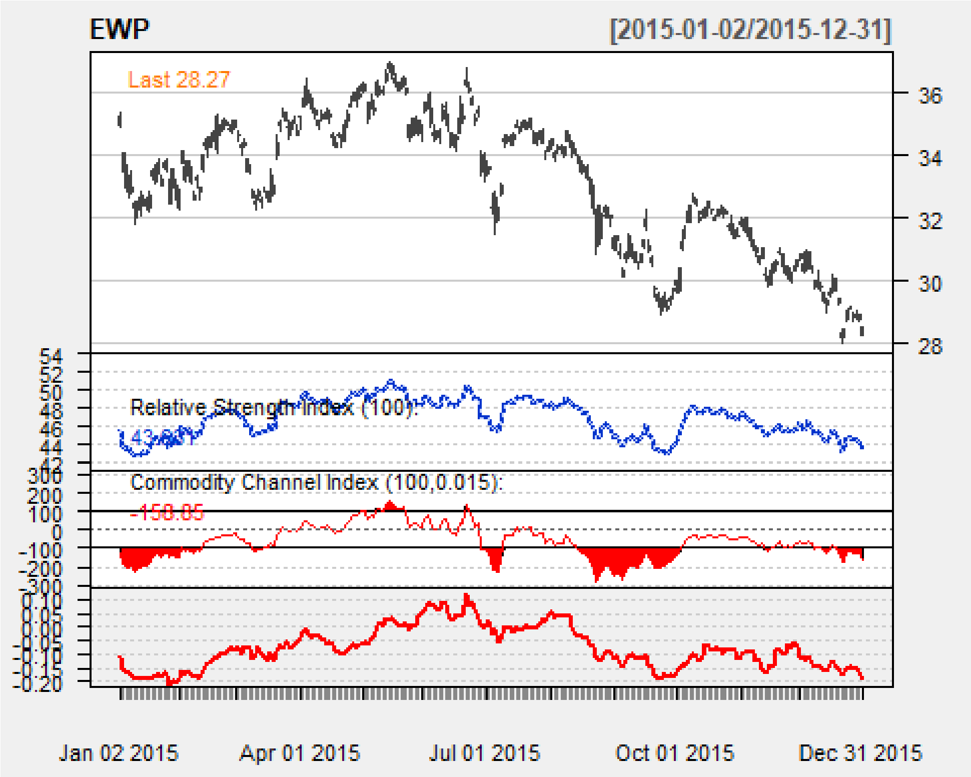


```
points(Short_Trades, col='red', cex=1, pch=18)
```

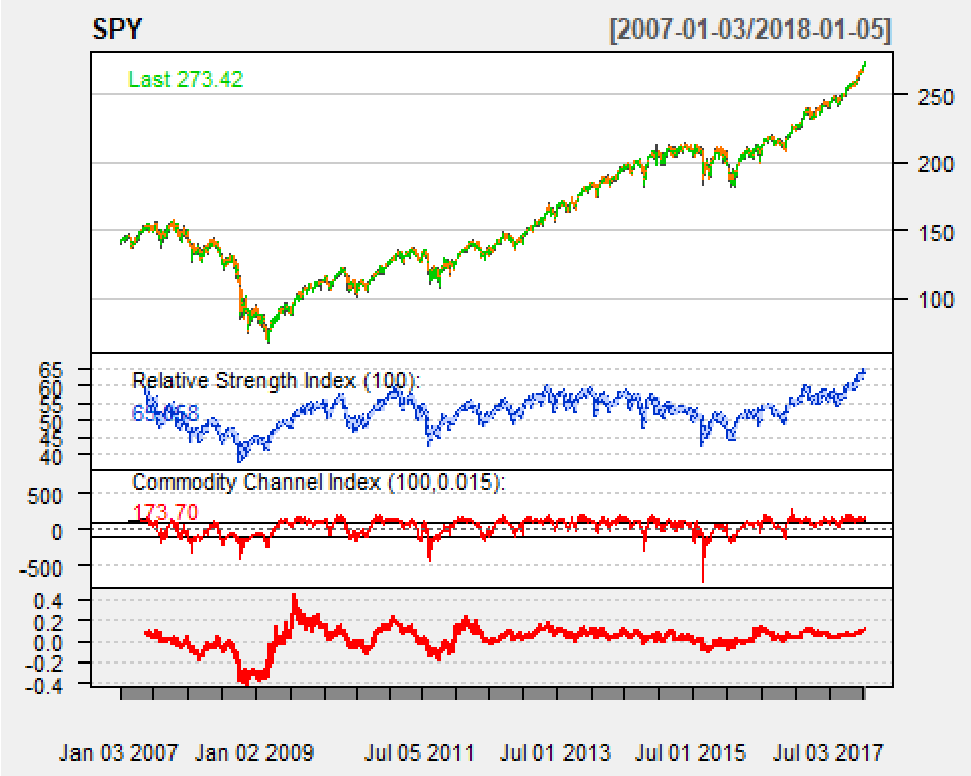


```
print ("STEP 2.7: Counter-Trend Systems including * Momentum Indicators *
Volatility Indicator * Counter-Trend Systems")

## [1] "STEP 2.7: Counter-Trend Systems including * Momentum Indicators *
Volatility Indicator * Counter-Trend Systems"

#Counter-trend systems are tricky. You trade raw counter trends when you're
sure you're in a range-bound market
#and are trading at the extremes otherwise you use added indicators to stay
aligned with longer-term trends.
#Raw counter-trend trading feels like picking tops and bottoms, and those
rarely work out.
#Here we'll focus on trading the short-term counter trend, while following
the long-term trend.


library(binhf)
library(quantmod)
getSymbols(c('EWP', 'SPY'), src='google')

## [1] "EWP" "SPY"

# remove any NAs
EWP <- EWP[!(rowSums(is.na(EWP))),]
```

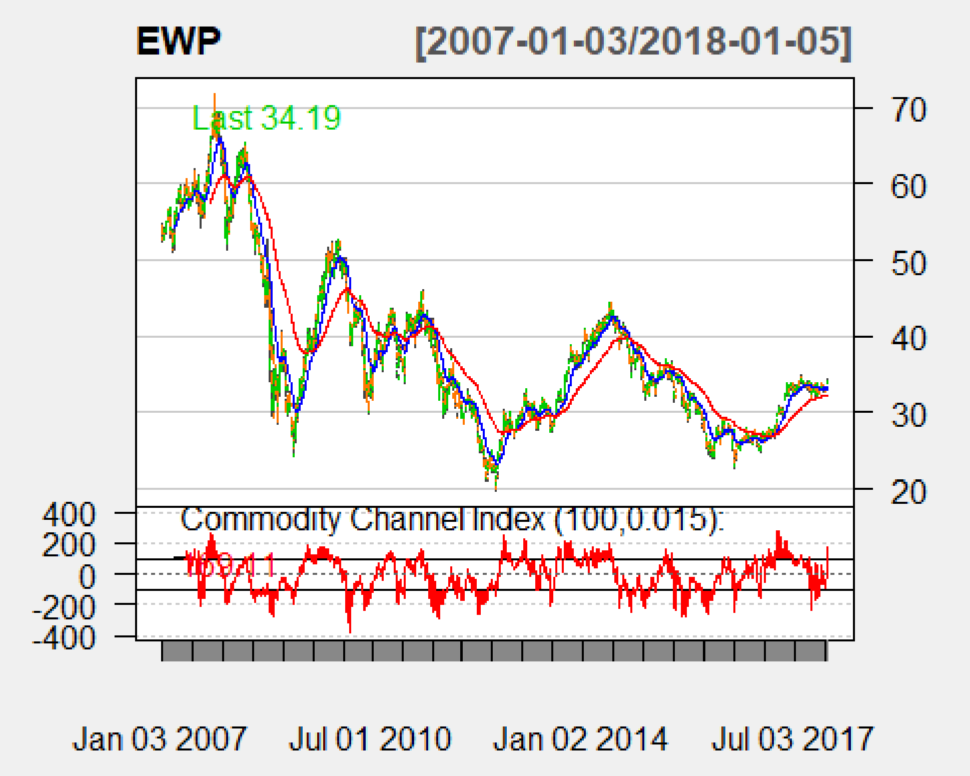
```
SPY <- SPY[!(rowSums(is.na(SPY))),]



#Momentum Indicators

#We’re going to look at 3 interesting momentum indicators that capture short-
term cycles:

#Relative Strength Index (RSI), is an momentum indicator that measures
movement. Its author, J. Welles Wilder, recommends using a period of 14 and
when it is over 70, it is strongly bought (or overbought) and under 30, it is
strongly sold (or oversold).
#REF: https://en.wikipedia.org/wiki/Relative_strength_index
#Commodity Channel Index (CCI) by Donald Lambert, is a price-derived
indicator revolving around 0, where 100 is usually considered overbought and
-100, oversold.
#REF:https://en.wikipedia.org/wiki/Commodity_channel_index
#Rate of Change (ROC), also a momentum indicator, looks at accelerating and
decelerating market moves.
#REF:https://en.wikipedia.org/wiki/Momentum_(technical_analysis)


#Let’s look at all 3 of them with a 20-period setting:

chartSeries(EWP, theme="white",
TA="addRSI(n=100);addCCI(n=100);addROC(n=100)", subset='2015')
```



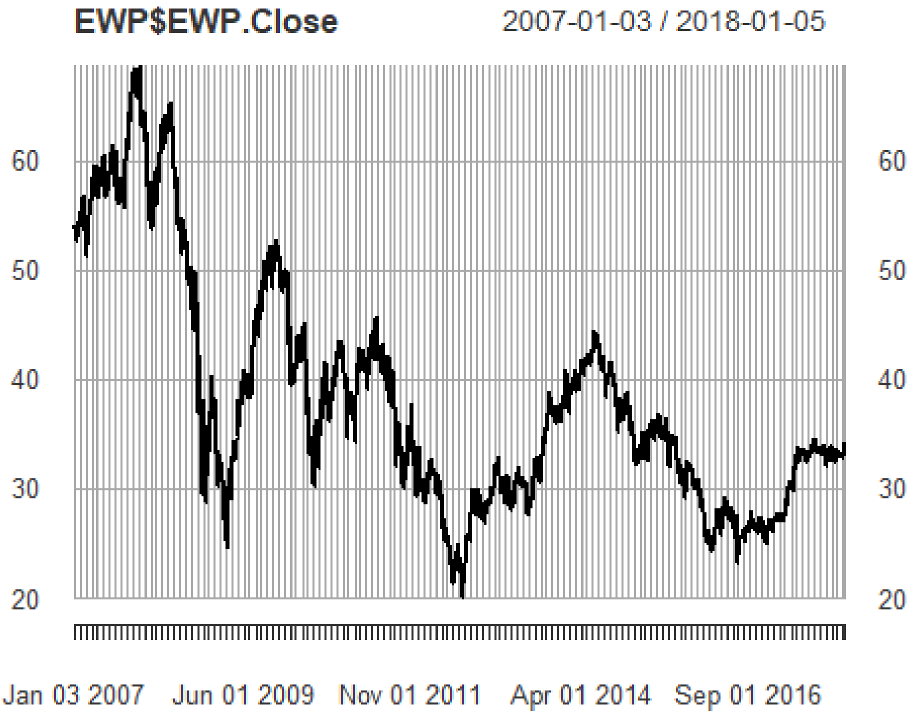
```
chartSeries(SPY, theme="white",
TA="addRSI(n=100);addCCI(n=100);addROC(n=100)")
```



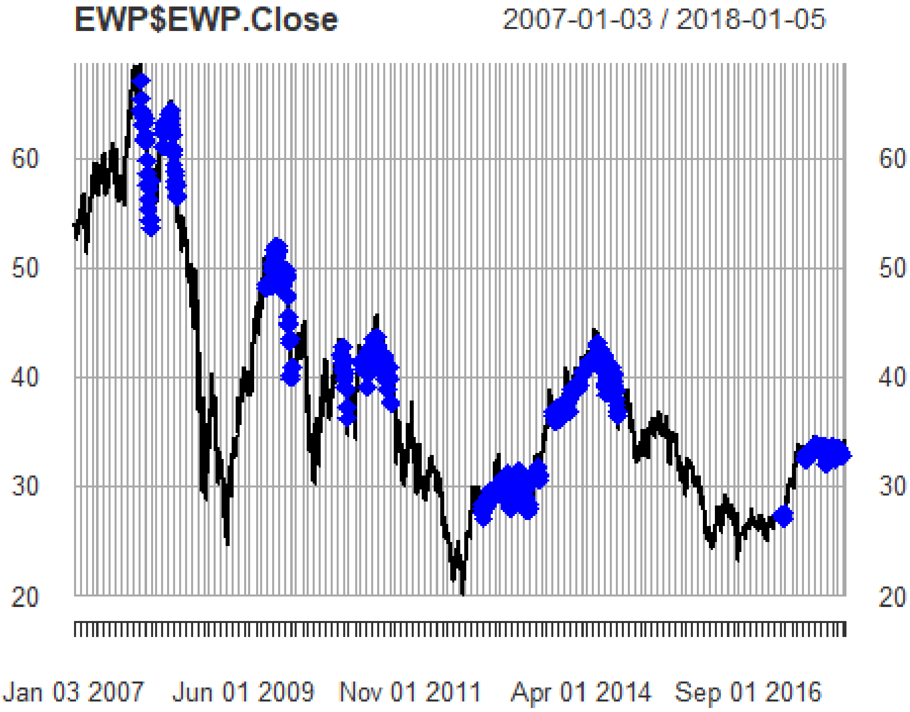
```
#Counter-Trend Systems

#For our counter-trend system, we will counter a faster cycle but stay in the
direction of the slower one. In essence, we’re trading with the slow trend
but against the fast one. While in the previous systems, we only took a trade
while both directions aligned in the direction of the long-term trend.

#The key is to use one of the derived indicators that best signals
overbought/oversold signals.

#We’ll try each one of them with a long-term EMA.

chartSeries(EWP, theme="white",
TA="addCCI(n=100);addEMA(n=50,col='blue');addEMA(n=200,col='red')")
```



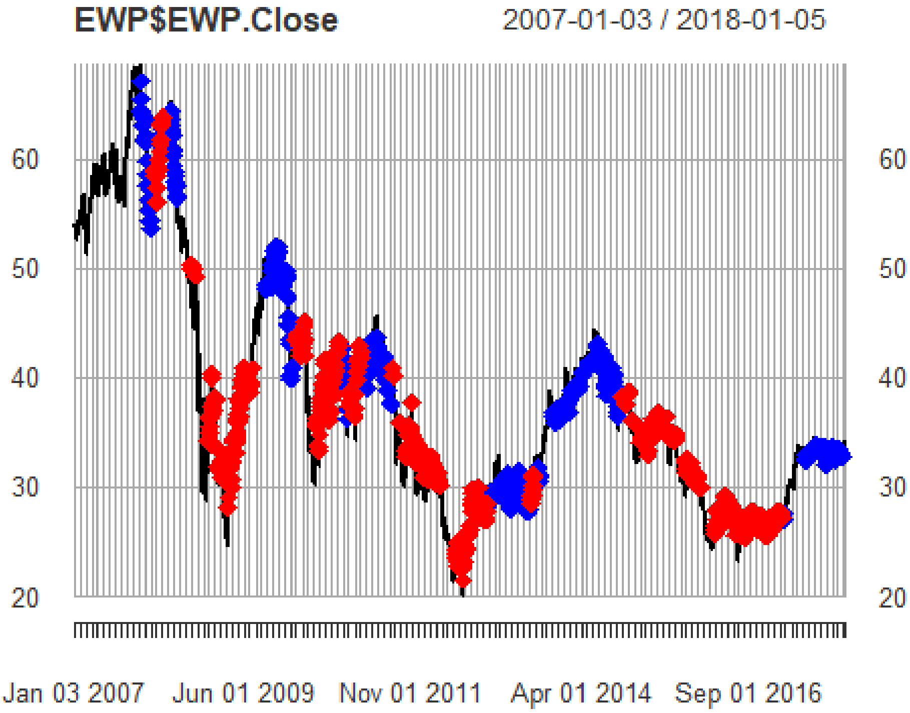
```
# create a slow ema difference
EWP.EMA.50 <- EMA(EWP$EWP.Close, n=50)
EWP.EMA.200 <- EMA(EWP$EWP.Close, n=200)
Slow.Diff <- EWP.EMA.50 - EWP.EMA.200
CCI.IND <- CCI(HLC=EWP[,c("EWP.High","EWP.Low","EWP.Close")],n=100)

# look for long entries
Long_Trades <- ifelse(
  shift(v=as.numeric(CCI.IND), places=1, dir="right") > CCI.IND &
    CCI.IND < 100 &
    Slow.Diff > 0, EWP$EWP.Close, NA)

# look for short entries
Short_Trades <- ifelse(
  shift(v=as.numeric(CCI.IND), places=1, dir="right") < CCI.IND &
    CCI.IND > -100 &
    Slow.Diff < 0, EWP$EWP.Close, NA)

plot(EWP$EWP.Close)
```



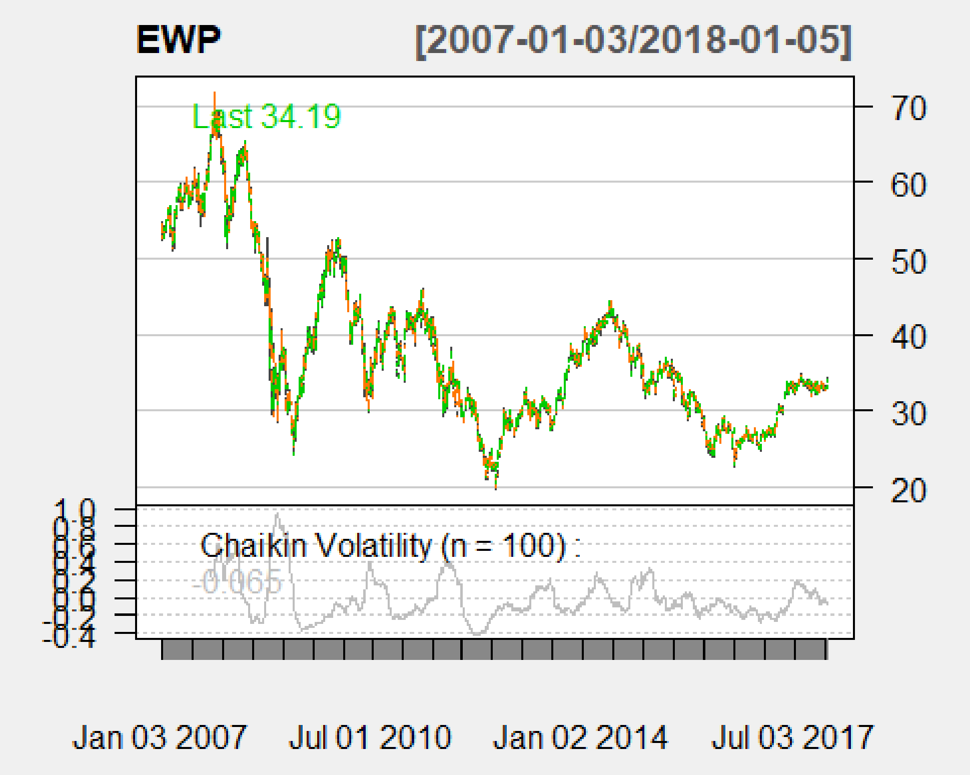
```
## Warning in plot.xts(EWP): only the univariate series will be plotted
points(Long_Trades, col='blue', cex=1.5, pch=18)
```



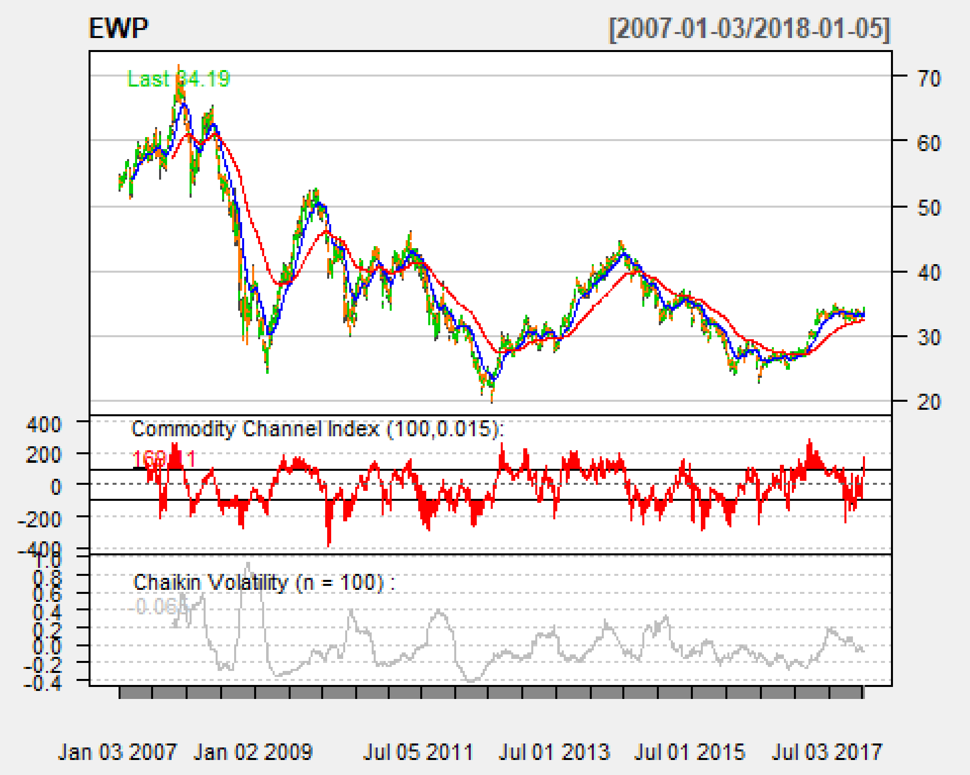
```
points(Short_Trades, col='red', cex=1.5, pch=18)
```



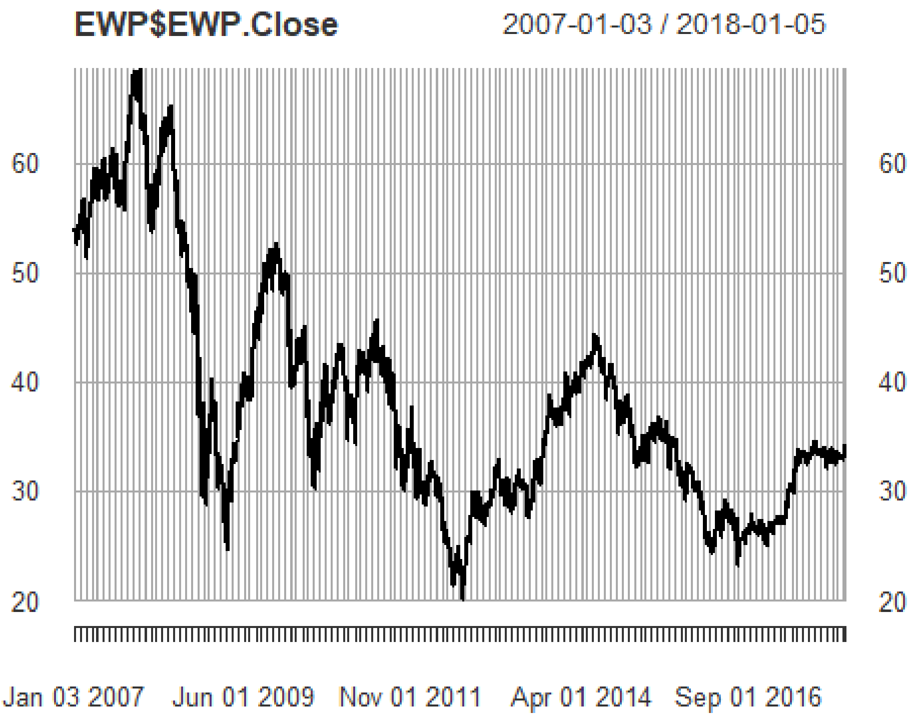
```
#Volatility indicator

#Chaikin Volatility, uses the high, low, close for its
accumulation/distribution and subtracts two moving averages of different
#periods of the AD.

chartSeries(EWP, theme="white", TA="addChVol(n=100);")
```



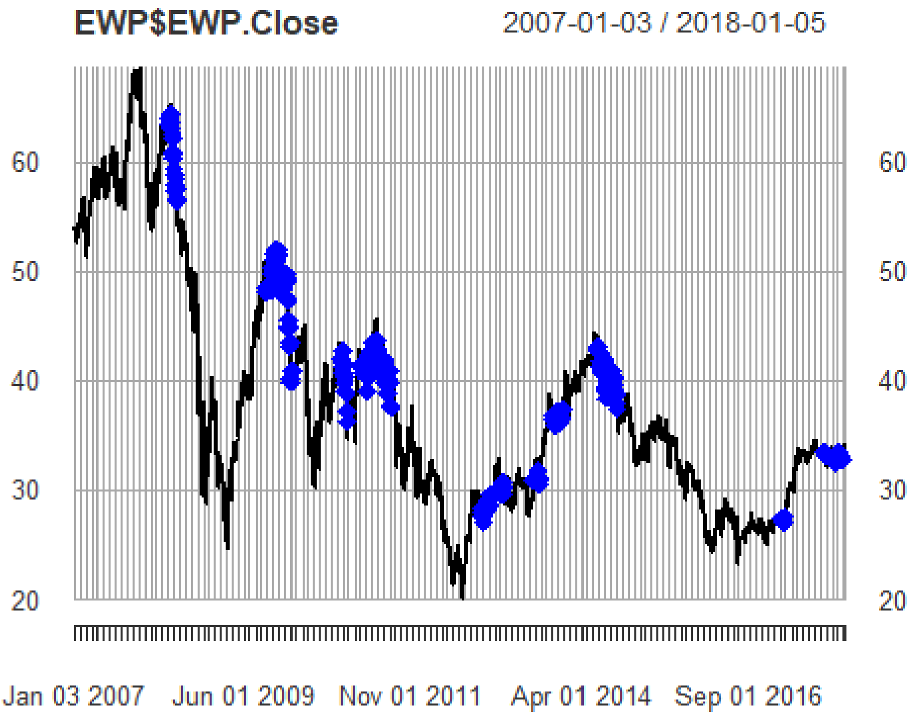
```
chartSeries(EWP, theme="white",
TA="addCCI(n=100);addEMA(n=50,col='blue');addEMA(n=200,col='red');addChVol(n=
100);")
```



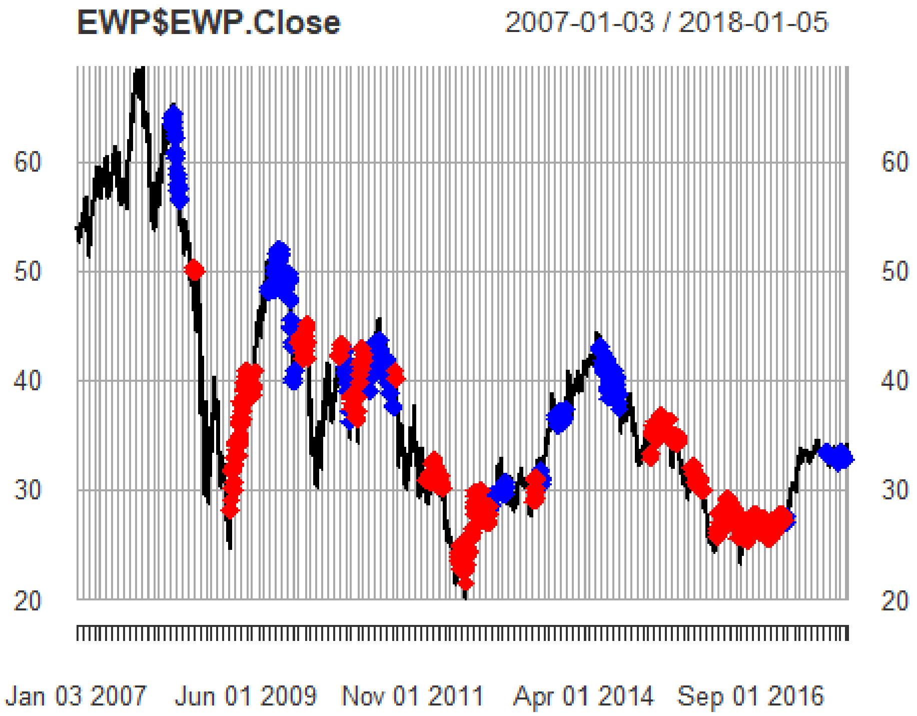
```
# create a slow ema difference
EWP.EMA.50 <- EMA(EWP$EWP.Close, n=50)
EWP.EMA.200 <- EMA(EWP$EWP.Close, n=200)
Slow.Diff <- EWP.EMA.50 - EWP.EMA.200
CCI.IND <- CCI(HLC=EWP[,c("EWP.High","EWP.Low","EWP.Close")],n=100)
CV.IND <- chaikinVolatility(HL=EWP[,c("EWP.High","EWP.Low")], n=100)

# look for long entries
Long_Trades <- ifelse(
     shift(v=as.numeric(CCI.IND), places=1, dir="right") > CCI.IND &
        CCI.IND < 100 &
        CV.IND < 0 &
        Slow.Diff > 0, EWP$EWP.Close, NA)

# look for short entries
Short_Trades <- ifelse(
       shift(v=as.numeric(CCI.IND), places=1, dir="right") < CCI.IND &
        CCI.IND > -100 &
        CV.IND < 0 &
        Slow.Diff < 0, EWP$EWP.Close, NA)

plot(EWP$EWP.Close)
```



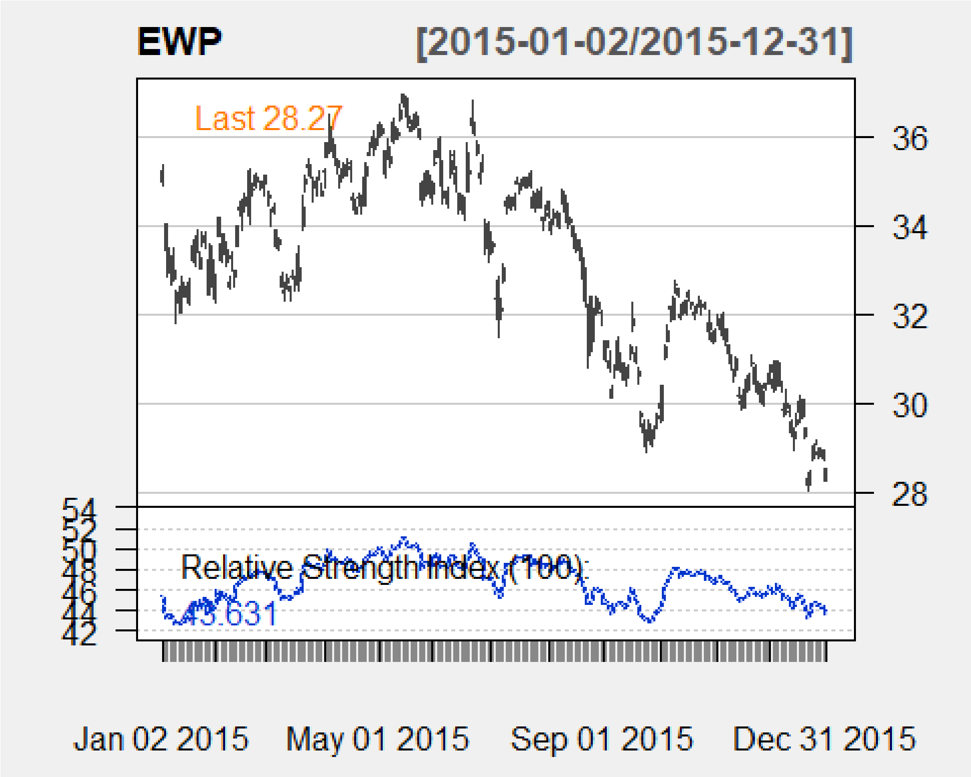
```
## Warning in plot.xts(EWP): only the univariate series will be plotted
points(Long_Trades, col='blue', cex=1.5, pch=18)
```



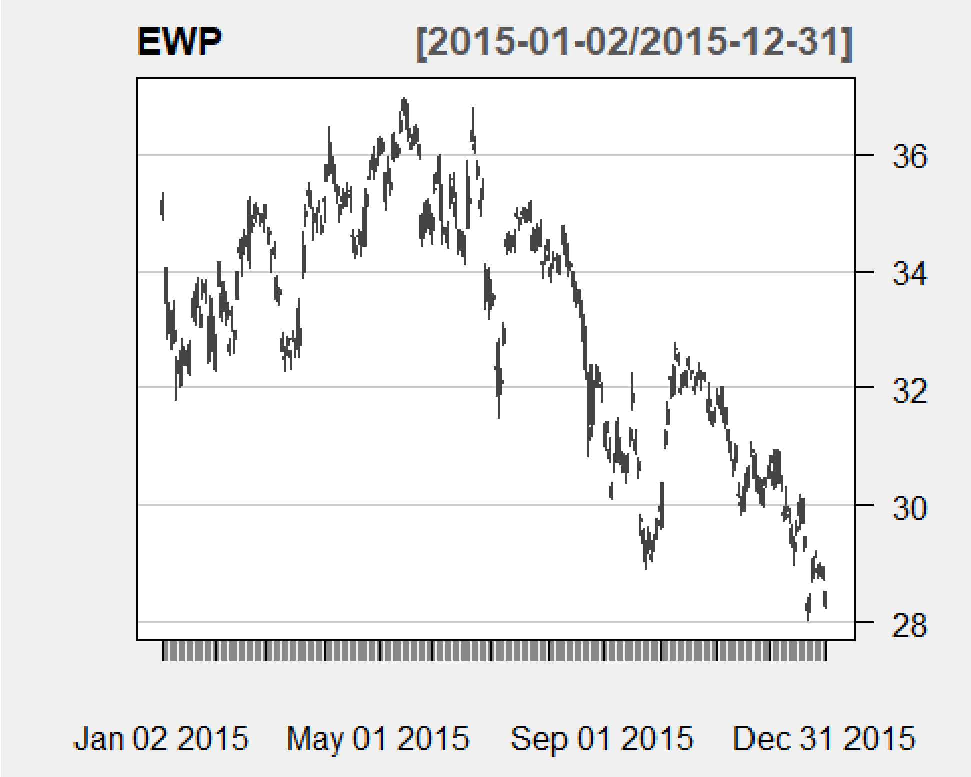
```
points(Short_Trades, col='red', cex=1.5, pch=18)
```



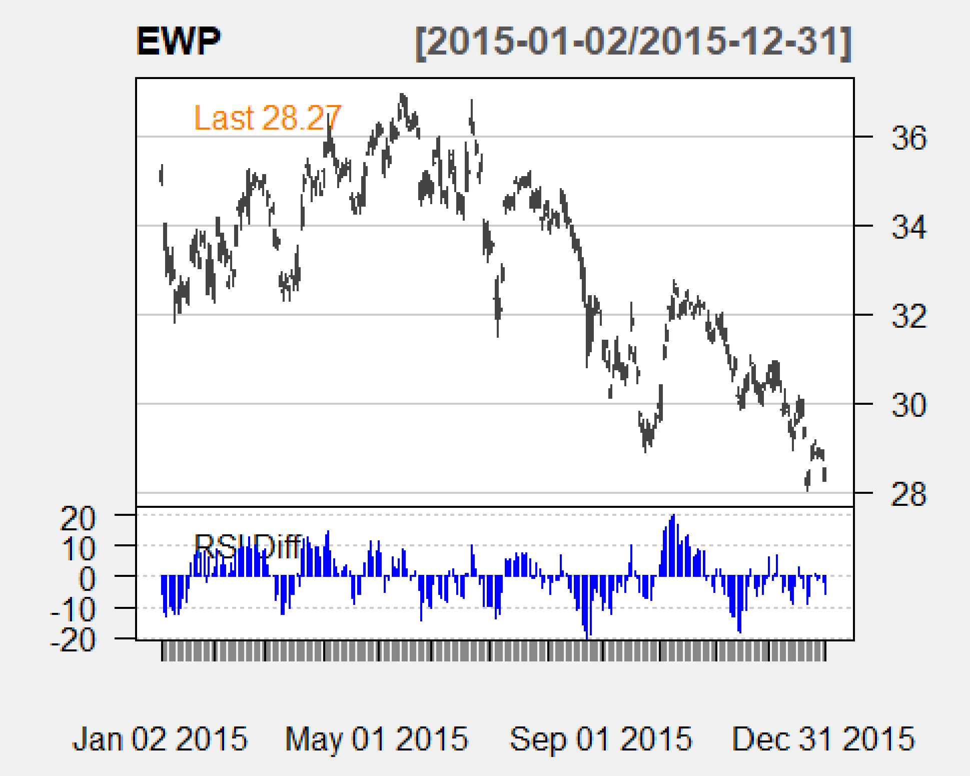
```
#What about shifting further back on the CCI, this ensures that it is a
retracement and not a random bump…

chartSeries(EWP, theme="white", TA="addRSI(n=100);", subset='2015')
```



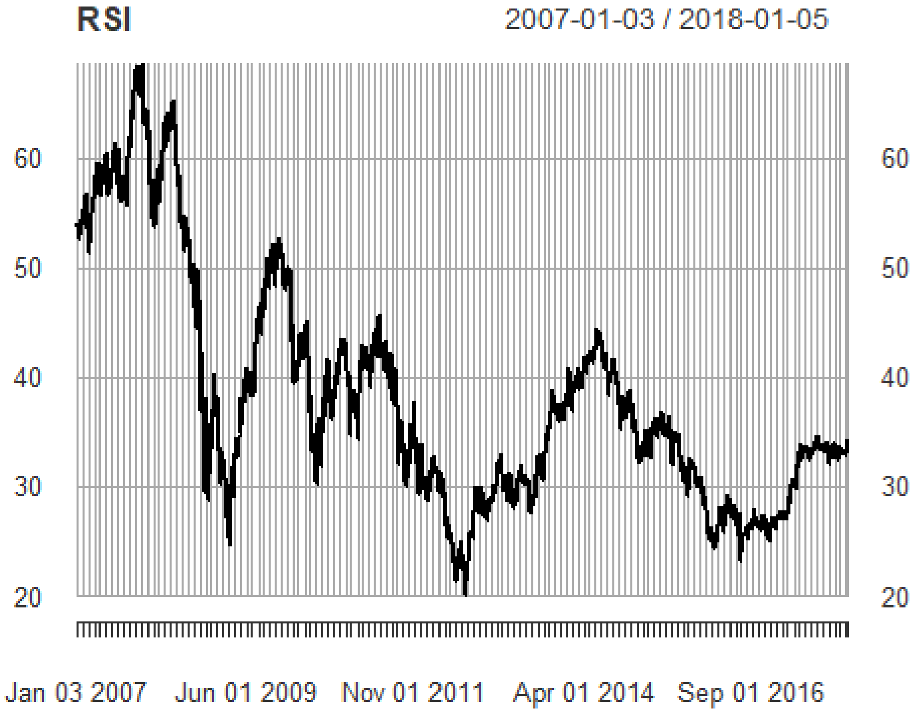
```
chartSeries(EWP, theme="white", TA=NULL, subset='2015')
```



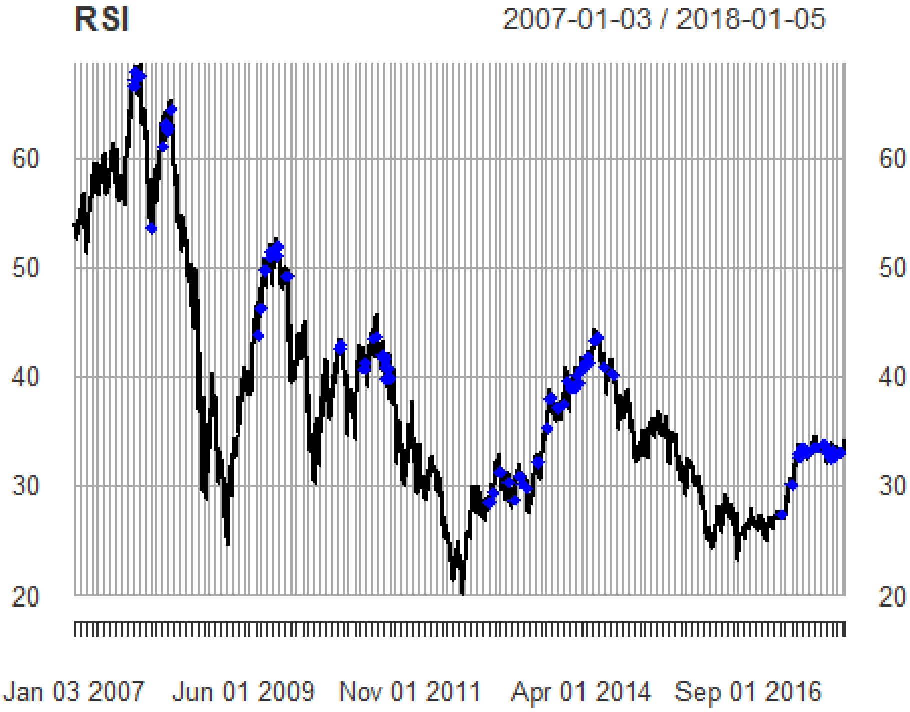
```
RSI.Fast <- RSI(price=EWP$EWP.Close,n=10)
RSI.Slow <- RSI(price=EWP$EWP.Close,n=30)
RSI.Diff <- RSI.Fast-RSI.Slow
addTA(RSI.Diff, col='blue', type='h',legend="RSI Diff")
```



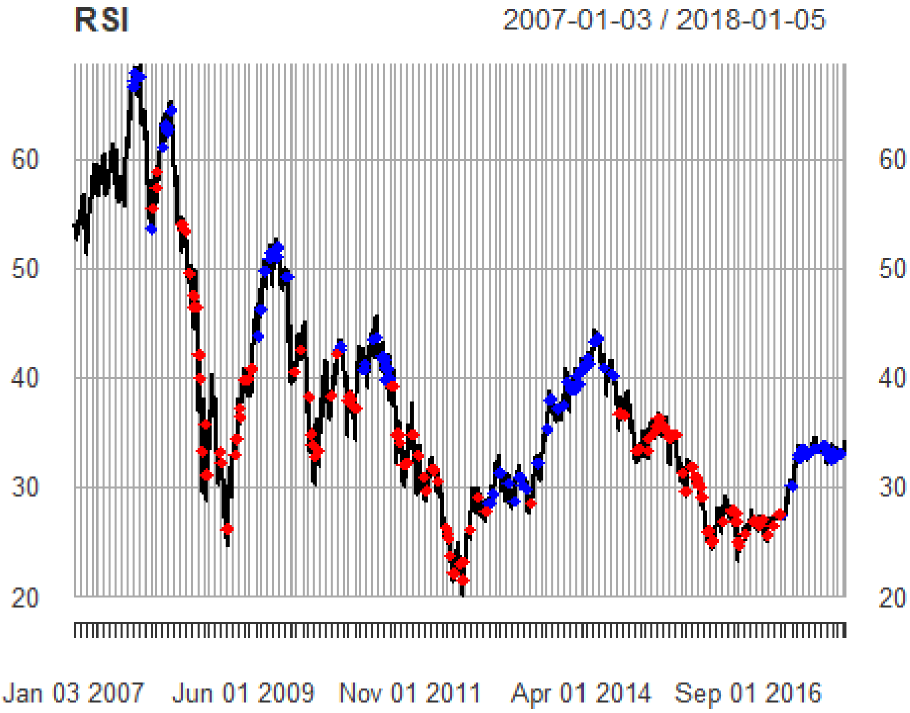
```
# create a slow ema difference
EWP.EMA.50 <- EMA(EWP$EWP.Close, n=50)
EWP.EMA.200 <- EMA(EWP$EWP.Close, n=200)
Slow.Diff <- EWP.EMA.50 - EWP.EMA.200

RSI.IND <- RSI(price=EWP$EWP.Close,n=30)

# look for long entries
Long_Trades <- ifelse(
  RSI.Diff  < 0 &
    shift(v=as.numeric(RSI.Diff ), places=1, dir="right") > 0  &
    Slow.Diff > 0, EWP$EWP.Close, NA)

# look for short entries
Short_Trades <- ifelse(
  RSI.Diff  > 0 &
    shift(v=as.numeric(RSI.Diff ), places=1, dir="right") < 0  &
    Slow.Diff < 0, EWP$EWP.Close, NA)

plot(EWP$EWP.Close, main='RSI')
```



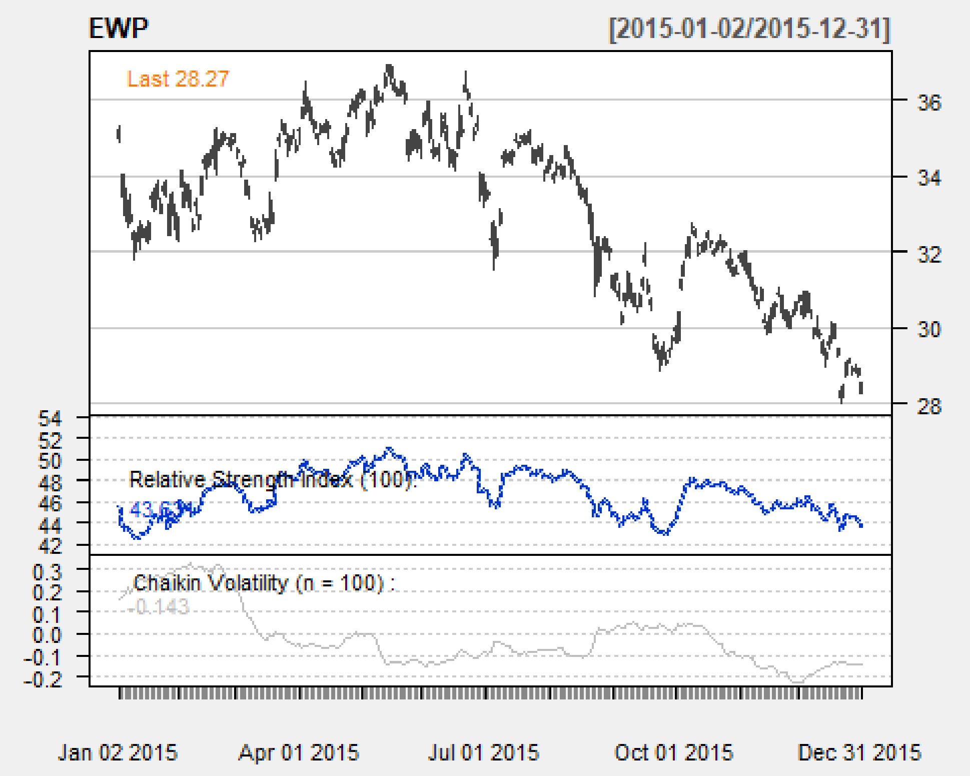
```
## Warning in plot.xts(EWP, main = "RSI"): only the univariate series will be
## plotted
points(Long_Trades, col='blue', cex=1, pch=18)
```



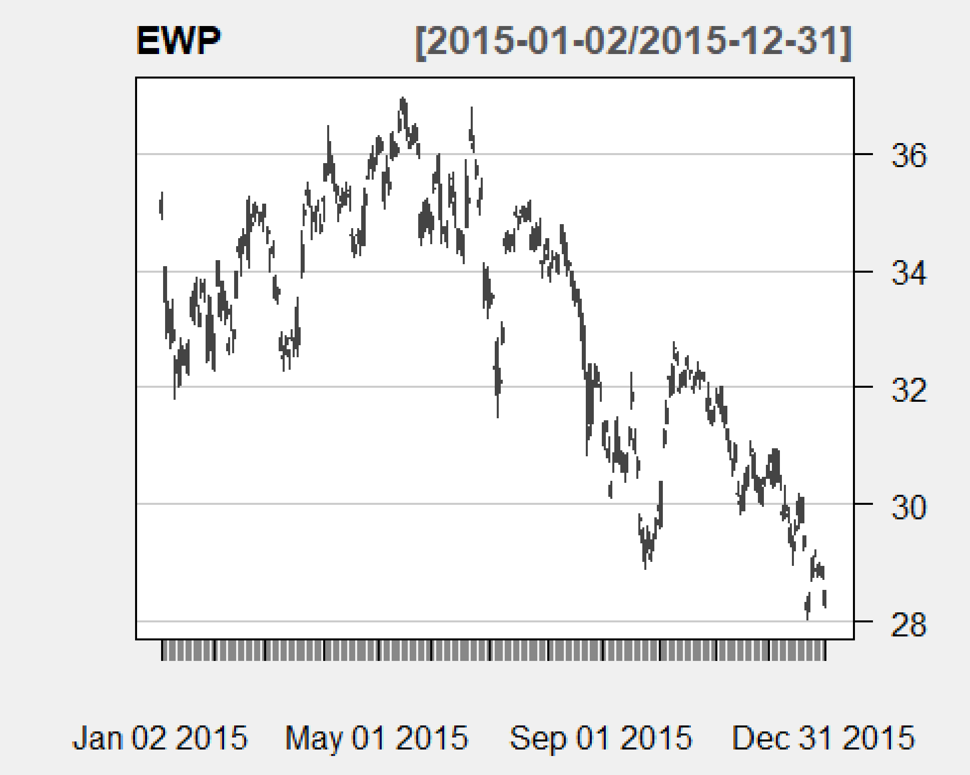
```r
points(Short_Trades, col='red', cex=1, pch=18)
```



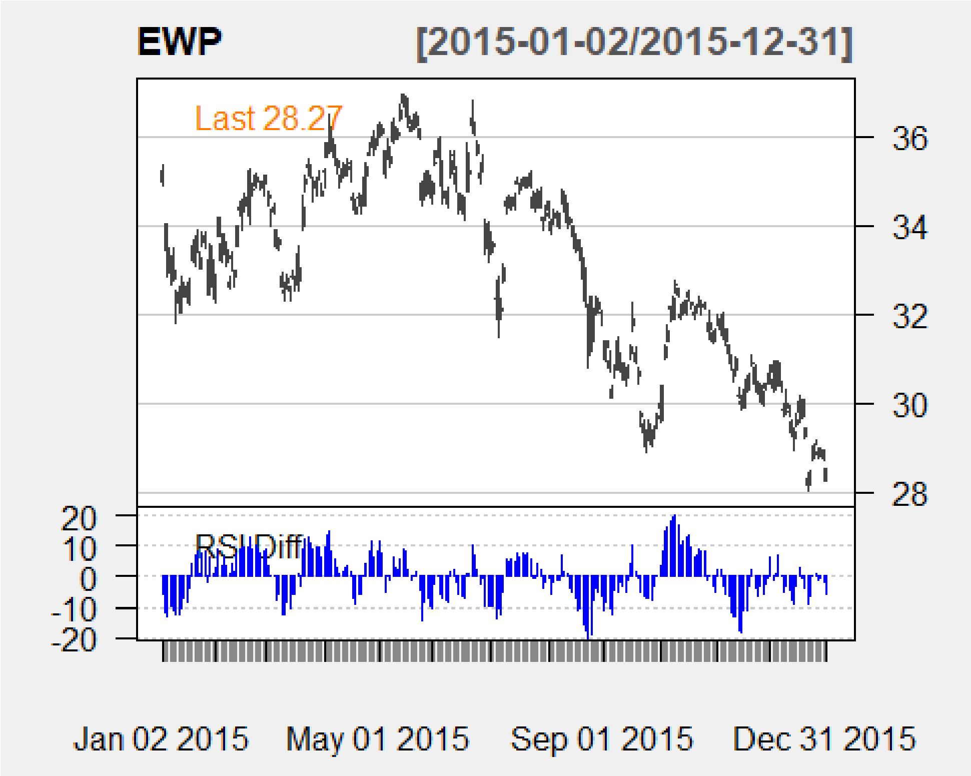
```
#Lets see if we can improve this by adding the Chaikin Volatility to the RSI
like we did earlier with the CCI counter-trading system.

chartSeries(EWP, theme="white", TA="addRSI(n=100);addChVol(n=100);",
subset='2015')
```



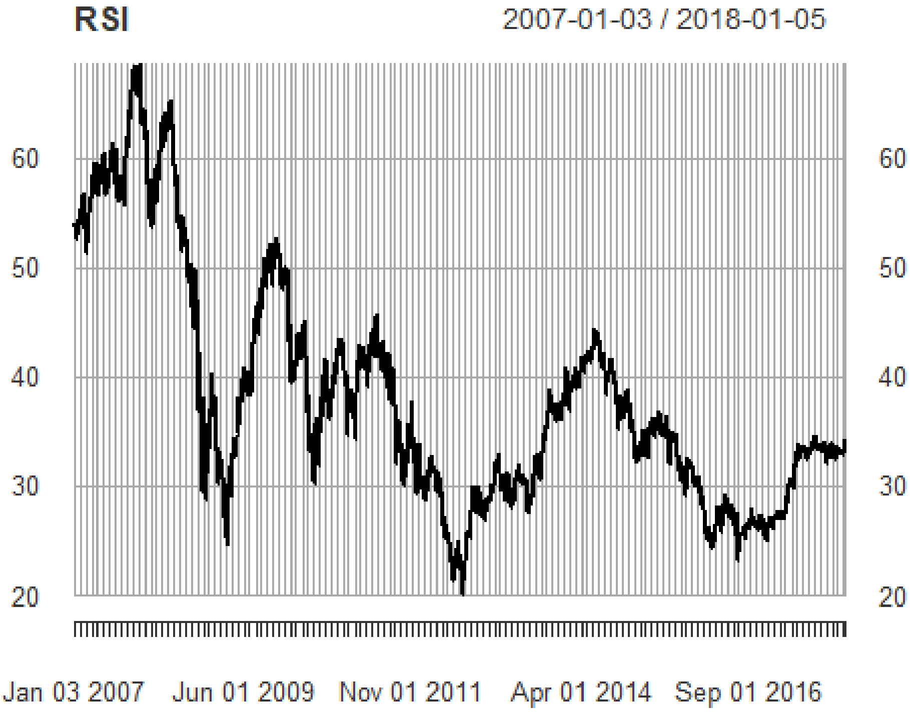
```
chartSeries(EWP, theme="white", TA=NULL, subset='2015')
```



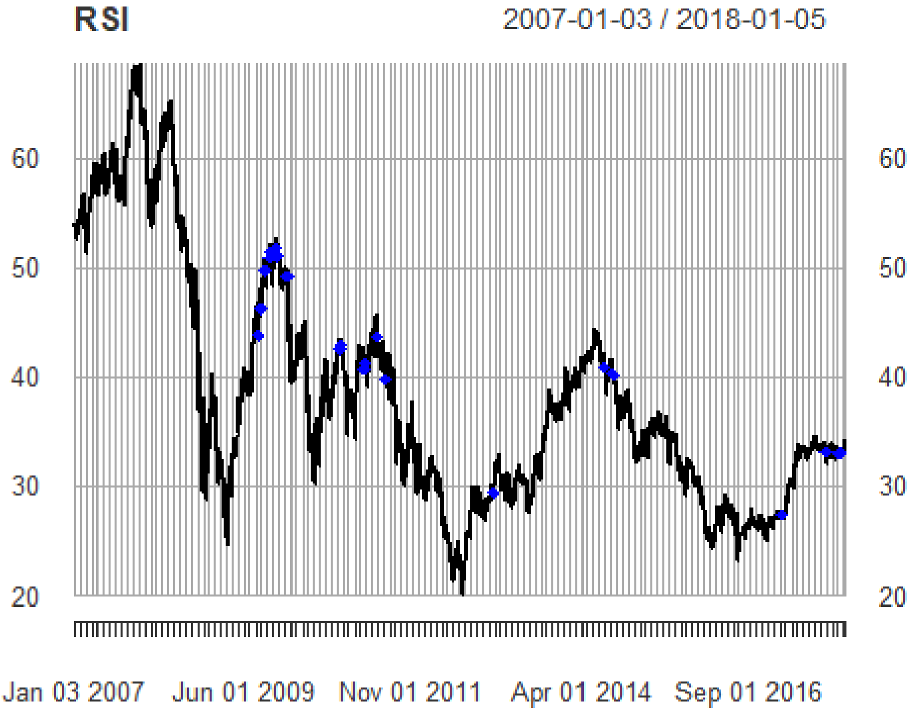
```
RSI.Fast <- RSI(price=EWP$EWP.Close,n=10)
RSI.Slow <- RSI(price=EWP$EWP.Close,n=30)
RSI.Diff <- RSI.Fast-RSI.Slow
addTA(RSI.Diff, col='blue', type='h',legend="RSI Diff")
```



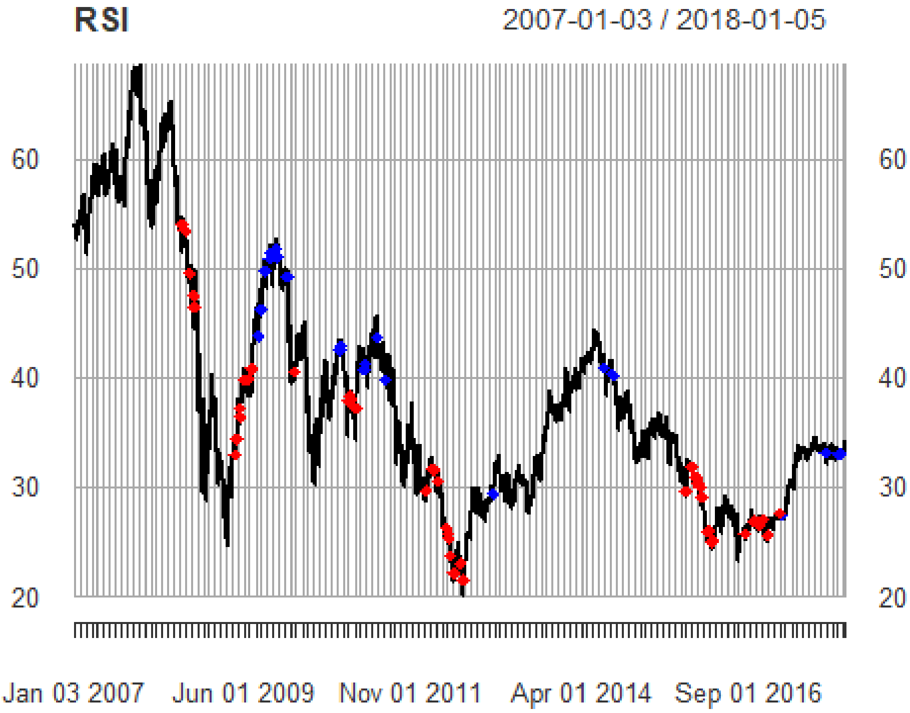
```
# create a slow ema difference
EWP.EMA.50 <- EMA(EWP$EWP.Close, n=50)
EWP.EMA.200 <- EMA(EWP$EWP.Close, n=200)
Slow.Diff <- EWP.EMA.50 - EWP.EMA.200
CV.IND <- chaikinVolatility(HL=EWP, n=100)
RSI.IND <- RSI(price=EWP$EWP.Close,n=30)

# look for long entries
Long_Trades <- ifelse(
  RSI.Diff  < 0 &
    shift(v=as.numeric(RSI.Diff ), places=1, dir="right") > 0  &
    CV.IND < -0.1 &
    Slow.Diff > 0, EWP$EWP.Close, NA)

# look for short entries
Short_Trades <- ifelse(
  RSI.Diff  > 0 &
    shift(v=as.numeric(RSI.Diff ), places=1, dir="right") < 0  &
    CV.IND < -0.1 &
    Slow.Diff < 0, EWP$EWP.Close, NA)

plot(EWP$EWP.Close, main='RSI')
```



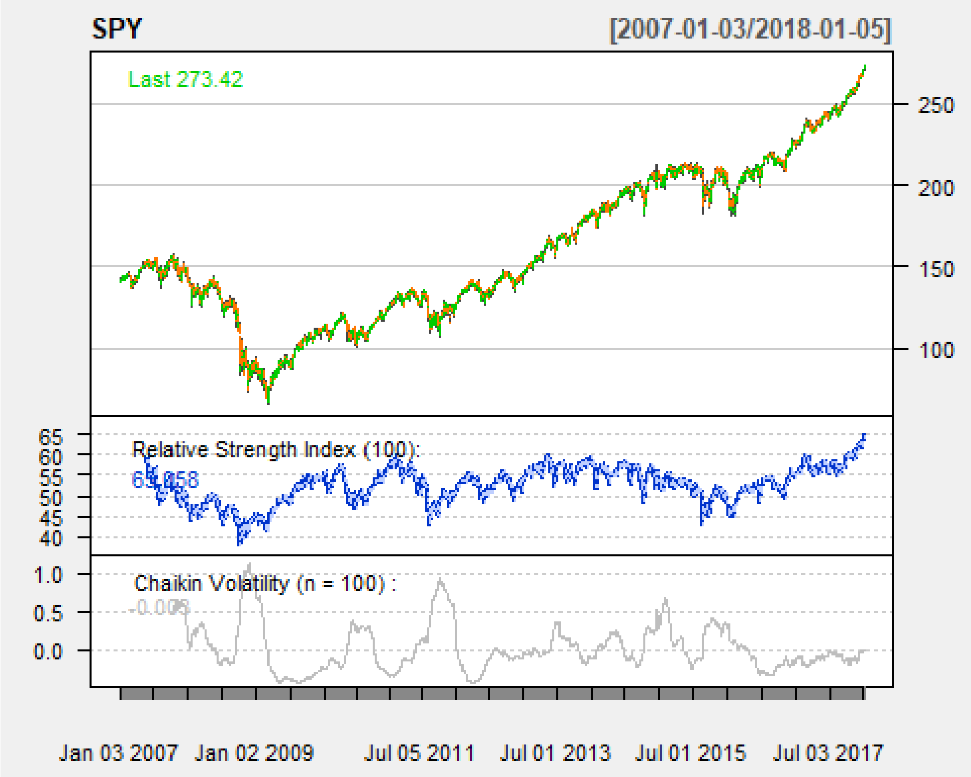
```
## Warning in plot.xts(EWP, main = "RSI"): only the univariate series will be
## plotted
points(Long_Trades, col='blue', cex=1, pch=18)
```



```
points(Short_Trades, col='red', cex=1, pch=18)
```

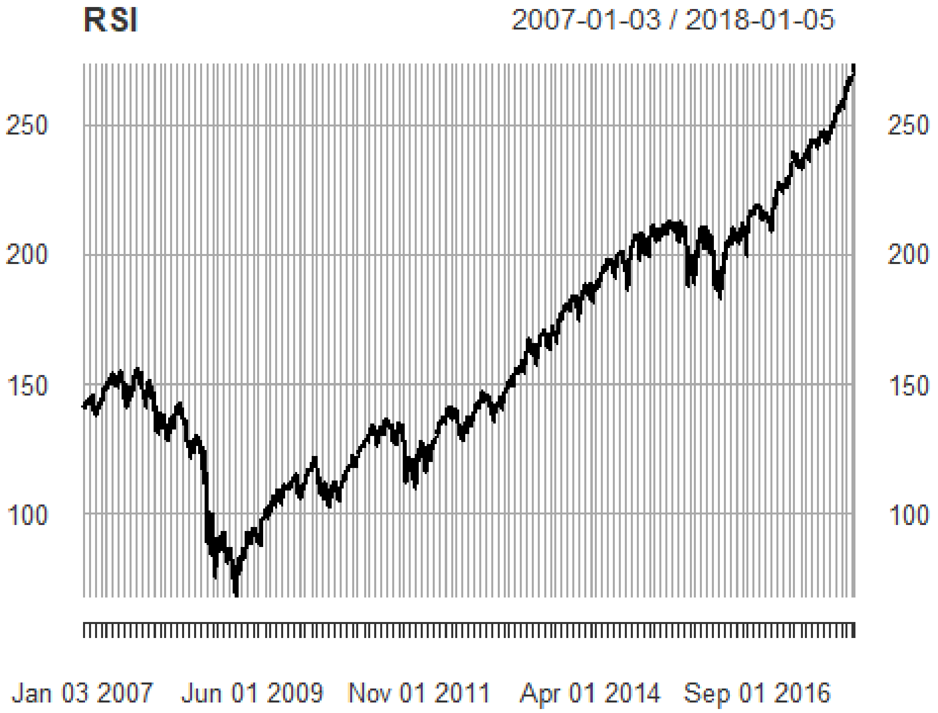


```
#Let’s try this final system on the S&P 500
chartSeries(SPY, theme="white", TA="addRSI(n=100);addChVol(n=100);")
```

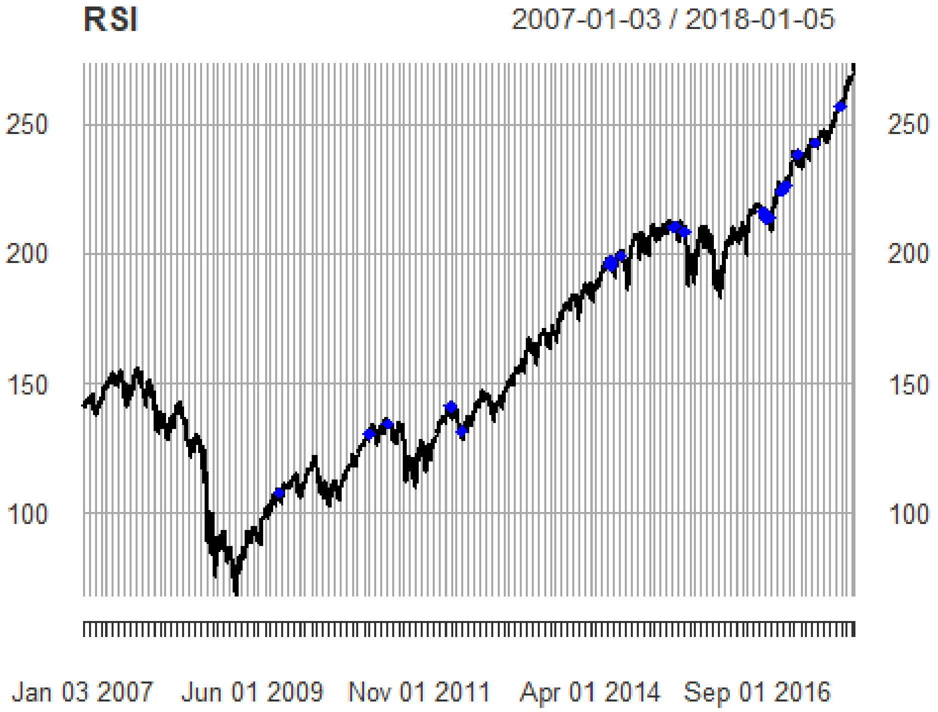


```
# create a slow ema difference
SPY.EMA.50 <- EMA(SPY$SPY.Close, n=50)
SPY.EMA.200 <- EMA(SPY$SPY.Close, n=200)
Slow.Diff <- SPY.EMA.50 - SPY.EMA.200

RSI.Fast <- RSI(price=SPY$SPY.Close,n=10)
RSI.Slow <- RSI(price=SPY$SPY.Close,n=30)
RSI.Diff <- RSI.Fast-RSI.Slow

CV.IND <- chaikinVolatility(HL=SPY, n=100)

# look for long entries
Long_Trades <- ifelse(
  CV.IND < -0.1 &
    RSI.Diff  < 0 &
    shift(v=as.numeric(RSI.Diff ), places=1, dir="right") > 0  &
    shift(v=as.numeric(RSI.Diff ), places=2, dir="right") < 0  &
    Slow.Diff > 0, SPY$SPY.Close, NA)

# look for short entries
Short_Trades <- ifelse(
  CV.IND < -0.1 &
```

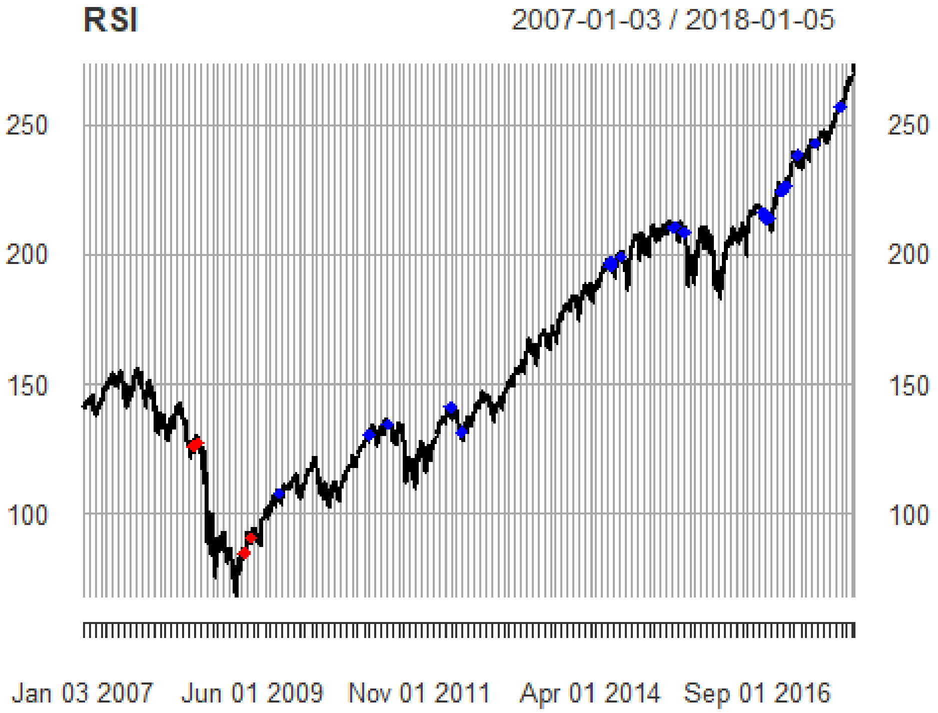
```
    RSI.Diff  > 0 &
    shift(v=as.numeric(RSI.Diff ), places=1, dir="right") < 0  &
    shift(v=as.numeric(RSI.Diff ), places=2, dir="right") > 0  &
    Slow.Diff < 0, SPY$SPY.Close, NA)

plot(SPY$SPY.Close, main='RSI')
```



```
## Warning in plot.xts(SPY, main = "RSI"): only the univariate series will be
## plotted
points(Long_Trades, col='blue', cex=1, pch=18)
```

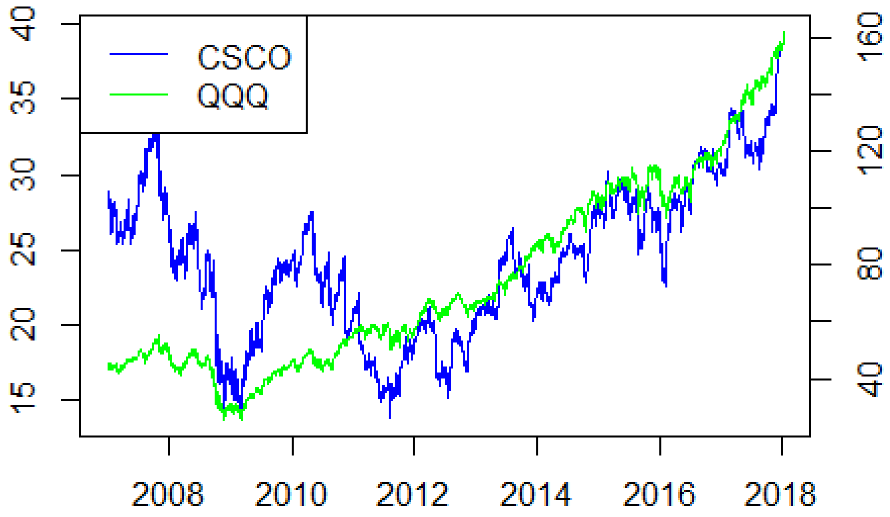


```
points(Short_Trades, col='red', cex=1, pch=18)
```

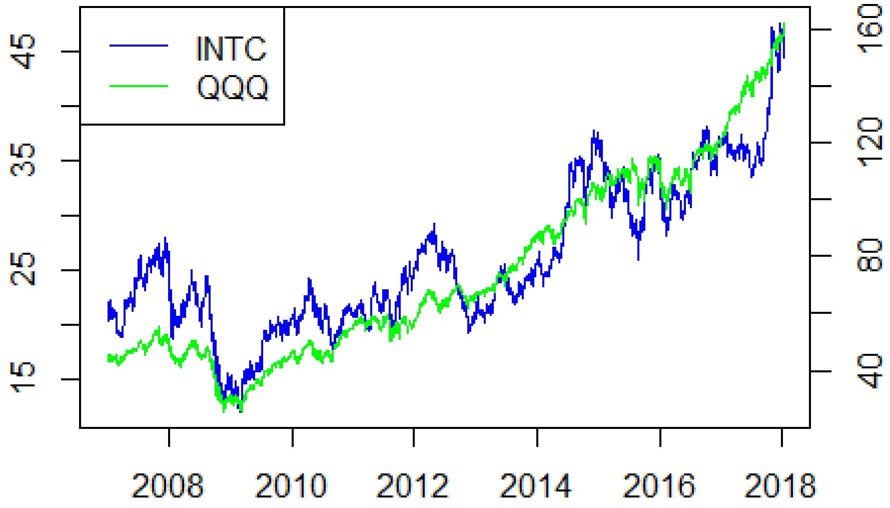


```
# Basket Analysis
#Basket of stocks related to the QQQ

#We'll use a few member stocks of the QQQ Index. This makes things easy for
us, but the concepts discussed here can be
#applied to any other financial product and index as long they are related in
some way.

#We'll focus on the following tech stocks:

#CSCO, INTC, MSFT, YHOO, TXN. They're fairly related, of similar size, and we
can donwload 10+ years of data for each.

print ("STEP 2.8: Basket of stocks related to the QQQ Index")

## [1] "STEP 2.8: Basket of stocks related to the QQQ Index"

library(quantmod)
basket_symbols <- c('MSFT', 'INTC', 'YHOO', 'CSCO', 'TXN', 'QQQ')
getSymbols(basket_symbols, src='google')

## [1] "MSFT" "INTC" "YHOO" "CSCO" "TXN"  "QQQ"

#We need to merge all the stocks into one data.frame. We'll use as.xts that
converts objects to xts class,
#this will merge by time all our columns into one data frame:
basket <- data.frame(as.xts(merge(MSFT, INTC, YHOO, CSCO, TXN, QQQ)))
head(basket,2)

##            MSFT.Open MSFT.High MSFT.Low MSFT.Close MSFT.Volume INTC.Open
## 2007-01-03     29.91     30.25    29.40      29.86    77574283     20.45
## 2007-01-04     29.70     29.97    29.44      29.81    46120855     20.63
##            INTC.High INTC.Low INTC.Close INTC.Volume YHOO.Open YHOO.High
## 2007-01-03     20.88    20.14      20.35    69803965     25.85     26.26
## 2007-01-04     21.33    20.56      21.17    89514297     25.64     26.92
##            YHOO.Low YHOO.Close YHOO.Volume CSCO.Open CSCO.High CSCO.Low
## 2007-01-03    25.26      25.61    26654067     27.46     27.98    27.33
## 2007-01-04    25.52      26.85    32565729     27.68     28.49    27.54
##            CSCO.Close CSCO.Volume TXN.Open TXN.High TXN.Low TXN.Close
## 2007-01-03      27.73    64882632    29.12    29.22   28.35     28.56
## 2007-01-04      28.46    73336624    28.50    29.11   28.41     29.10
##            TXN.Volume QQQ.Open QQQ.High QQQ.Low QQQ.Close QQQ.Volume
## 2007-01-03   20650100    43.46    44.06   42.52     43.24  168787533
## 2007-01-04   20117000    43.30    44.21   43.15     44.06  137380464

#To keep things simple, we'll only keep the Close column for all symbols:
basket <- basket[,names(basket)[grepl(x=names(basket), pattern='Close')]]
head(basket)
```

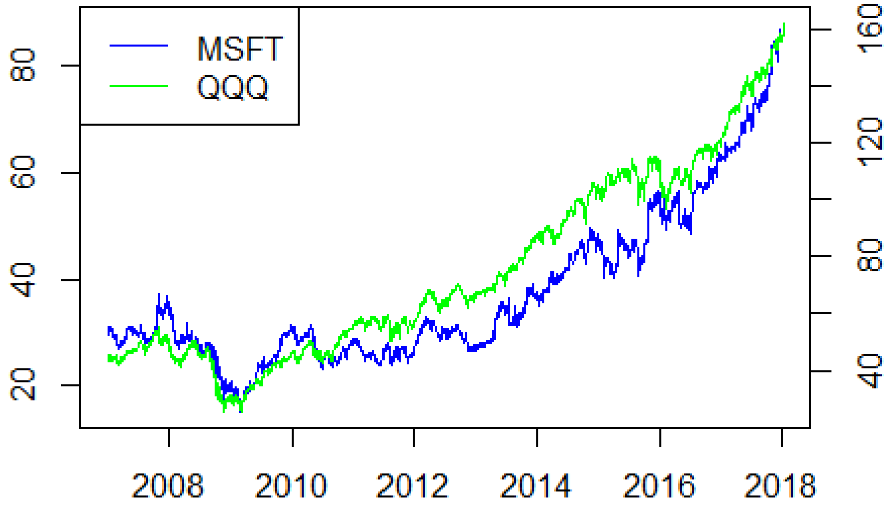
```
##            MSFT.Close INTC.Close YHOO.Close CSCO.Close TXN.Close QQQ.Close
## 2007-01-03      29.86      20.35      25.61      27.73     28.56     43.24
## 2007-01-04      29.81      21.17      26.85      28.46     29.10     44.06
## 2007-01-05      29.64      21.10      27.74      28.47     28.76     43.85
## 2007-01-08      29.93      21.01      27.92      28.63     28.90     43.88
## 2007-01-09      29.96      21.03      27.58      28.47     28.84     44.10
## 2007-01-10      29.66      21.52      28.70      28.68     29.33     44.62

#Let's pair every stock with the QQQ in a chart. We'll overlay them together,
and, even though they won't share the same price scale,
#it should still give us an idea of how they both move:
plot(as.Date(row.names(basket)), basket$CSCO.Close, col="blue", type='l',
ylab="", xlab="")
par(new=TRUE)
plot(as.Date(row.names(basket)), basket$QQQ.Close, col='green', type='l',
     xaxt="n", yaxt='n', xlab="",ylab="")
axis(4)
legend("topleft",col=c("blue","green"),lty=1,legend=c("CSCO","QQQ"))
```



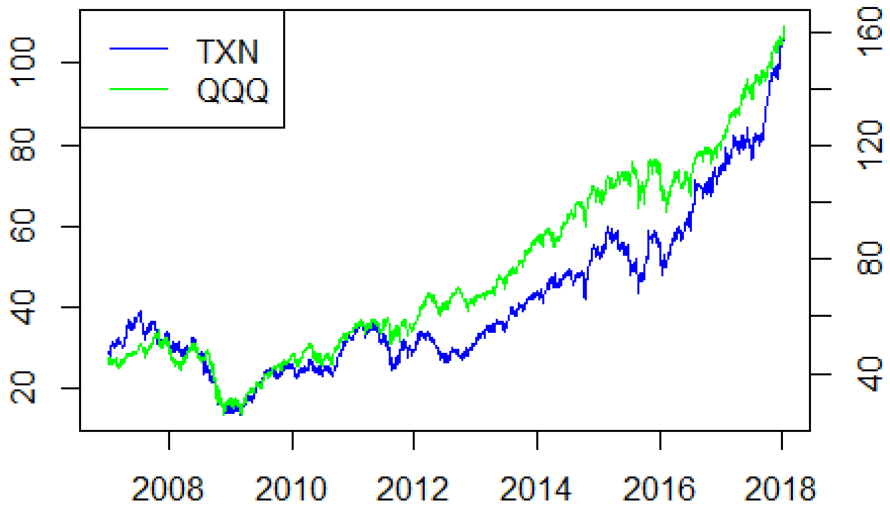
```
plot(as.Date(row.names(basket)), basket$INTC.Close, col="blue", type='l',
ylab="", xlab="")
par(new=TRUE)
plot(as.Date(row.names(basket)), basket$QQQ.Close, col='green', type='l',
     xaxt="n", yaxt='n', xlab="",ylab="")
axis(4)
legend("topleft",col=c("blue","green"),lty=1,legend=c("INTC","QQQ"))
```



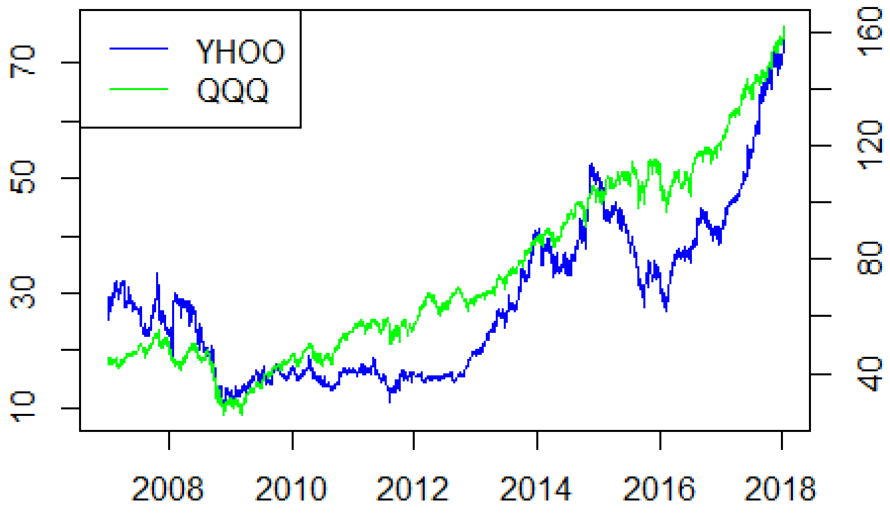
```
plot(as.Date(row.names(basket)), basket$MSFT.Close, col="blue", type='l',
ylab="", xlab="")
par(new=TRUE)
plot(as.Date(row.names(basket)), basket$QQQ.Close, col='green', type='l',
     xaxt="n", yaxt='n', xlab="",ylab="")
axis(4)
legend("topleft",col=c("blue","green"),lty=1,legend=c("MSFT","QQQ"))
```



```r
plot(as.Date(row.names(basket)), basket$TXN.Close, col="blue", type='l',
ylab="", xlab="")
par(new=TRUE)
plot(as.Date(row.names(basket)), basket$QQQ.Close, col='green', type='l',
     xaxt="n", yaxt='n', xlab="",ylab="")
axis(4)
legend("topleft",col=c("blue","green"),lty=1,legend=c("TXN","QQQ"))
```



```
plot(as.Date(row.names(basket)), basket$YHOO.Close, col='blue', type='l',
ylab="", xlab="")
par(new=TRUE)
plot(as.Date(row.names(basket)), basket$QQQ.Close, col='green', type='l',
     xaxt="n", yaxt='n', xlab="",ylab="")
axis(4)
legend("topleft",col=c("blue","green"),lty=1,legend=c("YHOO","QQQ"))
```



```
#All the stocks in our basket have followed the QQQ relatively well with the
exception of CISCO.
#The point here, is that there may be arbitrage opportunities with stocks
that deviate from their group or index but
#it's important to be cautious. Stocks deviate from their peers for a reason
and may want to investigate before jumping in -
#whether its just a perception or a serious change.

#Looking at direction

#There is a handy function in quantmod called OHLC.Transformations.
#This allows you to quickly tranform and compare time-series data.
#We'll use the ClCl function that will calculate the difference between the
current and previous close.
#We will use the difference between closes to determine if it is an up or
down day bar
#(if yesterday's close is lower than today's, then its an up day).

movement_MSFT <- ifelse(ClCl(MSFT)[-1] > 0, 1, -1)
movement_QQQ <- ifelse(ClCl(QQQ)[-1] > 0, 1, -1)
# use a table to see what matched and what didn't
table(movement_MSFT, movement_QQQ)

##                movement_QQQ
## movement_MSFT    -1     1
##            -1  956  418
##            1   295 1102

# Or a simpler way:
sum(movement_MSFT == movement_QQQ) / length(movement_QQQ)

## [1] 0.7426922

#The resulting table matrix tells us that out of the 2167 trading days
recorded,
#they both had the same down days 762 times and the same up days 843 times.
They basically were in sync 74% of the time.

#Let's compare our other symbols:

movement_INTC <- ifelse(ClCl(INTC)[-1] > 0, 1, -1)
sum(movement_INTC[-1] == movement_QQQ) / length(movement_QQQ)

## [1] 0.737279

movement_YHOO <- ifelse(ClCl(YHOO)[-1] > 0, 1, -1)
sum(movement_YHOO[-1] == movement_QQQ[-1]) / length(movement_QQQ)

## [1] 0.6997474

movement_CSCO <- ifelse(ClCl(CSCO)[-1] > 0, 1, -1)
sum(movement_CSCO == movement_QQQ[-1]) / length(movement_QQQ)

## [1] 0.7390834

movement_TXN <- ifelse(ClCl(TXN)[-1] > 0, 1, -1)
sum(movement_TXN == movement_QQQ[-1]) / length(movement_QQQ)

## [1] 0.7416095

print ("STEP 2.9:Basket Analysis * Overall correlation * Time-split
correlations")

## [1] "STEP 2.9:Basket Analysis * Overall correlation * Time-split
correlations"

library(quantmod)
basket_symbols <- c('MSFT', 'INTC', 'YHOO', 'CSCO', 'TXN', 'QQQ')
getSymbols(basket_symbols, src='google')

## [1] "MSFT" "INTC" "YHOO" "CSCO" "TXN"  "QQQ"

basket <- data.frame(as.xts(merge(MSFT, INTC, YHOO, CSCO, TXN, QQQ)))
basket <- basket[,names(basket)[grepl(x=names(basket), pattern='Close')]]

#Overall correlation
```

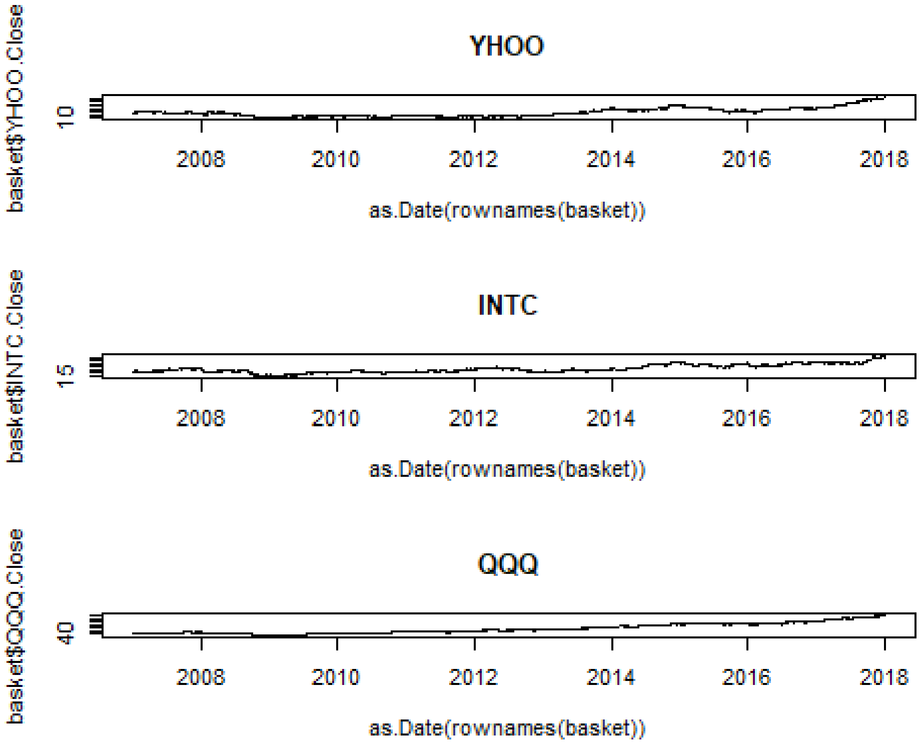
```
#So, how correlated are our stocks in our basket? Let's find out.
#We'll use the base cor function in R. It basically compares two vectors
applying covariances and standard deviations
# Look at the last column, this shows the QQQ's correlation to each stock:
results <- c()
for (basket_name in names(basket)) {
       result <- round(as.numeric(cor(basket)[,basket_name]),2)
       results <- rbind(results, c(basket_name,result))
}
results <- data.frame(results)
names(results)[-1] <- names(basket)
results

##           X1 MSFT.Close INTC.Close YHOO.Close CSCO.Close TXN.Close
## 1 MSFT.Close          1       0.92       0.91       0.79      <NA>
## 2 INTC.Close       0.92          1       0.86       0.75      <NA>
## 3 YHOO.Close       0.91       0.86          1        0.8      <NA>
## 4 CSCO.Close       0.79       0.75        0.8          1      <NA>
## 5  TXN.Close       <NA>       <NA>       <NA>       <NA>         1
## 6  QQQ.Close       0.96       0.93       0.89       0.69      <NA>
##   QQQ.Close
## 1      0.96
## 2      0.93
## 3      0.89
## 4      0.69
## 5      <NA>
## 6         1

#Time-split correlations

#Let's dig deeper and build a function to generelaize the process of getting
a correlation table.
#With this function in hand, we will split the data by time and compare
different time periods

# time for a correlation function
Get_Column_Correlations <- function(objDF){
        results <- c()
        for (col_name in names(objDF)) {
               result <- round(as.numeric(cor(objDF)[,col_name]),2)
               results <- rbind(results, c(col_name,result))
        }
        results <- data.frame(results)
        names(results)[-1] <- names(objDF)
        return (results)
}
Get_Column_Correlations(basket[as.Date(rownames(basket)) < '2015-01-
01',])[,c('X1','QQQ.Close')]
```

```
##            X1 QQQ.Close
## 1 MSFT.Close      0.88
## 2 INTC.Close      0.82
## 3 YHOO.Close      0.73
## 4 CSCO.Close      0.15
## 5  TXN.Close      <NA>
## 6  QQQ.Close         1
```

```
Get_Column_Correlations(basket[as.Date(rownames(basket)) >= '2015-01-
01',])[,c('X1','QQQ.Close')]
```

```
##            X1 QQQ.Close
## 1 MSFT.Close      0.96
## 2 INTC.Close       0.8
## 3 YHOO.Close       0.9
## 4 CSCO.Close      0.88
## 5  TXN.Close      0.95
## 6  QQQ.Close         1
```

```
par(mfrow=c(3,1))
plot(as.Date(rownames(basket)), basket$YHOO.Close, type='l', col='black',
main='YHOO')
plot(as.Date(rownames(basket)), basket$INTC.Close, type='l', col='black',
main='INTC')
plot(as.Date(rownames(basket)), basket$QQQ.Close, type='l', col='black',
main='QQQ')
```

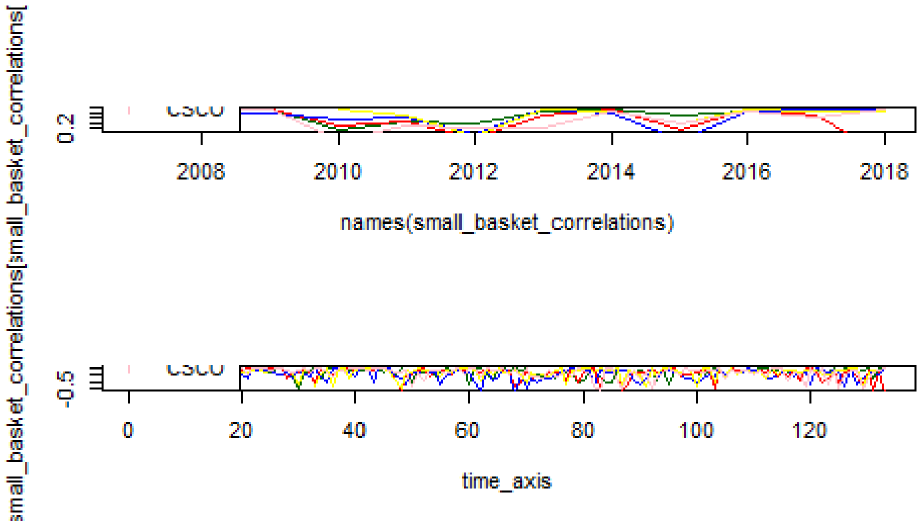


```
#Let's look at all of these by year and analyze correlations with the QQQ:
basket_years <- unique(substr(rownames(basket), start=1, stop=4))
small_basket <- basket
MSFT_QQQ <- c()
INTC_QQQ <- c()
YHOO_QQQ <- c()
TXN_QQQ <- c()
CSCO_QQQ <- c()
for (year in basket_years) {
        print(year)
        temp_df <- small_basket[substr(rownames(basket), start=1,
stop=4)==year,]
        MSFT_QQQ <- cbind(MSFT_QQQ, cor(temp_df$MSFT.Close,
temp_df$QQQ.Close))
        INTC_QQQ <- cbind(INTC_QQQ, cor(temp_df$INTC.Close,
temp_df$QQQ.Close))
        YHOO_QQQ <- cbind(YHOO_QQQ, cor(temp_df$YHOO.Close,
temp_df$QQQ.Close))
        TXN_QQQ <- cbind(TXN_QQQ, cor(temp_df$TXN.Close, temp_df$QQQ.Close))
        CSCO_QQQ <- cbind(CSCO_QQQ, cor(temp_df$CSCO.Close,
temp_df$QQQ.Close))
}
```

```
## [1] "2007"
## [1] "2008"
## [1] "2009"
## [1] "2010"
## [1] "2011"
## [1] "2012"
## [1] "2013"
## [1] "2014"
## [1] "2015"
## [1] "2016"
## [1] "2017"
## [1] "2018"
```

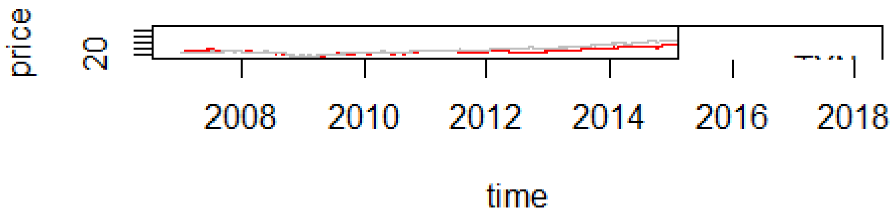
```
small_basket_correlations <- data.frame(rbind(MSFT_QQQ, INTC_QQQ, YHOO_QQQ,
TXN_QQQ, CSCO_QQQ))
colnames(small_basket_correlations) <- basket_years
plot(names(small_basket_correlations), small_basket_correlations[1,],
type='l', col='darkgreen')
lines(names(small_basket_correlations), small_basket_correlations[2,],
type='l', col='red')
lines(names(small_basket_correlations), small_basket_correlations[3,],
type='l', col='blue')
lines(names(small_basket_correlations), small_basket_correlations[4,],
type='l', col='yellow')
lines(names(small_basket_correlations), small_basket_correlations[5,],
type='l', col='pink')
```

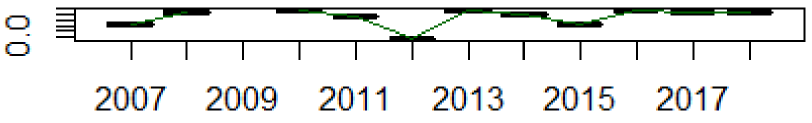
```
legend(x='bottomleft', legend=c("MSFT", "INTC", "YHOO", "TXN", "CSCO"),
col=c("darkgreen","red", "blue", "yellow", "pink"), lwd=1, lty=c(0,0),
        pch=c(3,3))


#This is very revealing how the correlation of both stocks with the index
waxes and wanes. Let's visualize these results.

basket_months <- unique(substr(rownames(basket), start=1, stop=7))
small_basket <- basket #[,names(basket)[grepl(x=names(basket),
pattern='MSFT|INTC|QQQ')]]
MSFT_QQQ <- c()
INTC_QQQ <- c()
YHOO_QQQ <- c()
TXN_QQQ <- c()
CSCO_QQQ <- c()
for (yearmonth in basket_months) {
        temp_df <- small_basket[substr(rownames(basket), start=1,
stop=7)==yearmonth,]
        MSFT_QQQ <- cbind(MSFT_QQQ, cor(temp_df$MSFT.Close,
temp_df$QQQ.Close))
        INTC_QQQ <- cbind(INTC_QQQ, cor(temp_df$INTC.Close,
temp_df$QQQ.Close))
        YHOO_QQQ <- cbind(YHOO_QQQ, cor(temp_df$YHOO.Close,
temp_df$QQQ.Close))
        TXN_QQQ <- cbind(TXN_QQQ, cor(temp_df$TXN.Close, temp_df$QQQ.Close))
        CSCO_QQQ <- cbind(CSCO_QQQ, cor(temp_df$CSCO.Close,
temp_df$QQQ.Close))
}

small_basket_correlations <- data.frame(rbind(MSFT_QQQ, INTC_QQQ, YHOO_QQQ,
TXN_QQQ, CSCO_QQQ))
time_axis <- seq(1,ncol(small_basket_correlations))
plot(time_axis, small_basket_correlations[1,], type='l', col='darkgreen')
lines(time_axis, small_basket_correlations[2,], type='l', col='red')
lines(time_axis, small_basket_correlations[3,], type='l', col='blue')
lines(time_axis, small_basket_correlations[4,], type='l', col='yellow')
lines(time_axis, small_basket_correlations[5,], type='l', col='pink')
legend(x='bottomleft', legend=c("MSFT", "INTC", "YHOO", "TXN", "CSCO"),
col=c("darkgreen","red", "blue", "yellow", "pink"), lwd=1, lty=c(0,0),
        pch=c(3,3))
```



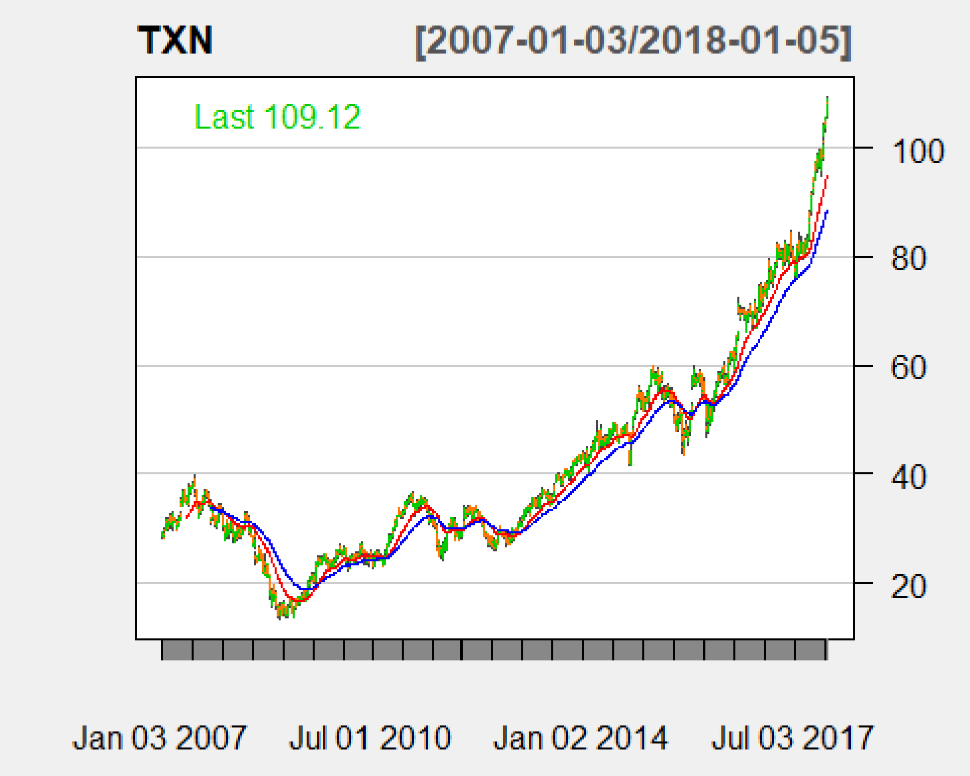
```
print ("STEP 2.10:Basket Analysis * Applying correlations to entries")

## [1] "STEP 2.10:Basket Analysis * Applying correlations to entries"

 library(quantmod)
 library(binhf)
 basket_symbols <- c('TXN', 'QQQ')
 getSymbols(basket_symbols, src='google')

## [1] "TXN" "QQQ"

 basket <- data.frame(as.xts(merge(TXN, QQQ)))
basket <- basket[,names(basket)[grepl(x=names(basket), pattern='Close')]]

#This is a very simplistic arbitrage-type trade.

#So, what if we buy/hold one of these whenever its far from the index?
#So , let's pick a stock that doesn't overly control the index TXN.

getSymbols(c('TXN', 'QQQ'), src='google')

## [1] "TXN" "QQQ"

basket_years <- unique(substr(rownames(basket), start=1, stop=4))
basket_months <- unique(substr(rownames(basket), start=1, stop=7))
small_basket <- basket[,names(basket)[grepl(x=names(basket),
pattern='TXN|QQQ')]]
```

```
TXN_QQQ <- c()
for (yearmonth in basket_years) {
        temp_df <- small_basket[substr(rownames(basket), start=1,
stop=4)==yearmonth,]
        TXN_QQQ <- cbind(TXN_QQQ, cor(temp_df$TXN.Close, temp_df$QQQ.Close))
}

small_basket_correlations <- data.frame(rbind(TXN_QQQ))
colnames(small_basket_correlations) <- basket_years

par(mfrow=c(2,1))
plot(as.Date(row.names(basket)), basket$TXN.Close, col='red',
     type='l', ylab="price", xlab='')
par(new=TRUE)
plot(as.Date(row.names(basket)), basket$QQQ.Close, col='gray', type='l',
xaxt="n",yaxt="n",ylab="", xlab='time')
legend("topright",col=c("red","gray"),lty=1,legend=c("TXN","QQQ"))

plot(type='l', col='darkgreen',
x=as.factor(names(small_basket_correlations)),
y=as.numeric(small_basket_correlations[1,]))
lines(type='l', col='darkgreen',
x=as.factor(names(small_basket_correlations)),
     y=as.numeric(small_basket_correlations[1,]))
```

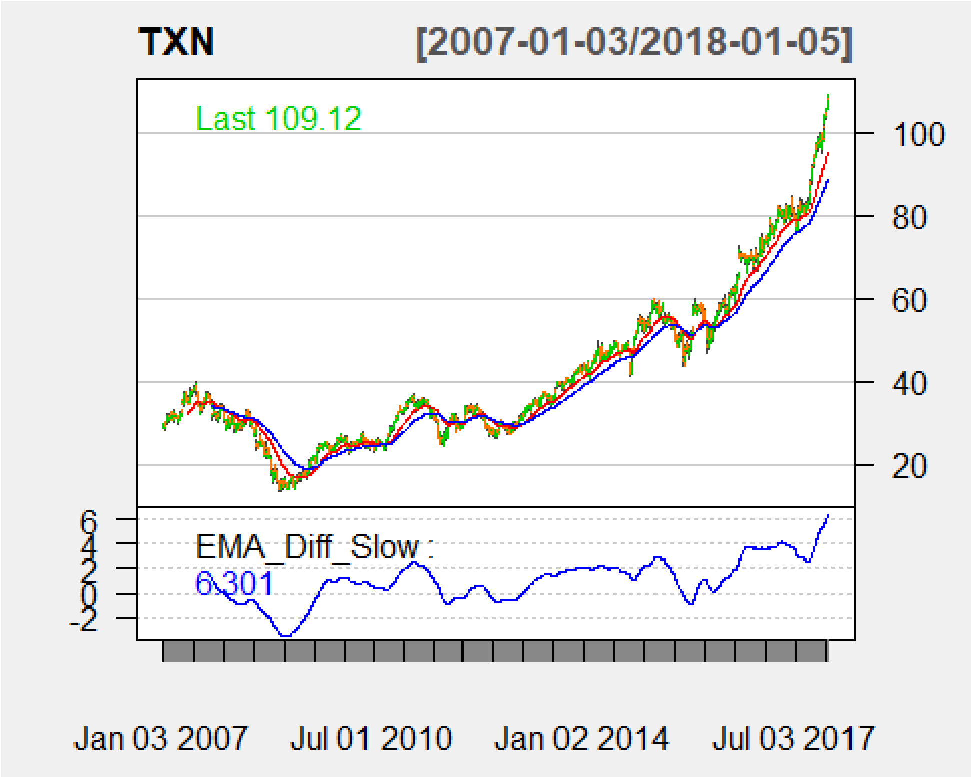




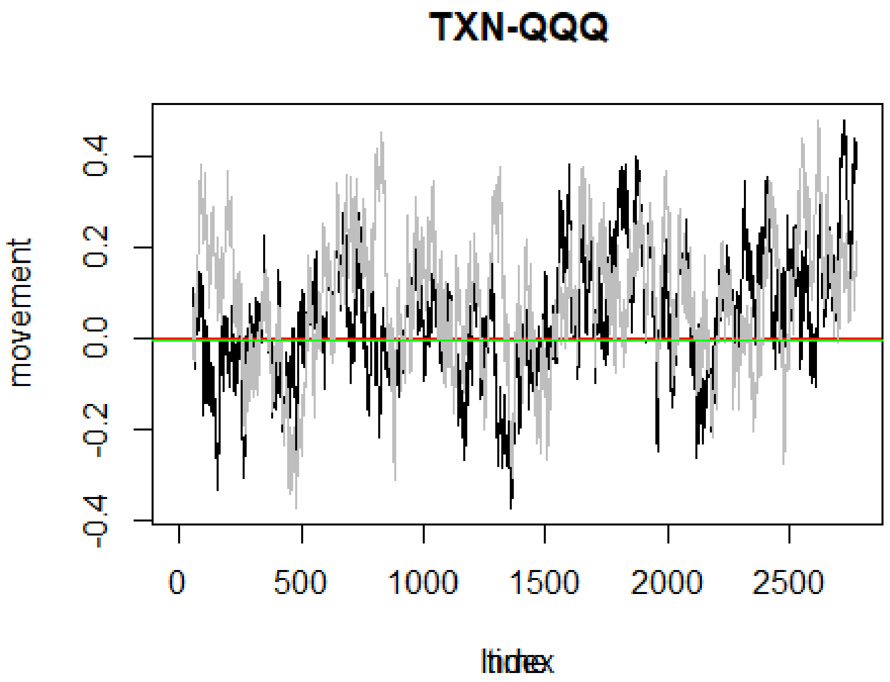
```
#So, let's create moving-average differences like we did in previous lectures
to capture trends:

EMA.Fast <- EMA(TXN$TXN.Close, n=30)
EMA.Medium <- EMA(TXN$TXN.Close, n=100)
EMA.Slow <- EMA(TXN$TXN.Close, n=200)
EMA_Diff_Fast <- EMA.Fast - EMA.Medium
EMA_Diff_Slow <- EMA.Medium - EMA.Slow

chartSeries(TXN, theme="white", TA="addEMA(n=100, col='red');addEMA(n=200,
col='blue')")
```



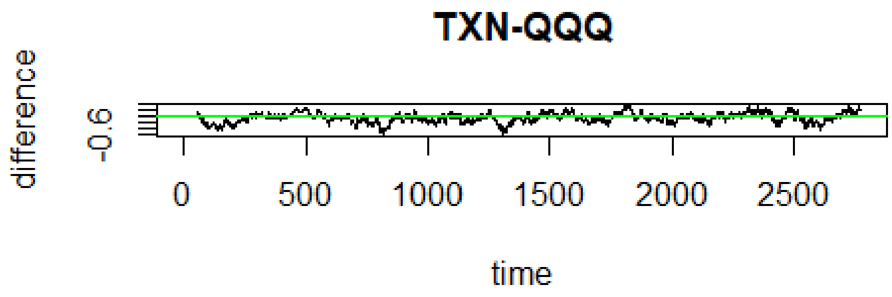
```
addTA(EMA_Diff_Slow, col='blue')
```

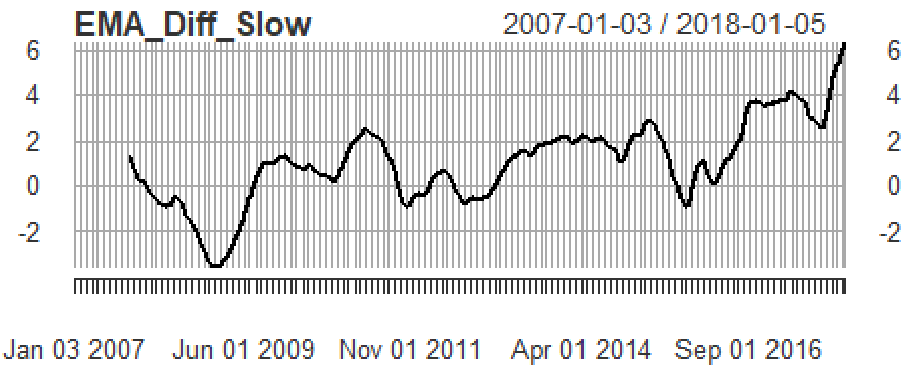


```
QQQ$QQQ.movement <- EMA(ifelse(ClCl(QQQ)  > 0, 1, -1),50)
TXN$TXN.movement <- EMA(ifelse(ClCl(TXN) > 0, 1, -1),50)

plot(as.numeric(TXN$TXN.movement ), col='black', ylab="movement",  main='TXN-QQQ', type = 'l')
abline(h=0, col='red')
par(new=TRUE)
plot(as.numeric(QQQ$QQQ.movement ), col='gray', xaxt="n",yaxt="n",ylab="", xlab='time', type='l')
abline(h=0, col='green')
```



```
par(mfrow=c(2,1))
diff <- as.numeric(TXN$TXN.movement)-as.numeric(QQQ$QQQ.movement)

## Warning in as.numeric(TXN$TXN.movement) - as.numeric(QQQ$QQQ.movement):
## longer object length is not a multiple of shorter object length

## Warning in as.numeric(TXN$TXN.movement) - as.numeric(QQQ$QQQ.movement):
## longer object length is not a multiple of shorter object length
plot(diff,
     col='black', type='l', xlab='time', ylab='difference',
     main='TXN-QQQ')
abline(h=0, col='green')
plot(EMA_Diff_Slow)
abline(h=0, col='green')
```





```
print ("end of script.")

## [1] "end of script."
```

Perl Script

```
#####################################################################################

#  Program name: DataCleanup.pl

# Porpose: Source Data whish is going to be used on my project is stock price history

# Source files are indeviual and has been downloaded from S&P 500, In addition

# There are several lines / securities which does not have any price for certain reason,

# in order to run my preditcion model, I need to have all recods data, Hence I need to a prgram .

#####################################################################################

# Algorithm:

#           1) All files should be merge into ONE master file, in order to recognize the symbol ,

#              symbol name or file name will be placed on first column. Files are in CSV format
```

```
#	2) All securites must have value as price (Open , Low, High, Close, Volumn). However they may not have

#	   been consistancly priced

#	3) I used mean to find the missing price by looking into the average of two close date into that missing date

#	4) There is no limitation for input files / stock market price, however I'm going to load around 500 symbol / files

#	   from S&P 500. http://www.nasdaq.com/symbol

################################################################################

#  Author	Version			Date			Descreption

#  Saeid Rezaei	0			12-Dec-2017		Initial version, for Capstone project

################################################################################


use strict;

use warnings;

use Data::Dumper;


my $scriptName ="DataCleanup.pl";

my $some_dir = "C:/CHM136/StockPriceHist/";

print ("Your file directory is:$some_dir \n");

my $outPut ='C:\CHM136\StockPriceHist\output\secPriceHistory.csv';

my ($headerLine,$symbol,$cntFile);

my @myLine;

my ($mySymbol,$myDate,$myCode,$myOpenPrice,$myHighPrice,$myLowPrice,$myClosePrice,$myVolumn);
my ($myPrvSymbol,$myPrvDate,$myPrvCode,$myPrvOpenPrice,$myPrvHighPrice,$myPrvLowPrice,$myPrvClosePrice,$myPrvVolumn);

print ("Script: $scriptName is started \n");

opendir(DIR, $some_dir) || die "can't opendir $some_dir: $!";

my @files = grep { /csv/ } readdir(DIR);

closedir DIR;

open OUT, ">>$outPut";

$headerLine ="stockName,Date,Usless,Open,High,Low,Close,Volumn";

printf OUT ("%s\n",$headerLine);

#exec("del C:\CHM136\StockPriceHist\output\secPriceHistory.csv");

$cntFile=0;

foreach my $f (@files) {

  # open IN, "<$f";

   print ("Input File is: <<$f>> \n");

   $cntFile ++;

   $symbol = substr($f,6,length($f)-10);

   print ("Symbol is :$symbol \n");

   open(my $fh, '<:encoding(UTF-8)', $some_dir.$f)

 or die "Could not open file '$f' $!";

  # my @cmpids = ();


($myPrvSymbol,$myPrvDate,$myPrvCode,$myPrvOpenPrice,$myPrvHighPrice,$myPrvLowPrice,$myPrvClosePrice,$myPrvVolumn)=0;

   while(my $line =<$fh>) {

      #push @cmpids, $_;
```

```
chomp $line;


@myLine = split (',',$line);
if (undef $mySymbol) {
  $mySymbol ='Undefined';
}
else {
  $mySymbol = $symbol;# $myLine[0];
}
#print ("$myLine[0] \n");
#unless ($myDate = $myLine[0];
if (defined ($myLine[0])) {  $myDate =$myLine[0];  }
else {$myDate = '19000101'; }
$myCode    = 0;#$myLine[1];
if (defined $myLine[2] and length ($myLine[2])!=0) { $myOpenPrice = $myLine[2]; }
else { $myOpenPrice=$myPrvOpenPrice;  }
if (defined  $myLine[3] and length ($myLine[3])!=0 ) {$myHighPrice = $myLine[3];}
else {$myHighPrice = $myPrvHighPrice;}
if (defined $myLine[4] and length ($myLine[4])!=0) {$myLowPrice = $myLine[4];}
else  { $myLowPrice = $myPrvLowPrice; }
if (defined  $myLine[5] and length ($myLine[5])!=0) {$myClosePrice = $myLine[5]; }
else {$myClosePrice = $myPrvClosePrice;}
if (defined $myLine[6] and length ($myLine[6])!=0) {$myVolumn = $myLine[6];}
else {$myVolumn=$myPrvVolumn;}
```

```
 #     printf ("Symbol:%s, Date:%s, Code:%s
OpenPrice:%s,HighPrice:%s,LowPrice:%s,ClosingPrice:%s,Volumn:%s
\n",$mySymbol,$myDate,$myCode,$myOpenPrice,$myHighPrice,$myLowPrice,$myClosePrice,$myVolu
mn);

       #print OUT ("$symbol,$line\n");

       printf  OUT ("%s,%s,%s,%s,%s,%s,%s,%s
\n",$symbol,$myDate,$myCode,$myOpenPrice,$myHighPrice,$myLowPrice,$myClosePrice,$myVolumn)
;



($myPrvSymbol,$myPrvDate,$myPrvCode,$myPrvOpenPrice,$myPrvHighPrice,$myPrvLowPrice,$myPrv
ClosePrice,$myPrvVolumn)=


($mySymbol,$myDate,$myCode,$myOpenPrice,$myHighPrice,$myLowPrice,$myClosePrice,$myVolumn
);

   #    print ("Prev: $myPrvSymbol $myPrvDate $myPrvCode $myPrvOpenPrice $myPrvHighPrice
$myPrvLowPrice $myPrvClosePrice $myPrvVolumn \n");



   }

   close $fh;

}

   #open OUT, ">>$outPut";

   #print OUT Dumper(\@cmpids);

   print ("$cntFile file is proccessed! \n");

   print ("end of script. \n");

   close OUT;

  exit 0;
```

Result from Tableau:

Please check the other called StockMarketAnalyses.Tableau.pdf

Configuration:

- Create folder under your local hard drive C: and name is CHM136
- Copy DataCleaning.pl into above folder
- Create another sub folder under CHM136 and call it result
- Copy all results there

Reference:

1. https://en.wikipedia.org/wiki/Stock_market

2. R. Bisoi, P.K. Dash, "Prediction of financial time series and its volatility using a hybrid dynamic neural network trained by sliding mode algorithm and differential evolution", Int. J. on Information and Decision Science, Inderscience Publications, U.K., Vol. 7, no. 2, pp.166-191, 2015.

3. Ortega, Edgar; Yalman, Onaran (December 4, 2006). "UBS, Goldman Threaten NYSE, Nasdaq With Rival Stock Markets". Bloomberg.com. Retrieved 2011-05-31.

4. Hong Keel Sul,Alan R. Dennis,Lingyao (Ivy) "Trading on Twitter: The Financial Information Content of Emotion in Social Media", 2014 47th Hawaii International Conference on System Science

Saeid