Contents

By: Sulekha AloorRavi

FileName: SulekhaAloorravi_SVAP_Asmt_R2.Rmd

Title: Forecast Stock Returns on a stock for 180 days using Technical Analysis

my github: https://github.com/sulekhaaloorravi/SVAP_Assignments

1.Frame

1.1. What we want to do?

This assignment aims to:

Q.No.1?: Predict rate of change of stock returns for a period of 180 days for any one of the equity stocks by providing it as an input to the model used in this assignment.

Q.No.2?: Visualise data and draw insight whether to buy the selected stock.

1.2. Approach:

1: Use a simple Technical Analysis to perform Prediction.

2: Make use of nnet to train and test historical data and use prophet and predict functions to come up a forecast of returns for 180 days.

1.3. Important Packages Used and their brief explanation:

1. quantmod The quantmod package for R is designed to assist the quantitative trader in the development, testing, and deployment of statistically based trading models.
2. nnet The globally convergent algorithm is based on the resilient backpropagation without weight backtracking and additionally modifies one learning rate, either the learning rate associated with the smallest absolute gradient (sag) or the smallest learning rate (slr) itself.
3. prophet Implements a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly and weekly seasonality, plus holidays. It works best with daily periodicity data with at least one year of historical data.

Stock Chosen for this assignment: Oil and Natural Gas Corporation - NSE traded

```
#install.packages("quantmod")
#install.packages("neuralnet")
#install.packages("prophet")
#install.packages("tseries")
#install.packages("PerformanceAnalytics")
#install.packages("DMwR")
#install.packages("magrittr")
#install.packages("ggplot2")
#devtools::install_github("joshuaulrich/quantmod")
#or
#devtools::install_github("joshuaulrich/quantmod", ref="157_yahoo_502")
require(nnet)
```

```
## Loading required package: nnet
```

```
require(PerformanceAnalytics)
```

```
## Loading required package: PerformanceAnalytics
```

```
## Loading required package: xts
```

```
## Loading required package: zoo
```

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

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

```
##
## Attaching package: 'PerformanceAnalytics'
```

```
## The following object is masked from 'package:graphics':
##
##     legend
```

```
library(quantmod)
```

```
## Loading required package: TTR
```

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

```
library(tseries)
library(magrittr)
library(ggplot2)
library(plotly)
```

```
##
```

```
## Attaching package: 'plotly'

## The following object is masked from 'package:ggplot2':
##
##     last_plot

## The following object is masked from 'package:stats':
##
##     filter

## The following object is masked from 'package:graphics':
##
##     layout
```

```
library(prophet)
```

```
## Loading required package: Rcpp
```

```
library(DMwR)
```

```
## Loading required package: lattice

## Loading required package: grid
```

```
library(randomForest)
```

```
## randomForest 4.6-12

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin
```

2.Acquire

We will concentrate on forecasting the rate of change of returns for any one of the NSE traded Stock(here: ONGC.NS).

The data required for this exercise is acquired from Yahoo finance Site. The method used to acquire data is downloading it directly from web.

Following values will be downloaded as part of daily stock quotes data:

1. Date
2. Open Price
3. Highest Price
4. Lowest Price
5. Closing Price
6. Volume traded
7. Adjusted close price

```
getSymbols('ONGC.NS',from='1970-01-01',to='2017-05-18')
```

```
##     As of 0.4-0, 'getSymbols' uses env=parent.frame() and
##  auto.assign=TRUE by default.
##
```

```
##  This  behavior  will be  phased out in 0.5-0  when the call  will
##  default to use auto.assign=FALSE. getOption("getSymbols.env") and
##  getOptions("getSymbols.auto.assign") are now 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 more details.
```

```
## [1] "ONGC.NS"
```

```
NIFTY500<-ONGC.NS
```

3.Refine

I. Refine data by first correcting the following two data conditions:

1.the High and Low are equal for any given period, or

2.the Volume is zero for any given period.

This will help in avoiding the error: "Series contains non-leading NAs"

II. Update column headers as required for ease of usage.

Further more refinement of data also occurs in sections involving - Transform, Explore and Model

```
colnames(NIFTY500) <- c("Open", "High", "Low", "Close","Volume","Adjusted")

NIFTY500[,2] <- NIFTY500[,2] +1e-6
NIFTY500[,5] <- NIFTY500[,5] +1e-6

NIFTY500<-as.xts(na.omit(as.data.frame(NIFTY500)))

tail(NIFTY500)
```

```
##              Open   High    Low  Close   Volume Adjusted
## 2017-05-11 189.10 190.25 183.10 183.60 10796830   183.60
## 2017-05-12 184.75 186.40 184.00 184.85  6242350   184.85
## 2017-05-15 187.30 188.30 186.15 186.75  5766195   186.75
## 2017-05-16 187.45 187.50 184.60 185.15  6960470   185.15
## 2017-05-17 185.20 186.65 183.90 184.50  6074714   184.50
## 2017-05-18 184.40 184.40 180.50 180.90  4628258   180.90
```

4.Transform

Transform data to perform calculations by creating formulas and functions that would compute Technical indicators.

```
indicator <- function(histdata,margin=0.025,days=10) {

  v <- apply(HLC(histdata),1,mean)

  r <- matrix(NA,ncol=days,nrow=NROW(histdata))
  for(x in 1:days) r[,x] <- Next(Delt(Cl(histdata),v,k=x),x)

  x <- apply(r,1,function(x) sum(x[x > margin | x < -margin]))
```

```
  if (is.xts(histdata)) xts(x,time(histdata)) else x
}
```

Functions used to transform data by performing various calculations on the stock quotes data to compute Techincal indicator:

1. ATR: Average True Range is a measure of volatility of a High-Low-Close series.
2. SMI: The stochastic oscillator is a momentum indicator that relates the location of each day's close relative to the high/low range over the past n periods.
3. ADX: Welles Wilder's Directional Movement Index
4. Aroon: The Aroon indicator attempts to identify starting trends.
5. BBands: Bollinger Bands are a way to compare a security's volatility and price levels over a period of time.
6. Delt: Calculate Percent Change
7. Chaikin Volatility: Measures the rate of change of the security's trading range
8. EMA: Moving Averages of a series
9. CMO: Chande Momentum Oscillator
10. MACD: Price oscillator
11. MFI: Money Flow Index a ratio of positive and negative money flow over time

```
avgrng <- function(x) ATR(HLC(x))[,'atr']
stochas <- function(x) SMI(HLC(x))[,'SMI']
dirind <- function(x) ADX(HLC(x))[,'ADX']
aroonind <- function(x) aroon(x[,c('High','Low')])$oscillator
bands <- function(x) BBands(HLC(x))[,'pctB']
chaikin <- function(x) Delt(chaikinVolatility(x[,c("High","Low")]))[,1]
expavg <- function(x) EMA(CLV(HLC(x)))[,1]
ease <- function(x) EMV(x[,c('High','Low')],x[,'Volume'])[,2]
movavg <- function(x) MACD(Cl(x))[,2]
mnyind <- function(x) MFI(x[,c("High","Low","Close")], x[,"Volume"])
stprev <- function(x) SAR(x[,c('High','Close')]) [,1]
volatile <- function(x) volatility(OHLC(x),calc="garman")[,1]
chande <- function(x) CMO(Cl(x)) [,1]
expdel <- function(x) EMA(Delt(Cl(x))) [,1]
del <- function(x) Delt(Cl(x),k=1:10) [,9:10]
relind <- function(x) RSI(Cl(x)) [,'EMA']
mvmean <- function(x) runMean(Cl(x)) [,1]
mvsd <- function(x) runSD(Cl(x)) [,1]
```

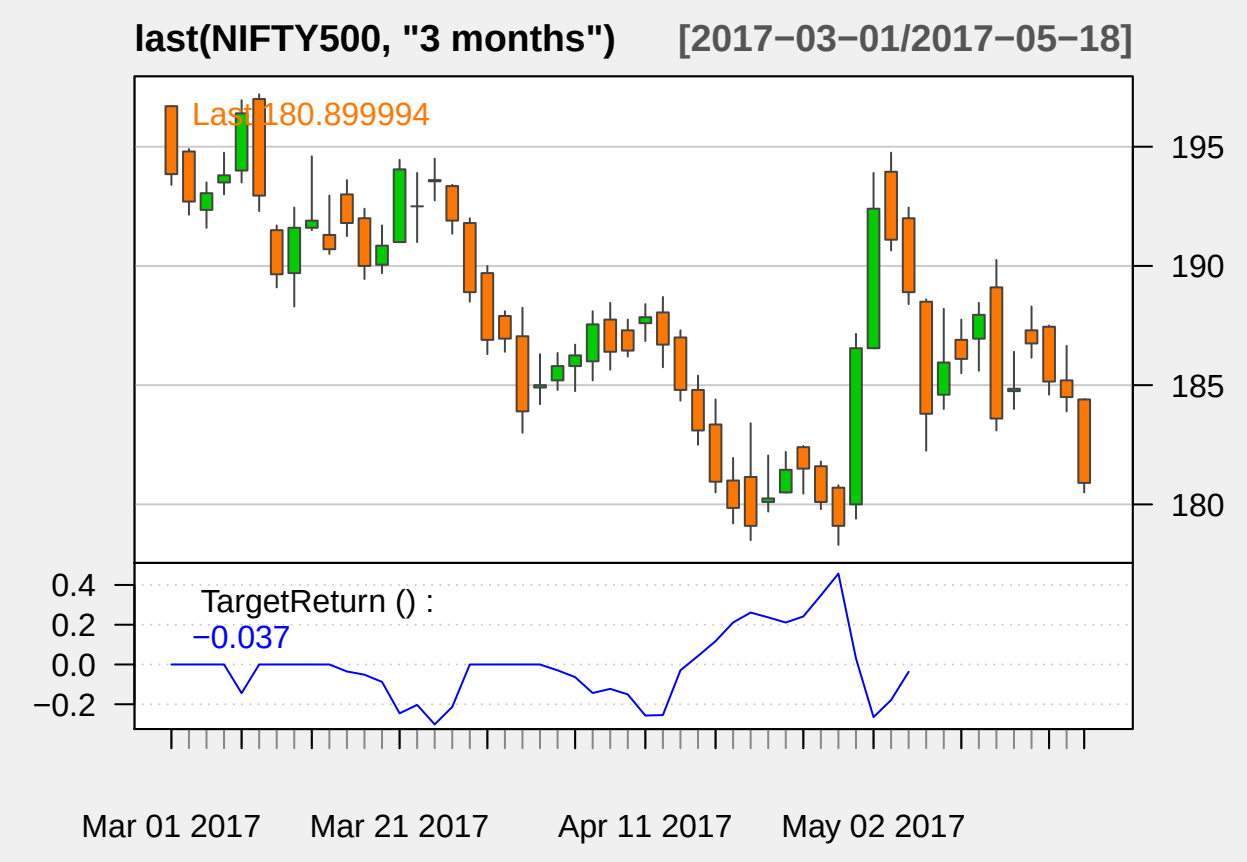
5.Explore

A function in quantmod named chartSeries is used to create standard financial charts given a time series like object.

This is a base function for future technical analysis additions. Possible chart styles include candles, matches, bars, and lines.

```
candle <- candleChart(last(NIFTY500,'3 months'),theme='white',TA=NULL)
```


```
addindicator <- newTA(FUN=indicator,col='blue',legend="TargetReturn")
addindicator()
```



addBBands - Bollinger Bands will be drawn on the current chart. Bollinger Bands are a way to compare a security's volatility and price levels over a period of time.

```
chartSeries(NIFTY500, subset='last 3 months')
```


```
addBBands(n = 20, sd = 2, ma = "SMA", draw = 'bands', on = -1)
```

6.Model

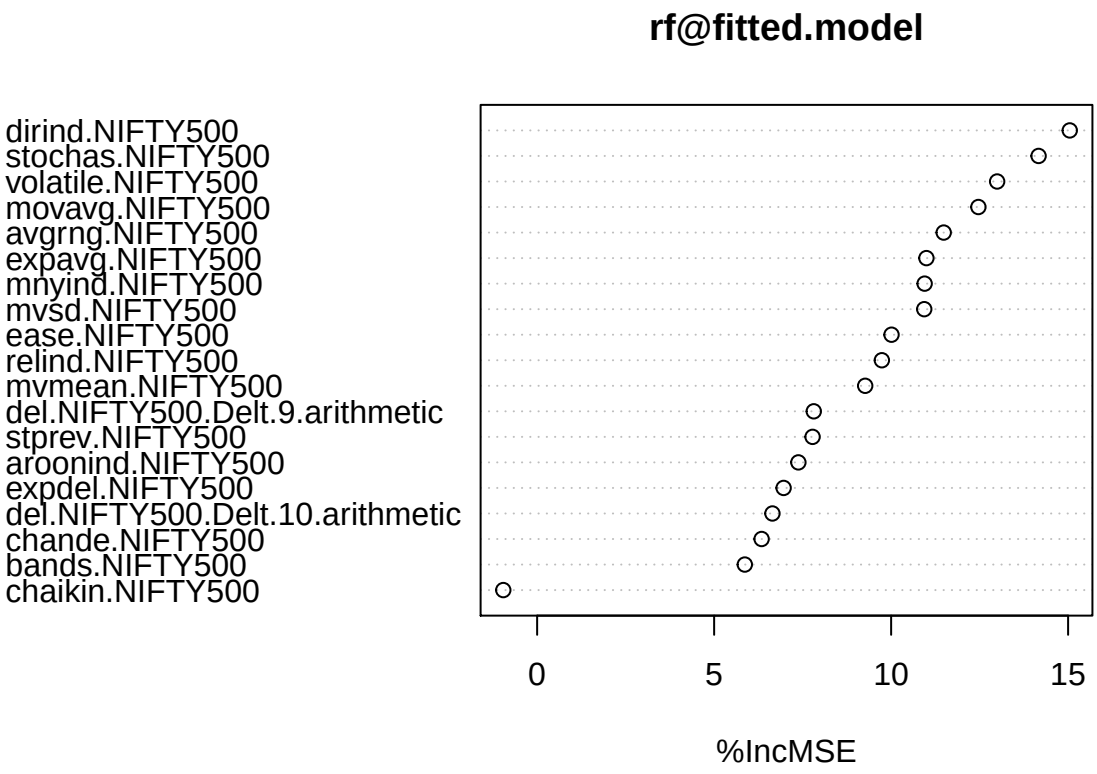
Define a Data Model, fit the model using randomforest and train the model using artificial neural networks.

```
data.model <- specifyModel(indicator(NIFTY500) ~ del(NIFTY500) + avgrng(NIFTY500) + dirind(NIFTY500)
                           + aroonind(NIFTY500) + bands(NIFTY500)  + chaikin(NIFTY500)
                  + expavg(NIFTY500)+ chande(NIFTY500) + expdel(NIFTY500) + volatile(NIFTY500)
                                              + movavg(NIFTY500) + mnyind(NIFTY500)
                  + relind(NIFTY500) + stprev(NIFTY500) + mvmean(NIFTY500) + mvsd(NIFTY500)
                  + ease(NIFTY500) + stochas(NIFTY500))

set.seed(1234)

rf <- buildModel(data.model,method='randomForest',
             training.per=c(start(NIFTY500),index(NIFTY500["2015-12-02"])),
             ntree=50, importance=T)

#Plot the importance of variable and pick variables accordingly, to fit into the model
varImpPlot(rf@fitted.model,type=1)
```




```
imp <- importance(rf@fitted.model,type=1)
rownames(imp)[which(imp > 10)]
```

```
## [1] "avgrng.NIFTY500"   "dirind.NIFTY500"   "expavg.NIFTY500"
## [4] "volatile.NIFTY500" "movavg.NIFTY500"   "mnyind.NIFTY500"
## [7] "mvsd.NIFTY500"     "ease.NIFTY500"     "stochas.NIFTY500"
```

Fit model based on variable importance plotted from the above random forest.

```
data.model <- specifyModel(indicator(NIFTY500) ~ del(NIFTY500) + avgrng(NIFTY500) + dirind(NIFTY500)
                           + expavg(NIFTY500)+ volatile(NIFTY500)  + mnyind(NIFTY500)
                           + mvsd(NIFTY500) + ease(NIFTY500) + stochas(NIFTY500))

DataFrame <- as.data.frame(modelData(data.model))
traincount<-nrow(DataFrame) - 3030
totalcount<-nrow(DataFrame)
testcount<- 1+traincount

trainDF<-DataFrame[1:traincount,]

testDF<-DataFrame[testcount:totalcount,]

form <- as.formula('indicator.NIFTY500 ~ .')
train.data <- scale(trainDF)
test.data <- scale(testDF)
```

Model: Artifical neural network

Train a window of data using Artificial neural networks. Then test a window of data with the model.

```
nn<-nnet(form,train.data,size=10,decay=0.01,maxit=1000,linout=T,trace=F)
prediction <- predict(nn,test.data)

preds <- unscale(prediction,test.data)

write.csv(preds,"pred.csv")
pred<-read.csv("pred.csv")
```

Plot - Predictions

```
predplot <- as.data.frame(cbind(as.character(pred$X),pred$V1))
predplot$V1 <- as.Date(predplot$V1)
predplot$V2 <- as.character(predplot$V2)
predplot$V2 <- as.numeric(predplot$V2)

colnames(predplot) <- c("Date","TechnicalIndicator")

test1<-as.xts(testDF)
test1 <- test1[,9]

signal1 = Lag(ifelse(predplot$TechnicalIndicator < 0.025, -1, 1))
neuralret <- ROC(test1) * signal1
neuralret = neuralret['2016-11-15/2017-05-18']
portfolio1 = exp(cumsum(neuralret))
table.CalendarReturns(neuralret)
```

```
##      Jan   Feb  Mar  Apr   May Jun Jul Aug Sep Oct  Nov  Dec mvsd.NIFTY500
## 2016  NA    NA   NA   NA    NA  NA  NA  NA  NA  NA 18.1 -7.7           9.0
## 2017 0.2 -29.2 10.2 21.8 -97.5  NA  NA  NA  NA  NA   NA   NA         -97.6
```

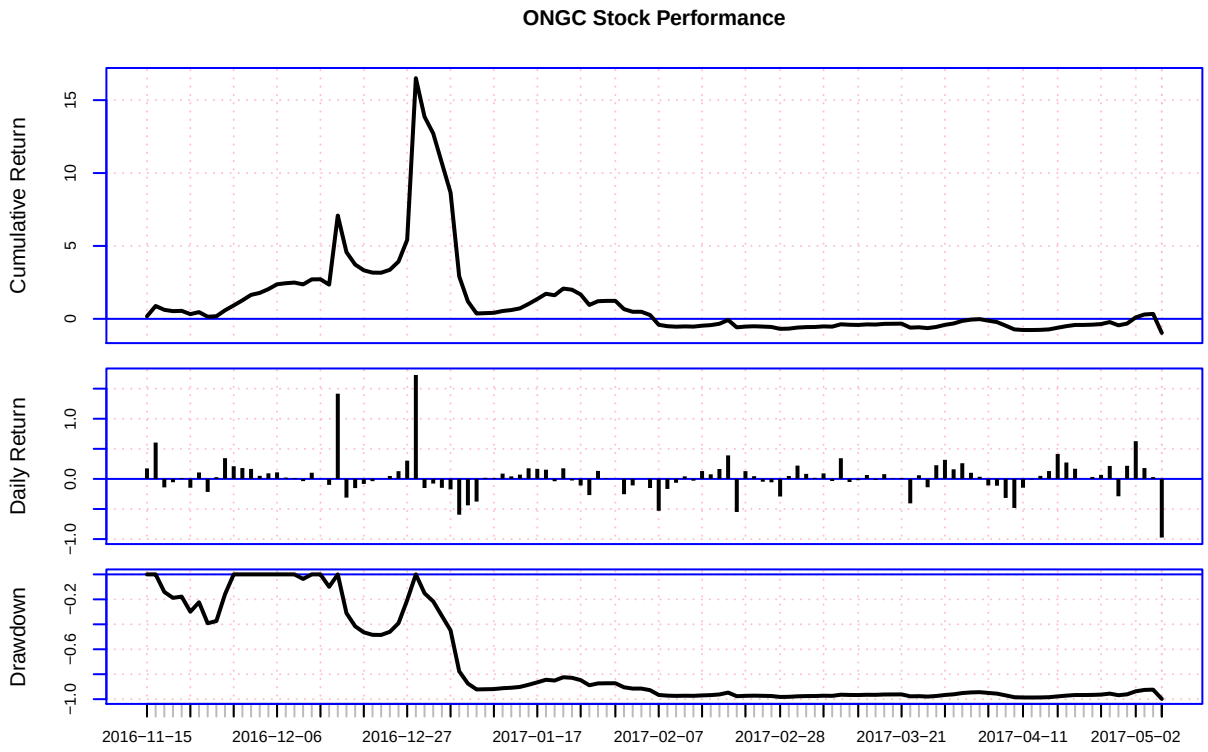
```
table.Drawdowns(neuralret)
```

```
##         From     Trough         To   Depth Length To Trough Recovery
## 1 2016-12-29 2017-05-18       <NA> -0.9981     87        86       NA
## 2 2016-12-16 2016-12-22 2016-12-28 -0.4853      9         5        4
## 3 2016-11-17 2016-11-24 2016-11-29 -0.3916      9         6        3
## 4 2016-12-14 2016-12-14 2016-12-15 -0.0994      2         1        1
## 5 2016-12-09 2016-12-09 2016-12-12 -0.0365      2         1        1
```

```
table.DownsideRisk(neuralret)
```

```
##                               mvsd.NIFTY500
## Semi Deviation                       0.1854
## Gain Deviation                       0.2737
## Loss Deviation                       0.1954
## Downside Deviation (MAR=210%)        0.1758
## Downside Deviation (Rf=0%)           0.1721
## Downside Deviation (0%)              0.1721
## Maximum Drawdown                     0.9981
## Historical VaR (95%)                -0.4115
## Historical ES (95%)                 -0.5953
## Modified VaR (95%)                  -0.2243
## Modified ES (95%)                   -0.2783
```

```
charts.PerformanceSummary(neuralret,grid.color = "pink",
                          element.color = "blue",main="ONGC Stock Performance")
```



Day wise - daily rate of change of return forecast based on artificial neural networks and prophet and predict. Trained and tested model data is provided as input to prophet to forecast future rate of change of returns for next 180 days.

```
colnames(neuralret) <- c("Returns")
neuralret<-as.data.frame(neuralret)
write.csv(neuralret,"neuralreturns.csv")
neuralreturns<-read.csv("neuralreturns.csv")

neuretplot <- as.data.frame(cbind(as.character(neuralreturns$X),neuralreturns$Returns))

neuretplot$V1 <- as.Date(neuretplot$V1)
neuretplot$V2 <- as.character(neuretplot$V2)
neuretplot$V2 <- as.numeric(neuretplot$V2)
colnames(neuretplot) <- c("Date","Returns")

ts1 <- neuretplot %>%
  select(Date,Returns)
colnames(ts1)<-c("ds","y")
m1<-prophet(ts1)
```

```
## STAN OPTIMIZATION COMMAND (LBFGS)
## init = user
## save_iterations = 1
## init_alpha = 0.001
```

```
## tol_obj = 1e-012
## tol_grad = 1e-008
## tol_param = 1e-008
## tol_rel_obj = 10000
## tol_rel_grad = 1e+007
## history_size = 5
## seed = 1922624226
## initial log joint probability = -8.27768
## Optimization terminated normally:
##   Convergence detected: absolute parameter change was below tolerance
```

```
future1 <- make_future_dataframe(m1,period=180,freq="day")
forecast1 <- predict(m1,future1)

forecast1plot<-plot(m1,forecast1) + ylab("ROC of Returns") + xlab("Date Series")
forecast<-forecast1[,c(1,3,10,13,16,17)]
```

7.Communicate - Insight

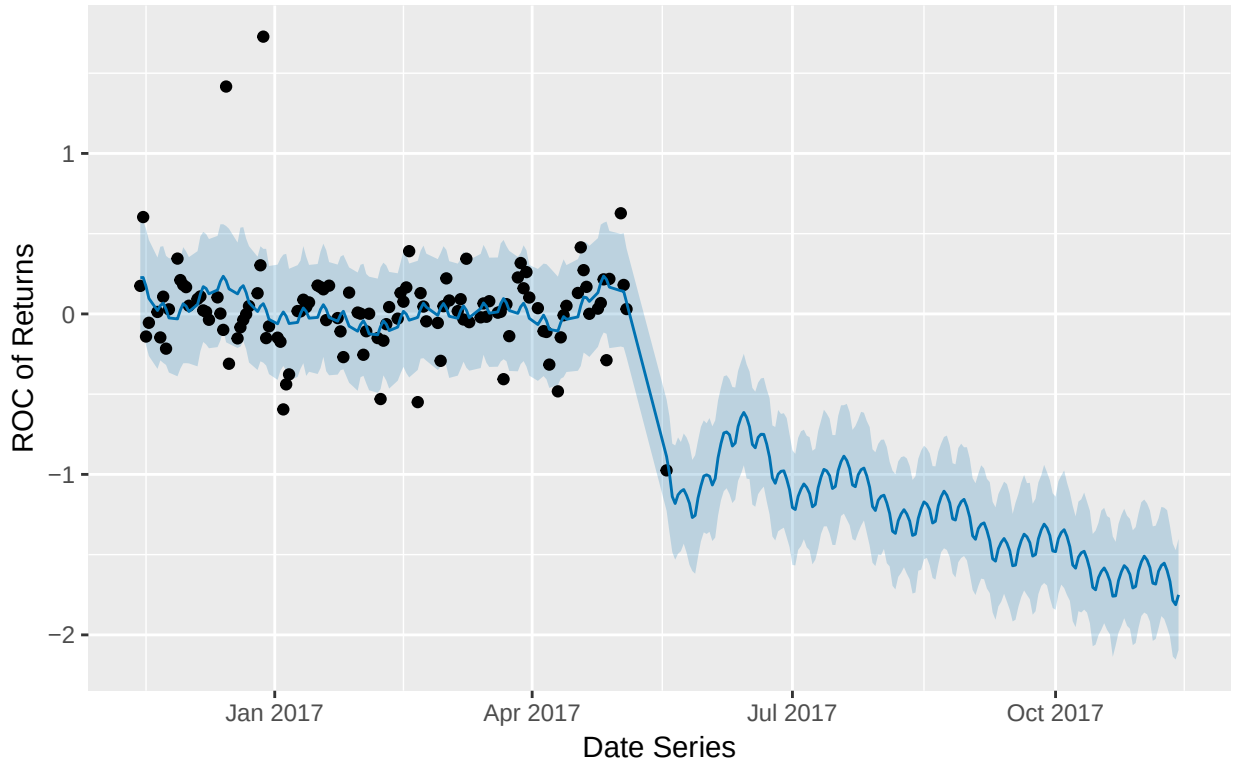
Visualise the forecasts predicted by the above described model and decide whether to buy the stock based on the direction of forecasted rate of change of returns . If the ROC of return is forecasted above the margin of 0.025 then buy, if it is less than 0.025 do not buy the stock.

Plot forecast predicted by prophet

```
ggplotly(forecast1plot)
```

Forecasted rate of change of returns for next 180 days

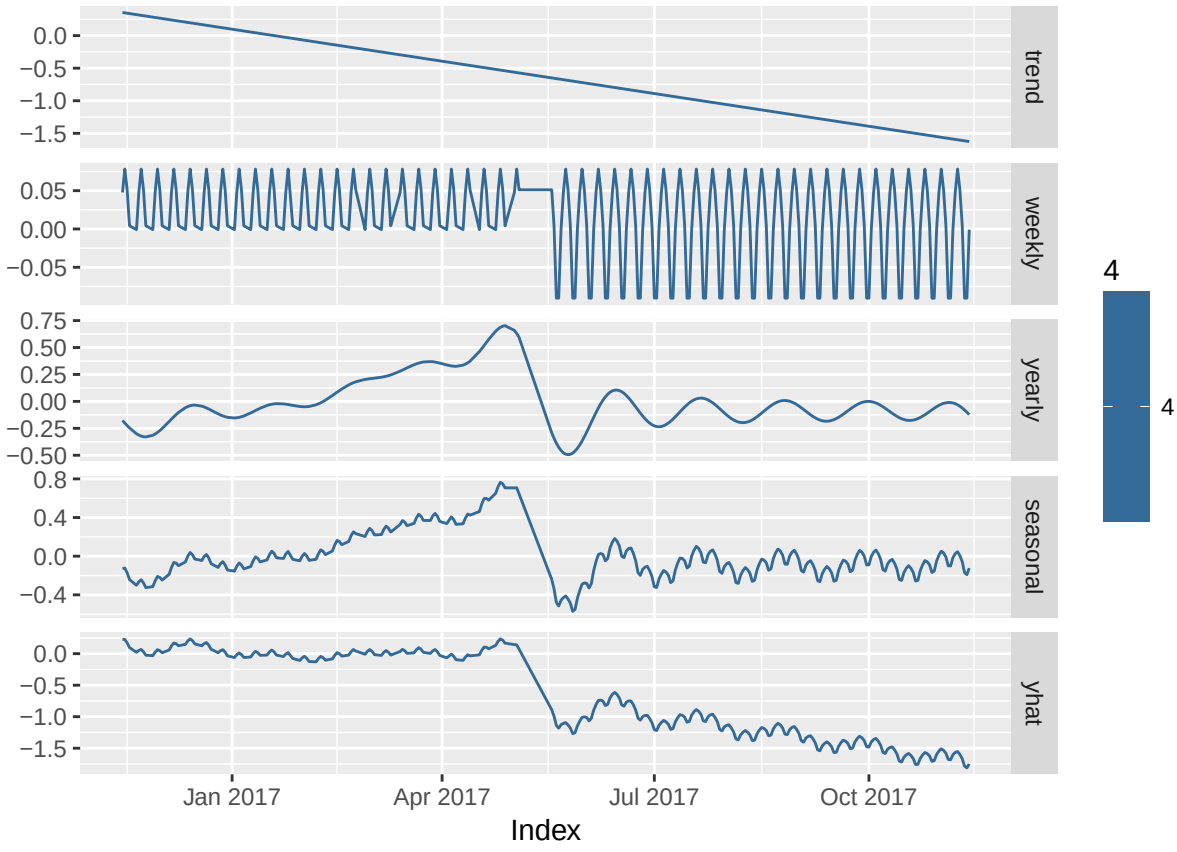
```
plot(forecast1plot)
```




```
z<-read.zoo(forecast)
```

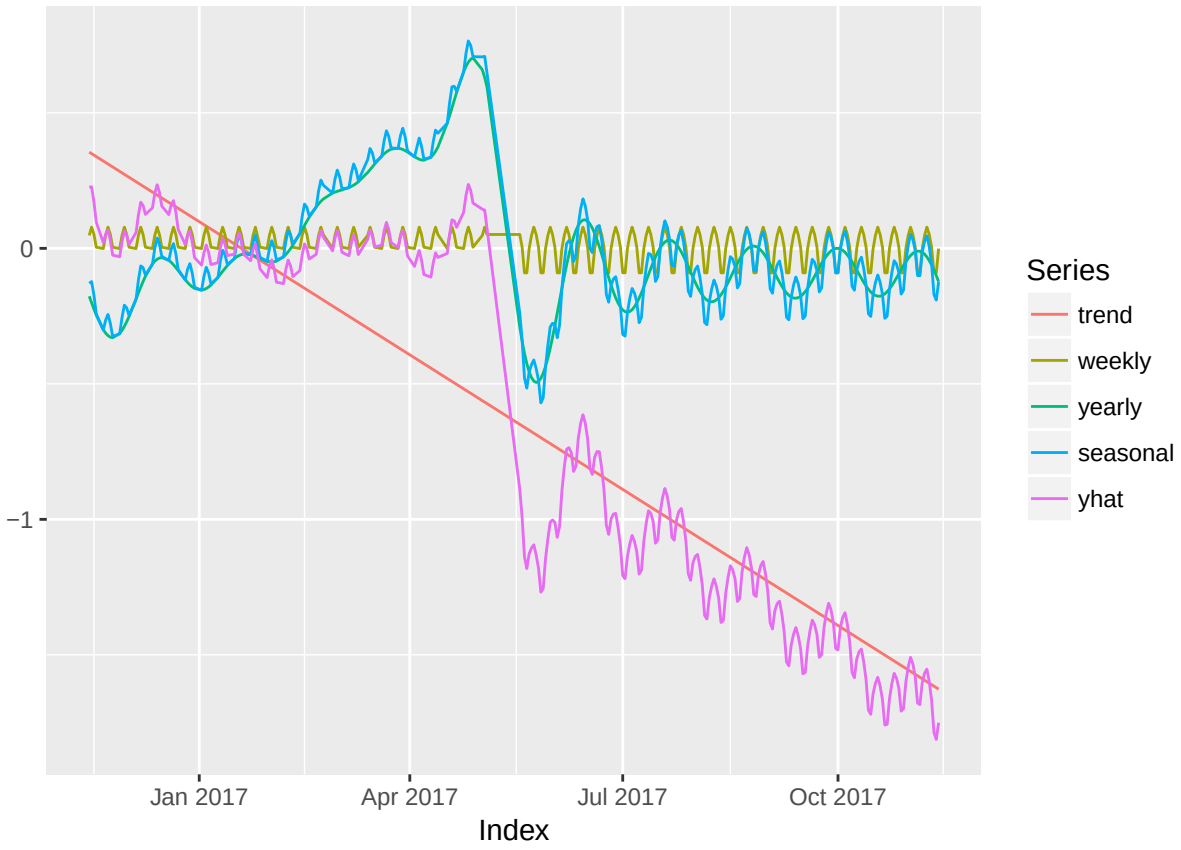
Forecasted time series with trend of returns divided in facets

```
autoplot(z,col=4) + facet_free()
```



Forecasted time series with trend of returns at weekly, monthly, yearly and seasonal trends

```
autoplot(z, facet = NULL)
```



This Technical Analysis described in this assignment is based only on technical indicator and not fundamental analysis. Also, this forecasting would give a relatively fare prediction when forecasted for lesser time periods such as <1 year and the trend may look repetitive if it exceeds one year.

Conclusion:

Looking at its predicted return trends from the above graphs. Returns might go down in the next 6 months.

Oil and Natural Gas Corporation stock is not recommended to buy.