CA02

Amarnath V

November 11, 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.

Introduction:

Let me state my role as "Portfolio Manager".

Portfolio Manager:

The Portfolio Manager is a person who drafts an investment strategy to make an investment decision for performing investment activities like investing a mutual or closed end fund's assets for the vested financial institutions or individuals [1].

This role includes corresponding with clients and coming to arrangements and resolutions with them.

I am required to liaise with liquidators, third parties and colleagues of mine in other departments in my organization.

I am also required to assess Standard Financial Statements and proposals sent in by customers and attempt to work with the customer to come to a solution.

Scenario:

One day, an individual investor named "Bill Gates"(you heard right, the founder of Microsoft which is one of the biggest technology organizations) approached the financial institution in which I am working as a portfolio manager. He is interested to invest in familiar commodities in the U.S. namely, soybean and natural gas and he is also interested to invest in mutual funds and ETF's (Exchange-Traded funds) but he is quite worried, how his investment performs so he needs our institution to analyze for him and let him know how his investments perform if he does the investment on the above mentioned categories. Being a portfolio manager, I am responsible to perform the investment analysis and advice him how his investment would perform if he would invest.

Let's first draft an insights for the investments he have chosen,

Insights of Soybean:

As we all know US top exporting commodity is Soybean and China is the highest consumer of the soybean. China is the most populous country in the world so obviously consumption will be high. Even though China looking elsewhere to import soybean due to trade conflict with US but it's very hard to implement and if they can it will take years. So, soybean is the right choice to invest but when to buy it? This is what a portfolio manager discovers via his analysis, being a portfolio manager let me apply a suitable statistical model (Artificial Neural Network) to the data we have and find the perfect time to buy or sell the investment to get good returns. https://tradingeconomics.com/commodity/soybeans/forecast

Insights of natural gas:

Investing in commodity is a right choice and that too investing in energy and agricultural commodity is the best choice because of the past market trends. Even though natural gas market is not stable, with the help of our organization investor can invests at low cost and sells at high rate. It will become possible with the help of Artificial Neural Network model. By applying this model to the commodity futures data we could predict the future value of commodities and advice the investor based on whether he needs to sell or buy the investments on commodities.

Insight on Mutual Funds

Mutual funds are one of the most familiar investments on the market. Most people select mutual funds for their investment portfolio because of the funds' expertized management, many people purchase mutual funds because of their competitive returns. Others are like them because they are easy to buy and sell. Still others cite the fact that mutual funds, because they hold several investments, can spread risk. Mutual fund schemes are made up of stocks belonging to the market. After seeing the performance of the stock market at the end of 1 year, decision to invest in mutual funds or not will be taken. With the help of time series forecast model ARIMA let's study the performance of requested stock market index and let's advice the investor the right time for buying and selling the mutual funds.

Insights on Exchange trade funds:

Exchange-traded fund or simply ETF, is a pool of securities that trades on an exchange, more like a stock. ETFs are usually very transparent—their underlying holdings are fully revealed on a daily basis—and are highly liquid. Similar to mutual funds ETF's also depends on stock market so with the help of ARIMA model let's forecast the stock's performance and advice investor which is a right time to buy and which is a right time to sell.

The key academic work defending my insights will be deeply explained in the "Methods" section of this document.

Dataset:

While starting my work for the customer, my boss, Simon gave a new task to prepare a consolidated currently trending investment dataset for the organization future worl. As my current work related to this I have decided to take up this task to create a consolidated dataframe and fetch the required features from the consolidated dataframe itself.

Take a look at the data of various trending investments that my boss suggested below and I could see all the required features that I need to perform analysis for the customer.

```
Investments <- c("Crypto-currencies","Fiat
currencies","Commodities","Technology Companies","Banking","Stock Market")
Datasets <- c("Dash and ZCash","Canadian Dollar and Chinese Yuan","Soybean
and Natural gas","CISCO and Google",
              "Bank of China and Royal Bank of Canada","NASDAQ-S&P500 and
EUREX-STOXX50")
choice <- data.frame(Investments,Datasets)
library(pander)
pandoc.table(choice, style = "grid", caption = "Consolidated trending
investment dataset of my organization")
```

```
##
##
## +----------------------+------------------------------+
## |     Investments      |           Datasets           |
## +======================+==============================+
## |  Crypto-currencies   |        Dash and ZCash        |
## +----------------------+------------------------------+
## |   Fiat currencies    | Canadian Dollar and Chinese  |
## |                      |             Yuan             |
## +----------------------+------------------------------+
## |     Commodities      |   Soybean and Natural gas    |
## +----------------------+------------------------------+
## | Technology Companies |       CISCO and Google       |
## +----------------------+------------------------------+
## |       Banking        | Bank of China and Royal Bank |
## |                      |          of Canada           |
## +----------------------+------------------------------+
## |     Stock Market     |      NASDAQ-S&P500 and       |
## |                      |        EUREX-STOXX50         |
## +----------------------+------------------------------+
##
## Table: Consolidated trending investment dataset of my organization
```

Before starting consolidating dataset let's have a brief note on each of the datasets he suggested.

Cryptocurrency:

Cryptocurrency is a computer-generated or digital currency devised to operate as a medium of exchange. It uses cryptography approaches to safeguard and authorize transactions as well as to control the creation of new parts of a particular cryptocurrency. Essentially, cryptocurrencies are some degree of registers in a database that no one can alter unless particular conditions are satisfied.

Dash and Zcash:

Dash and Zcash are an open source (source code is freely available) cryptocurrency which is developed by a decentralized independent company that provides confidentiality and choosy transparency of dealings. Among these cryptocurrencies Dash is run by a subsection of users, called “masternodes”. Some of us want our transaction to be privacy from end to end and yes! we do have privacy coins such as dash and zcash which are being top 15 coin-market cap and those cryptocurrencies are built by security specialized technical team.

Fiat currency:

Fiat currency is authorized tender whose worth is supported by the government that provided it. The U.S. dollar is fiat money, as are the euro and many other major world currencies. This process varies from money whose worth is supported by some physical good such as gold or silver, called commodity money.

Canadian Dollar

The Canadian currency outclassed almost all of its key peers in 2017. Expecting the same for upcoming years. This is the reason for incorporating this currency data as one of my datasets. Canadian Dollar exchange rate against Euro

Chinese Yuan

China is the major manufacturing economy in the world, it is also the world’s largest exporter nation, and the world’s second largest economy. With its enormous and still developing financial footprint, the financial world is observing the Chinese yuan as a global currency and worthwhile investment. Can the yuan ever serve as a potential alternative to the U.S. dollar? Analysts tracking the Chinese yuan believe that now may be the better period to invest in the Chinese currency. Chinese yuan exchange rate against euro

Commodities:

Let’s discuss about this in “Insight” section of this document

Technology companies:

Alphabet Inc.:

Alphabet is a familiar organization which is the parent company for technology giant google– the source of discovering information to get answer the any questions of the people from all over the world. Google is directing the people to path in global innovation, from new types of artificial intelligence to driverless cars.

CISCO SYSTEMS INC.

CISCO SYSTEMS INC. IS THE GLOBAL FRONT-RUNNER in networking for the Internet. The company was started in 1984 by two computer experts from Stanford University looking a simple step to inter-connect various kinds of computer systems. In 1986, Cisco Systems dispatched its first creation and now it has grown up as a multi-national corporation, with more than 35,000 employees in all over115 countries. Cisco's networking solutions connect people and electrical devices via their network solutions, let people to transfer or take a look at the information without any barriers.

Banks:

The Royal Bank of canada

The Royal Bank of Canada is the biggest Canadian bank by market capitalization, at ~$94 billion as placed on the NYSE. It was established in 1869 as a sector bank, and from then it has developed to assist customers in Canada, the US, and 37 various other countries. The bank's dividend streak is remarkable, holding paid stable or growing dividends each year since 1943. It fixed its dividend for 2 years over the financial crisis, and has enlarged it each year since 2010. Source: Investor Presentation

Bank of china:

Bank of China was officially implemented in February 1912. The Bank continuously assisted successively as the country's central bank, international exchange bank and specialized international trade bank. Satisfying its promise to helping the public and emerging China's financial services sector, the Bank grew to a demanding place in the Chinese financial industry and created a good positioning in the international financial group, despite various difficulties and failures. Throughout China's reform and staring up time, the Bank achieved the historic chance provided by the government's plan of capitalizing on foreign funds and innovative technologies to progress economic growth, and became the country's key foreign financing medium by developing its competitive benefits in foreign exchange trade. In 1994, the Bank was converted into a exclusively state-owned commercial bank. In August 2004, Bank of China Limited was established. The Bank was registered and placed on the Hong Kong Stock Exchange and the Shanghai Stock Exchange in June and July 2006 correspondingly, changing into the first Chinese commercial bank to present an A-Share and H-Share primary public offering and accomplish a dual listing in both markets. In 2016, Bank of China was again chosen as a

Global Systemically Important Bank, thus becoming the only financial institution from developing economies to be chosen as a Global Systemically Important Bank for six successive years.

Market index:

Let's discuss about this in "Insight" section of this document

Let me create the consolidated investment dataset for my organization.

```
library(Quandl)

## Loading required package: xts

## Loading required package: zoo

##
## Attaching package: 'zoo'

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

Quandl.api_key("syeqdyM7PVHT4VNkyX15")
#Fetching the daily data for a year from Quandl
####Crypto-currency
#Dash <-> USD exchange
DAS <- Quandl("BITFINEX/DSHUSD", collapse="daily", start_date="2017-11-
16",end_date="2018-11-15", type="raw")

#ZCash <-> USD exchange
ZAS <- Quandl("BITFINEX/ZECUSD", collapse="daily", start_date="2017-11-
16",end_date="2018-11-15", type="raw")



###Fiat Currencies
#Canadian Dollar <-> Euro
CAEU <- Quandl("ECB/EURCAD", collapse="daily", start_date="2017-11-
16",end_date="2018-11-15", type="raw")

#Euro <-> Chinese Yuan
EUCY <- Quandl("ECB/EURCNY", collapse="daily", start_date="2017-11-
16",end_date="2018-11-15", type="raw")



###Commodities
#Soybean futures, continuous contract
SOY <- Quandl("CHRIS/CME_S1", collapse="daily", start_date="2017-11-
16",end_date="2018-11-15", type="raw")

#E-mini Natural Gas futures, continuous contract
```

```
NG <- Quandl("CHRIS/CME_QG1", collapse="daily", start_date="2017-11-
16",end_date="2018-11-15", type="raw")



###Technology Companies
#CISCO - BATS U.S. Stock Exchange
CIS <- Quandl("BATS/EDGA_CSCO", collapse="daily", start_date="2017-11-
16",end_date="2018-11-15", type="raw")

#Alphabet Inc. - BATS U.S. Stock Exchanges
GOO <- Quandl("BATS/EDGA_GOOG", collapse="daily", start_date="2017-11-
16",end_date="2018-11-15", type="raw")



###Bank
#Royalbank of Canada – Boerse Stuttgart stock exchange
RBC <- Quandl("SSE/RYC", collapse="daily", start_date="2017-11-
16",end_date="2018-11-15", type="raw")

#KBC – EURONEXT exchange
KBC <- Quandl("EURONEXT/KBC", collapse="daily", start_date="2017-11-
16",end_date="2018-11-15", type="raw")



###Market index
#S&P500 - American stock market index
SP <- Quandl("CHRIS/CME_SP1", collapse="daily", start_date="2017-11-
16",end_date="2018-11-15", type="raw")

#STOXX50 - European stock market index
ST <- Quandl("CHRIS/EUREX_FESX1", collapse="daily", start_date="2017-11-
16",end_date="2018-11-15", type="raw")

#Let's take a look at the top 6 columns and structure of each dataset
nam <- list(DAS,ZAS,CAEU,EUCY,SOY,NG,CIS,GOO,RBC,KBC,SP,ST)
for(i in nam){
  head(i)
  str(i)
}

## 'data.frame':    357 obs. of  8 variables:
##  $ Date  : Date, format: "2018-11-15" "2018-11-14" ...
##  $ High  : num  148 167 170 163 165 ...
##  $ Low   : num  131 136 160 159 158 ...
##  $ Mid   : num  141 147 165 161 162 ...
##  $ Last  : num  141 147 164 161 162 ...
##  $ Bid   : num  141 147 164 161 162 ...
##  $ Ask   : num  141 148 165 161 162 ...
##  $ Volume: num  16957 23092 6868 3764 2806 ...
```

```
##  - attr(*, "freq")= chr "daily"
## 'data.frame':    356 obs. of  8 variables:
##  $ Date  : Date, format: "2018-11-15" "2018-11-14" ...
##  $ High  : num  115 132 139 132 140 ...
##  $ Low   : num  101 106 125 128 130 ...
##  $ Mid   : num  112 113 135 130 130 ...
##  $ Last  : num  112 113 135 130 130 ...
##  $ Bid   : num  112 113 135 130 130 ...
##  $ Ask   : num  113 113 136 130 130 ...
##  $ Volume: num  26184 58410 29879 8397 18263 ...
##  - attr(*, "freq")= chr "daily"
## 'data.frame':    255 obs. of  2 variables:
##  $ Date : Date, format: "2018-11-15" "2018-11-14" ...
##  $ Value: num  1.49 1.49 1.49 1.49 1.5 ...
##  - attr(*, "freq")= chr "daily"
## 'data.frame':    255 obs. of  2 variables:
##  $ Date : Date, format: "2018-11-15" "2018-11-14" ...
##  $ Value: num  7.84 7.85 7.84 7.84 7.89 ...
##  - attr(*, "freq")= chr "daily"
## 'data.frame':    252 obs. of  9 variables:
##  $ Date                     : Date, format: "2018-11-15" "2018-11-14" ...
##  $ Open                     : num  884 869 872 874 867 ...
##  $ High                     : num  898 875 876 874 876 ...
##  $ Low                      : num  884 869 867 868 863 ...
##  $ Last                     : num  889 872 867 870 875 ...
##  $ Change                   : num  5.25 3.25 4.5 3.5 7.75 0.25 4.25 1
2.25 6.25 ...
##  $ Settle                   : num  889 870 867 872 875 ...
##  $ Volume                   : num  80828 243 2529 3178 3311 ...
##  $ Previous Day Open Interest: num  302082 411 1488 2301 2582 ...
##  - attr(*, "freq")= chr "daily"
## 'data.frame':    252 obs. of  9 variables:
##  $ Date                     : Date, format: "2018-11-15" "2018-11-14" ...
##  $ Open                     : num  4.67 4.07 3.94 3.73 3.56 ...
##  $ High                     : num  4.79 4.91 4.11 3.96 3.83 ...
##  $ Low                      : num  3.88 4.05 3.9 3.73 3.55 ...
##  $ Last                     : num  3.9 4.68 4.07 3.94 3.72 ...
##  $ Change                   : num  0.799 0.736 0.313 0.069 0.176 0.012 NA
0.012 0.283 0.047 ...
##  $ Settle                   : num  4.04 4.84 4.1 3.79 3.72 ...
##  $ Volume                   : num  10629 15852 4746 2868 2508 ...
##  $ Previous Day Open Interest: num  2372 1862 2188 2300 2088 ...
##  - attr(*, "freq")= chr "daily"
## 'data.frame':    252 obs. of  3 variables:
##  $ Date        : Date, format: "2018-11-15" "2018-11-14" ...
##  $ Short Volume: num  471006 228333 180005 199314 145392 ...
##  $ Total Volume: num  719466 349062 385382 436687 236373 ...
##  - attr(*, "freq")= chr "daily"
## 'data.frame':    252 obs. of  3 variables:
##  $ Date        : Date, format: "2018-11-15" "2018-11-14" ...
```

```
##  $ Short Volume: num  14656 8716 8639 10718 13525 ...
##  $ Total Volume: num  25422 14043 14085 18588 29728 ...
##  - attr(*, "freq")= chr "daily"
## 'data.frame':    253 obs. of  6 variables:
##  $ Date              : Date, format: "2018-11-15" "2018-11-14" ...
##  $ High              : num  63.7 64.1 64.2 64.3 64.4 ...
##  $ Low               : num  63.2 63.4 63.9 64 64.1 ...
##  $ Last              : num  63.6 63.4 63.9 64.2 64.1 ...
##  $ Previous Day Price: num  63.4 63.9 64.2 64.1 64.4 ...
##  $ Volume            : num  0 0 0 0 0 0 0 0 0 0 ...
##  - attr(*, "freq")= chr "daily"
## 'data.frame':    254 obs. of  7 variables:
##  $ Date    : Date, format: "2018-11-15" "2018-11-14" ...
##  $ Open    : num  62.3 60.3 61.9 62.2 60.8 ...
##  $ High    : num  62.8 61.1 61.9 62.3 62.2 ...
##  $ Low     : num  60.8 59.6 60.9 61.6 60.7 ...
##  $ Last    : num  62.3 61.1 61.4 61.7 61.8 ...
##  $ Volume  : num  1182621 1062421 881637 611309 635188 ...
##  $ Turnover: num  73316670 64569369 54126209 37799808 39076937 ...
##  - attr(*, "freq")= chr "daily"
## 'data.frame':    252 obs. of  9 variables:
##  $ Date                     : Date, format: "2018-11-15" "2018-11-14" ...
##  $ Open                     : num  2700 2732 2734 2775 2804 ...
##  $ High                     : num  2736 2748 2755 2795 2811 ...
##  $ Low                      : num  2672 2687 2716 2722 2766 ...
##  $ Last                     : num  2735 2698 2728 2729 2778 ...
##  $ Change                   : num  35.9 29 0.2 51.1 29.8 7.7 57.4 19.4
15.3 13.8 ...
##  $ Settle                   : num  2734 2699 2728 2728 2779 ...
##  $ Volume                   : num  3269 1559 949 1259 3588 ...
##  $ Previous Day Open Interest: num  55843 55215 54956 53997 52715 ...
##  - attr(*, "freq")= chr "daily"
## 'data.frame':    255 obs. of  7 variables:
##  $ Date                   : Date, format: "2018-11-15" "2018-11-14" ...
##  $ Open                   : num  3205 3200 3193 3234 3215 ...
##  $ High                   : num  3221 3233 3221 3240 3231 ...
##  $ Low                    : num  3155 3170 3186 3179 3196 ...
##  $ Settle                 : num  3182 3197 3219 3185 3220 ...
##  $ Volume                 : num  1428755 1375879 1123299 1005702 999196
...
##  $ Prev. Day Open Interest: num  3983938 4014613 3888048 3865011 3874033
...
##  - attr(*, "freq")= chr "daily"
```

Treating missing value and inputing value for weekend using Last-value carried forward methond:

Last-value carried forward method is often used by several data providers and financial institution for imputing missing values in financial time series [2].

```
library(tidyr)
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:xts':
##
##     first, last

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

#function to treat normal missing value and the weekend missing value.
wend <- function (dfa){
dfa <- dfa %>%
        mutate(Date = as.Date(Date)) %>%
        complete(Date = seq.Date(min(Date), max(Date), by="day"))
#Last observation carried out function
        dfa <- na.locf(dfa)}



#We need to remove the spaces from the column names in order to prevent a
conflict that can occur when applied to neural network.
#Function to remove spaces in column name :
spaceless <- function(x) {colnames(x) <- gsub(" ", "_", colnames(x));x}




#Let's combine all 12 datasets into a single consolidated dataframe by
refering the date. Before that, by taking a glance of
#the dataset we can see many of them has same column names in order to avoid
confusion lets prefix their column name
#with dataframe name.

#Let's use below command to rename columns of the dataset inorder to avoid
conflicts between their names.
#names(data) <- c("new_name", "another_new_name")
colnames(DAS)[2:length(DAS)] <-
paste(colnames(DAS[2:length(DAS)]),"DAS",sep="")
colnames(ZAS)[2:length(ZAS)] <-
paste(colnames(ZAS[2:length(ZAS)]),"ZAS",sep="")
```

```
colnames(CAEU)[2:length(CAEU)] <-
paste(colnames(CAEU[2:length(CAEU)]),"CAEU",sep="")
colnames(EUCY)[2:length(EUCY)] <-
paste(colnames(EUCY[2:length(EUCY)]),"EUCY",sep="")
colnames(SOY)[2:length(SOY)] <-
paste(colnames(SOY[2:length(SOY)]),"SOY",sep="")
colnames(NG)[2:length(NG)] <- paste(colnames(NG[2:length(NG)]),"NG",sep="")
colnames(CIS)[2:length(CIS)] <-
paste(colnames(CIS[2:length(CIS)]),"CIS",sep="")
colnames(GOO)[2:length(GOO)] <-
paste(colnames(GOO[2:length(GOO)]),"GOO",sep="")
colnames(RBC)[2:length(RBC)] <-
paste(colnames(RBC[2:length(RBC)]),"RBC",sep="")
colnames(KBC)[2:length(KBC)] <-
paste(colnames(KBC[2:length(KBC)]),"KBC",sep="")
colnames(SP)[2:length(SP)] <- paste(colnames(SP[2:length(SP)]),"SP",sep="")
colnames(ST)[2:length(ST)] <- paste(colnames(ST[2:length(ST)]),"ST",sep="")

#Combining 12 datasets into a single dataframe
TOTdf <- Reduce(function(x, y) merge(x, y, by="Date"),
list(DAS,ZAS,CAEU,EUCY,SOY,NG,CIS,GOO,RBC,KBC,SP,ST))

#Applying both function to all the datasets fetched from Quandl and top 6
columns and structure of each dataset after removing missing values
TOTdf <- wend(TOTdf)
TOTdf <- spaceless(TOTdf)
head(TOTdf)
```

```
## # A tibble: 6 x 62
##   Date       HighDAS LowDAS MidDAS LastDAS BidDAS AskDAS VolumeDAS HighZAS
##   <date>       <dbl>  <dbl>  <dbl>   <dbl>  <dbl>  <dbl>     <dbl>   <dbl>
## 1 2017-11-16    441.   409.   421.    421.   421.   422.    19586.    295
## 2 2017-11-17    427.   409.   421.    420.   420.   421.    20297.    324.
## 3 2017-11-18    427.   409.   421.    420.   420.   421.    20297.    324.
## 4 2017-11-19    427.   409.   421.    420.   420.   421.    20297.    324.
## 5 2017-11-20    455    424.   442.    442.   442.   442.    17345.    304.
## 6 2017-11-21    497.   437    475.    476.   474.   476.    18114.    309.
## # ... with 53 more variables: LowZAS <dbl>, MidZAS <dbl>, LastZAS <dbl>,
## #   BidZAS <dbl>, AskZAS <dbl>, VolumeZAS <dbl>, ValueCAEU <dbl>,
## #   ValueEUCY <dbl>, OpenSOY <dbl>, HighSOY <dbl>, LowSOY <dbl>,
## #   LastSOY <dbl>, ChangeSOY <dbl>, SettleSOY <dbl>, VolumeSOY <dbl>,
## #   Previous_Day_Open_InterestSOY <dbl>, OpenNG <dbl>, HighNG <dbl>,
## #   LowNG <dbl>, LastNG <dbl>, ChangeNG <dbl>, SettleNG <dbl>,
## #   VolumeNG <dbl>, Previous_Day_Open_InterestNG <dbl>,
## #   Short_VolumeCIS <dbl>, Total_VolumeCIS <dbl>, Short_VolumeGOO <dbl>,
## #   Total_VolumeGOO <dbl>, HighRBC <dbl>, LowRBC <dbl>, LastRBC <dbl>,
## #   Previous_Day_PriceRBC <dbl>, VolumeRBC <dbl>, OpenKBC <dbl>,
## #   HighKBC <dbl>, LowKBC <dbl>, LastKBC <dbl>, VolumeKBC <dbl>,
## #   TurnoverKBC <dbl>, OpenSP <dbl>, HighSP <dbl>, LowSP <dbl>,
## #   LastSP <dbl>, ChangeSP <dbl>, SettleSP <dbl>, VolumeSP <dbl>,
```

```
## #   Previous_Day_Open_InterestSP <dbl>, OpenST <dbl>, HighST <dbl>,
## #   LowST <dbl>, SettleST <dbl>, VolumeST <dbl>,
## #   Prev._Day_Open_InterestST <dbl>

str(TOTdf)

## Classes 'tbl_df', 'tbl' and 'data.frame':    365 obs. of  62 variables:
##  $ Date                          : Date, format: "2017-11-16" "2017-11-17"
...
##  $ HighDAS                       : num  441 427 427 427 455 ...
##  $ LowDAS                        : num  409 409 409 409 424 ...
##  $ MidDAS                        : num  421 421 421 421 442 ...
##  $ LastDAS                       : num  421 420 420 420 442 ...
##  $ BidDAS                        : num  421 420 420 420 442 ...
##  $ AskDAS                        : num  422 421 421 421 442 ...
##  $ VolumeDAS                     : num  19586 20297 20297 20297 17345 ...
##  $ HighZAS                       : num  295 324 324 324 304 ...
##  $ LowZAS                        : num  269 285 285 285 294 ...
##  $ MidZAS                        : num  288 295 295 295 300 ...
##  $ LastZAS                       : num  288 294 294 294 300 ...
##  $ BidZAS                        : num  287 294 294 294 300 ...
##  $ AskZAS                        : num  289 295 295 295 300 ...
##  $ VolumeZAS                     : num  41221 88068 88068 88068 19486 ...
##  $ ValueCAEU                     : num  1.5 1.51 1.51 1.51 1.51 ...
##  $ ValueEUCY                     : num  7.81 7.82 7.82 7.82 7.82 ...
##  $ OpenSOY                       : num  976 973 973 973 990 ...
##  $ HighSOY                       : num  978 992 992 992 992 ...
##  $ LowSOY                        : num  970 972 972 972 983 ...
##  $ LastSOY                       : num  972 990 990 990 989 ...
##  $ ChangeSOY                     : num  4.25 18.5 18.5 18.5 0.5 1 8.25 8.25
4 4 ...
##  $ SettleSOY                     : num  972 990 990 990 990 ...
##  $ VolumeSOY                     : num  85723 115931 115931 115931 88069
...
##  $ Previous_Day_Open_InterestSOY: num  323185 327103 327103 327103 319186
...
##  $ OpenNG                        : num  3.09 3.06 3.06 3.06 3.07 ...
##  $ HighNG                        : num  3.11 3.13 3.13 3.13 3.08 ...
##  $ LowNG                         : num  3.04 3.06 3.06 3.06 3.02 ...
##  $ LastNG                        : num  3.07 3.13 3.13 3.13 3.04 ...
##  $ ChangeNG                      : num  0.027 0.044 0.044 0.044 0.05 0.03
0.049 0.049 0.155 0.155 ...
##  $ SettleNG                      : num  3.05 3.1 3.1 3.1 3.05 ...
##  $ VolumeNG                      : num  1036 928 928 928 938 ...
##  $ Previous_Day_Open_InterestNG : num  1219 1260 1260 1260 1349 ...
##  $ Short_VolumeCIS               : num  716540 144234 144234 144234 222817
...
##  $ Total_VolumeCIS               : num  1009369 321193 321193 321193 402545
...
##  $ Short_VolumeGOO               : num  9973 13143 13143 13143 13032 ...
```

```
##  $ Total_VolumeGOO               : num  20168 27828 27828 27828 22268 ...
##  $ HighRBC                       : num  67.1 66.9 66.9 66.9 67.2 ...
##  $ LowRBC                        : num  66.3 66.3 66.3 66.3 66.7 ...
##  $ LastRBC                       : num  67.1 66.9 66.9 66.9 67.2 ...
##  $ Previous_Day_PriceRBC         : num  66.3 67.1 67.1 67.1 66.9 ...
##  $ VolumeRBC                     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ OpenKBC                       : num  70.5 69.2 69.2 69.2 67.1 ...
##  $ HighKBC                       : num  70.5 69.3 69.3 69.3 67.4 ...
##  $ LowKBC                        : num  68.1 67.2 67.2 67.2 66.8 ...
##  $ LastKBC                       : num  69.6 67.4 67.4 67.4 67 ...
##  $ VolumeKBC                     : num  1064491 1482100 1482100 1482100
1045550 ...
##  $ TurnoverKBC                   : num  7.37e+07 1.01e+08 1.01e+08 1.01e+08
7.00e+07 ...
##  $ OpenSP                        : num  2564 2582 2582 2582 2574 ...
##  $ HighSP                        : num  2589 2586 2586 2586 2583 ...
##  $ LowSP                         : num  2562 2575 2575 2575 2568 ...
##  $ LastSP                        : num  2585 2576 2576 2576 2582 ...
##  $ ChangeSP                      : num  19.9 8.7 8.7 8.7 5.8 14.2 1.7 1.7
6.4 6.4 ...
##  $ SettleSP                      : num  2585 2576 2576 2576 2582 ...
##  $ VolumeSP                      : num  2397 2682 2682 2682 2646 ...
##  $ Previous_Day_Open_InterestSP : num  64596 65063 65063 65063 65885 ...
##  $ OpenST                        : num  3552 3563 3563 3563 3526 ...
##  $ HighST                        : num  3575 3575 3575 3575 3564 ...
##  $ LowST                         : num  3549 3537 3537 3537 3523 ...
##  $ SettleST                      : num  3560 3544 3544 3544 3559 ...
##  $ VolumeST                      : num  992582 951137 951137 951137 916183
...
##  $ Prev._Day_Open_InterestST    : num  3819776 3715466 3715466 3715466
3656381 ...
```

Firstly, let us find the required features from my organization's consolidated trending investment dataset that consists of key variables of all currenlty trending investment's dataset and choose financial datasets for those investments. Mostly stock markets are forecasted using ARIMA model but many times ANN outperforms it. So in this study let us discuss which method fits well to the dataset and contrast both method. The main features required to apply both methods are follows for the dataset Soyabean,Natural gas, S&P500 and STOXX50,

1. Open
2. High
3. Low
4. Last
5. Volume
6. Settle

Method:

ARIMA (Autoregressive Integrated Moving Average):

ARIMA (Autoregressive Integrated Moving Average) is a key tool adopted in time series analysis that uses current value of a variable to predict its future values. The application of ARIMA is to identify the features of the connection among our residuals, that would deliver our method with a specific degree of predicting strength. Firstly, to perform a time series analysis we should state our dataset in the form of logarithms. If our data is stated as it is in price terms, then this method does not let for continuous compounding of returns across time and will provide deceptive results.The much common ARIMA procedure model can be inscribed as an AR if the MA process is invertible. Since AR models are modest to evaluate, have well improved model selection criteria, and require less pretesting, they are the form of ARIMA used here [3].

An ARIMA model involves coordinates (p, d, q):

1. p stands for the number of autoregressive terms, i.e. the number of observations from past time values used to forecast future values. For example. if the value of p is 2, then this means that two previous time observations in the series are being used to forecast the future trend.

2. d denotes the number of differences needed to make the time series stationary (i.e. one with a constant mean, variance, and autocorrelation). For instance, if d = 1, then it means that a first-difference of the series must be obtained to transform it into a stationary one.

3. q represents the moving average of the previous forecast errors in our model, or the lagged values of the error term. As an example, if q has a value of 1, then this means that we have one lagged value of the error term in the model.

Autocorrelation and Partial Autocorrelation Plots

We use the ACF and PACF (Autocorrelation and Partial Autocorrelation) interpretations to find whether our data is stationary upon differencing. The autocorrelation and partial autocorrelation function both quantify to varying degrees, the correlation coefficient among a series and lags of the variable thru time. An autoregressive process is when a time series follows a particular pattern in that its present value is in some way correlated to its past value(s). For instance, if we are able to use regression analysis to discern the present value of a variable from using its past value, then we refer to this as an AR(1) process.

Dickey-Fuller Test

For using an ARIMA model, we now like to perform extra formal test to identify if our time series is stationary; i.e. do we have a constant mean, variance and autocorrelation across our time series dataset. To identify this, we adopt the Dickey-Fuller Test. At the 5% level of significance: H0: Non-stationary series HA: Stationary series

auto.arima function:

To produce an ARIMA plot and output by allowing R itself choose the suitable parameters using ARIMA, we use the auto.arima function for defining our time series.

Ljung-Box Test

When we could effectively adopt this method to forecast future values for price, a significant test used to confirm the results of the ARIMA model is the Ljung-Box test. Basically, the test would be used to identify if the residuals of our time series stay o a random pattern, or if there is a significant degree of non-randomness.

H0: Residuals follow a random pattern HA: Residuals do not follow a random pattern

Note that the model for picking a exact number of lags for Ljung-Box can be quite arbitrary. For this purpose, we will perform the Ljung-Box test with lags 5, 10, and 15. To execute this test in R, we would use the function mentioned below, Box.test(df$var, lag=5, type="Ljung-Box")

Artificial Neural Network model:

Artificial neural networks are computational model imitated on the inner structure of the brain. ANN have been mainly utilized for finding financial economic glitches. Distinctive applications in finance include mortgage risk valuation, economic forecast, risk valuation of exchange-traded fixed-income investments, portfolio choice/divergence, mockup of market conduct, index building, and discovery of detailed economic factors. For instance, in 1989, the US government initiated a five-year, multi-million-dollar project for neural network research, but financial services companies have been the major promoters of research in neural network applications [3].

ANN consists of a sum of inter-connected simply processing units called neurons or nodes. Each neuron gets an input indicator from other neurons or outer inputs and then after processing the signals nearby via a shifting function, it gives out a reformed signal to other neurons real outputs. ANNs are processed by the network architecture, that is, the sum of layers, the sum of neurons in each layer and how the neurons are inter-connected. In a familiar form, multi-layer perceptron (MLP), all nodes and layers are formed in a feed-forward manner. The first or the lowest layer is an input layer where external information is received. The last or the outermost layer is an output layer where the network creates the model results. In between, there are one or more than one hidden range which are critical for ANNs to identify the complex patterns in the data. All nodes in next to layers are linked by acyclic arcs from a lower part to a higher part. A multi-layer perceptron with a single hidden part and one output neuron is the mostly used ANN structure for more true applications, and will be adopted in this work. Neural networks are strong predicting tools which can does difficult function mapping. Unlike linear regression method which is restricted to the linear function mappings, neural networks are able to identify difficult nonlinear relations in the data. The issue exists still is whether neural networks can give perfect predictions or a fair forecast to the defined function. This is overcome by the following work in neural network research which exposes that a feed-forward neural

network with only single hidden segment can predict any continuous function with arbitrarily defined accuracy provided adequate hidden neural used. The hopeful results of ANN application to financial data, motivate us to compare and contrast the probability of each variance of model on financial data.

Previously existing methods originate contradictory evidence regarding the persistence and predictability of mutual fund act. This is because these methods adopted linear models as the key tools, they would not be proficient to capture complexity in the data. Additionally, while economic environment influences the fund managers' actions, macroeconomic variables are beneath their control. Given an economic environment, fund managers must consequently focus their attention on fund-specific features that influence performance. From an investor's perspective, taking about 75±80% of professional money managers into an account do not defeat the market index, forecast of mutual fund performance centers on information about fund features that can support him/her pick well performing funds is tremendously worthful. This issue can be addressed by adopting an artificial neural network method. ANN studies which fund-specific features are important performance predictors, taking as provided environment where the fund performs[4].

Why I am preferring ANN over ARIMA model?

In the study "Neural networks for technical analysis: a study on KLCI," [5] the stock predicting accuracy of ANN and ARIMA models are compared and exposed that the ANN model attained more accurate returns than the traditional ARIMA models. Similar to this study, "Time series prediction with genetic-algorithm designed neural networks: an empirical comparison with modern statistical models," [6] contrasted the forecasting accuracy of ANNs and ARIMA on time series prediction to expose that the ANNs outclassed ARIMA in forecasting stock movement track as the latter was able to sense hidden forms in the data used. Empirical outcomes gotten also exposed that the ANN model is greater than the ARIMA model. "Stock price prediction: Comparison of Arima and artificial neural network methods—an Indonesia stock's case," [7] performed same contrast both model on the Indonesia stock exchange and got better accuracy with ANN than the ARIMA model. More literature has exposed the predominant application of ANNs as an effective tool for stock price prediction. This makes ANN a promising technique or potential hybrid for the prediction of movement in time series [8].

Insights

Before applying the method to derive the insights let's first discuss the abount components of insights in detail. ###Commodities: Commodities are the lifeblood of our economic system, that are standardized goods— such as copper, gold, oil, and wheat—which are broadly traded crosswise between national boundaries. For the purposes of investment, commodities are generally broken down into the following divisions: energy commodities (crude oil, natural gas, etc.); agricultural commodities (rice, wheat, soybeans, etc.); industrial/base metals (zinc, nickel, copper, etc.); and precious metals (platinum, gold, palladium, etc.). Because of the storage costs are high most of the commodities cannot be

stored easily. This reason influenced investors to usually seek experience to commodities via derivatives, naturally in the form of futures contracts. Investments are usually executed via index tracking funds, hedge funds, and other active investment strategies [10].

Investments in Commodities:

Most of the commodity investments are made thru index funds. Commodity index investments are classically categorized by a passive tactic devised to obtain exposure to commodity price movements as part of a portfolio diversification tactic. Exposure to commodity price movements can be related to investment in a broad index of commodities, a sub-index of related commodities, or a single-commodity index. Index funds purchase a frontward position, then trade it when it is about to expire and utilize the earnings from this sale to buy forward again by one or more months (a process known as rolling)[10].

SoyBean

A. E. Staley Manufacturing Company commenced to crush soybeans 100 years ago, a significant invention that directed to the making of soy oil which is used in cooking a soy meal and also used for feeding chickens and hogs. Today, the soybean is one of the world's most important legumes and a significant base of protein.

Crush spread is the tactic which is used by most of soybean traders, in which they purchase one contract of soybeans at the same time trades one contract of soybean oil and one contract of soybean meal. This is performed in order to generate a hedge against supply and demand aspects. Crush spread is simply profit margin achieved by breaking soybeans into soyoil[11].

Production

The major portion of soybean crops are assigned for vegetable oil and animal feed. While tofu, soy milk and other soy products have added admiration in current years, only a minor percentage of the crop is used for such foods. The U.S. harvests more soybeans than any other country, followed by Brazil, Argentina, China, India, Paraguay and Canada. Soybeans are the leading oilseed crop in the U.S. about 90% of entire oilseed harvest. In U.S., most soybeans are planted during May and early June and harvested between September and October [10].

Price Dependencies

Soybean prices are affected by the below mentioned factors: • the demand for biodiesel • intensity of the U.S. dollar • real and perceived health hazards • evolving market need • emerging conditions

China presently clutches a foremost spot within the world consumer market for and, in the last ten years, has developed the biggest importer of soybean from U.S. [12].

Natural Gas:

Natural gas is a fossil fuel created from departed plant matter stuck between rock deposits deep underneath the earth's surface. Consisting mostly of methane, natural gas is discovered in conjunction with coal beds and supplementary fossil fuel deposits all over the world. It is used widely all over the U.S. to heat houses and produce electricity and has important purposes in both commercial and industrial settings [10].

Price affecting factors

Minor variations in supply or demand can outcome in significant price changes which eventually fetch supply and demand back into equilibrium. Factors that influences supply and demand, and hence price, include:

• Changes in natural gas making

• the size of imports and exports

• storage stages • monetary development

• changes in winter and summer weather

• prices of rival fuels

Investing in natural gas sounds like a winner: demand growing constantly, fracking ensures to encounter that demand, and prices have moved in the past numerous years. In the meantime it's hyped as an substitute to coal, and to a slighter range, oil. There's a lot to like. But financing in commodities on a futures market is still a complex commercial for maximum retail stockholders. Taxes and expenses can be complexed to solve. Hence the exchange-traded savings can be a better vehicle [10].

What is a 'Market Index'?

A market index is a calculated average of numerous stocks or further investment vehicles from a unit of the stock market, and it is evaluated from the price of the chosen stocks. Market indexes are proposed to signify a complete stock market and follow the market's variations over long period of time. For instance, the broadly used Standard and Poor's 500 (S&P500) Index is calculated by uniting 500 large-cap U.S. stocks into a index value. Investors can follow variations in the index's value over a period of time and use it as a scale for their own portfolio returns.

BREAKING DOWN 'Market Index'

Market indices calculate the worth of a set of stocks. If an index increases by one level, or 1%, that defines a set of stocks has, harmoniously, increases its worth by one level and also attract more investors.

In 1980's launch of stock index futures contracts is believed as a momentous creation. Even though there are some disagreements with index arbitrage, the program trading that established after October stock market crash in 1987, these contracts became very successful and valuable to the portfolio managers. Stock index futures are being used as a major trading tool by the institutional investors[13].

S&P 500:

The Standard & Poor's 500, usually shortened as the S&P 500, or just the S&P, is an American stock market index constructed on the market capitalizations of 500 big organizations containing common stock listed on the NYSE or NASDAQ. How do S&P 500 futures work? S&P 500 futures are a kind of capital asset contract that delivers a trader the claim to a predefined choice of stocks and on a predefined forthcoming date recorded on the S&P 500 stock market index. There are various dimensions of stock baskets for the S&P 500; the Chicago Mercantile Exchange, or CME, provides a "big contract" and an "e-mini" contract. The big futures contract was firstly valued by multiplying the determined futures price by $500. For instance, if the S&P was selling at $800, the worth of the big contract was $400,000, or $500 x $800. Ultimately, the CME reduce the contract multiplier in half to $250 times the price of the futures index. E-mini futures are one-tenth the worth of the big contract. If the S&P 500 futures price is $800, this outcomes in an e-mini being valued at $40,000. The "e" in e-mini stands for electronic [10].

What is the 'Euro STOXX 50 Index'?

The Euro STOXX 50 Index is a market capitalization calculated stock index of 50 big, blue-chip European companies functioning within Eurozone countries. Components are chosen from the Euro STOXX Index which contains big-, mid- and small-cap stocks in the Eurozone.

BREAKING DOWN 'Euro STOXX 50 Index'

The Euro STOXX 50 Index is governed and authorized by STOXX Limited which gives indexes signifying equity market investments over the entire world. The Euro STOXX 50 Index consists of Eurozone's 50 biggest organizations by market cap.

STOXX 50:

STOXX Limited is held by Deutsche Börse AG. It has been governing and authorizing indexes since 1998. The Euro STOXX 50 Index was among the first STOXX indexes commenced in 1998. The company has widened its contributions significantly since it's begun which aimed on European stock indexes. It now provides indexes representing nearly all countries and area of the world. Asset classes include equity, static pay and currency. It also offers indexes by division, aspect, approach and theme. The Euro STOXX Index contains organizations of entire market capitalization stages from Austria, Belgium, Finland, France, Germany, Ireland, Italy, Luxembourg, the Netherlands, Portugal and Spain.

The Euro STOXX 50 Index naturally signifies around 60% of the Euro STOXX Index. The Euro STOXX 50 Index is revised yearly in September for any index component variations.

As of January 8, 2018, the top ten components in the Euro STOXX 50 Index comprised the following:

1. TOTAL
2. SIEMENS
3. SAP
4. BCO SANTANDER
5. BAYER
6. ALLIANZ
7. BASF
8. SANOFI
9. UNILEVER 10.BNP PARIBAS [10]

On 25 May 2010 the S&P 500 Dividend Index and choices on the S&P500 Annual Dividend Index were presented on CBOE, whilst choices on Euro STOXX50 Index Dividend Futures and Euro STOXX50 Index Dividend Points (DVP) were also announced on Eurex. The futures contract on the Dow Jones Euro STOXX50 DVP index announced on 30 June 2008 by Eurex has faced a impressive growth. This is barely astonishing so far then reinvested dividends registered for nearly half of the Dow Jones Euro STOXX50 entire yields since the last week of December 1991 [14].

Mutual fund:

A mutual fund is an investment vehicle crafted with a assortment of money accumulated from various investors for the determination of investing in safeties such as stocks, bonds, money market instruments and other assets. Mutual funds are functioned by qualified money directors, who deal the fund's investments and try to yield capital profits and/or revenue for the fund's investors. A mutual fund's portfolio is organized and governed to tie the investment aims specified in its brochure.

Types of Mutual Funds

Mutual funds are classified into various types of divisions, signifying the types of securities the mutual fund director invests in. One of the majors is the fixed income type. A fixed income mutual fund aims on investments that pay a static rate of return, such as government bonds, corporate bonds or other debt instruments. The knowledge is the fund portfolio produces a higher amount of interest income, which can then be transfered on to stockholders.

One more group comes under the moniker "index funds." The investment plan is based on the trust that it is very tough, and often costly, to attempt to constantly overcome the market. So the index fund manager just purchases stock that match with a key market index such as the S&P 500 or the Dow Jones Industrial Average. This plan needs fewer investigation from analysts and advisors, so there are lesser expenditures to eat up yields before they are transferred on to shareholders. These funds are usually devised using cost-sensitive investors in account.

Balanced funds capitalize in both stocks and bonds with the focus of lowering risk of revelation to one asset class or another. alternative name of this type is “asset allocation fund.” An investor may about to identify the provision of these funds between asset classes comparatively invariable, though it will vary between funds. Though their aim is asset appreciation with reduced risk, these funds hold the similar risk and are as subject to variation as other division of funds.

Other usual types of mutual funds are money market funds, sector funds, equity funds, alternative funds, smart-beta funds, target-date funds and even funds-of-funds, or mutual funds that buy shares of other mutual funds [10].

Advantages of Mutual Funds

1. Diversification
2. Economies of Scale
3. Easy Access
4. Professional Management
5. Individual-Oriented
6. Investors have the independence to study and choose from managers with a difference of styles and management aims.

Exchange Traded Funds:

An ETF, or exchange-traded fund, is a merchantable security that follows a stock index, a commodity, bonds, or a basket of assets. Even though ETF’s are same in various behaviors, ETFs vary from mutual funds due to shares trade like usual stock on an exchange. The price of an ETF’s shares will vary all over the day as they are bought and sold. The biggest ETFs characteristically have greater mean day-to-day volume and lesser charges than mutual fund shares which does ETF’s an striking substitute for distinct investors. While almost all ETFs follow stock indexes, there are also ETFs which invest in commodity markets, currencies, bonds, and other asset classes. Several ETFs also hold choice exists for investors to utilize income, speculation, or hedging strategies [10].

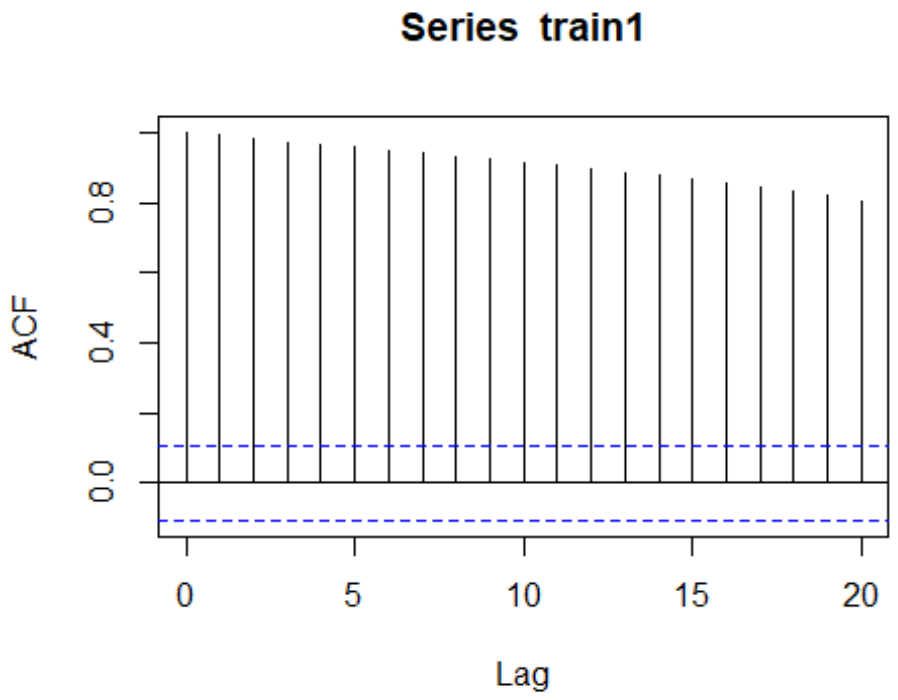
Example:

Euro STOXX 50 Index Funds

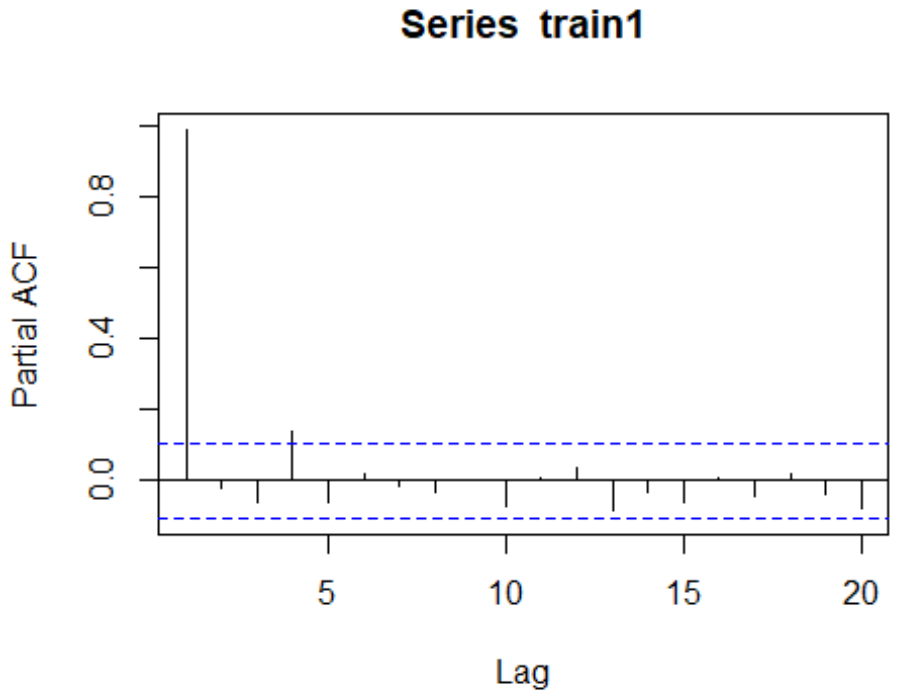
The Euro STOXX 50 Index is a foremost market index for investors looking to track the Eurozone’s biggest equity stock investments. Almost all passive index funds in the investment market that follows the Euro STOXX 50 Index are exchange-traded funds (ETFs). One of the biggest and most famous for investors is the SPDR Euro STOXX 50 ETF (FEZ). The SPDR Euro STOXX 50 ETF has $4.34 billion in assets under management. The ETF has been available to investors since October 2002 [10].

Applying ARIMA method

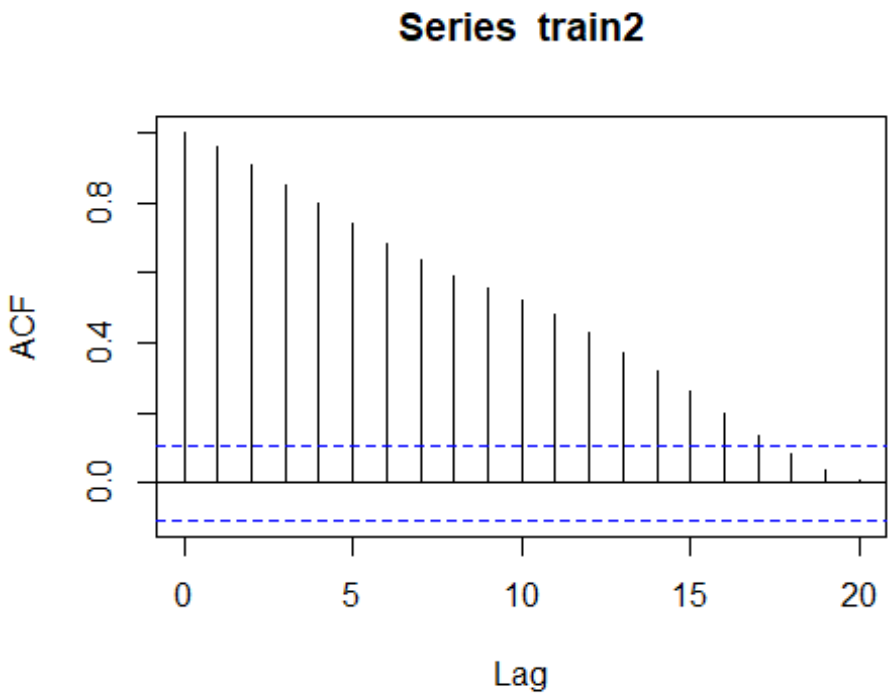
```
#ARIMA Model
#Loading required libraries
library(MASS)

##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select

library(tseries)
library(forecast)
library(ggplot2)
#I am going to use logarithm of first 339 variables as my training data to
perform time series -
#analysis. The reason for conversing to log because stock based on returns
and returns are based -
#on percentages so converting in order to fit in the time series

train1 <- log(TOTdf$SettleSOY[1:339])
train2 <- log(TOTdf$SettleNG[1:339])
train3 <- log(TOTdf$SettleSP[1:339])
train4 <- log(TOTdf$SettleST[1:339])
#Correllelogram
# Autocorrelation and Partial Autocorrelation Plots
acf(train1, lag.max=20)
```



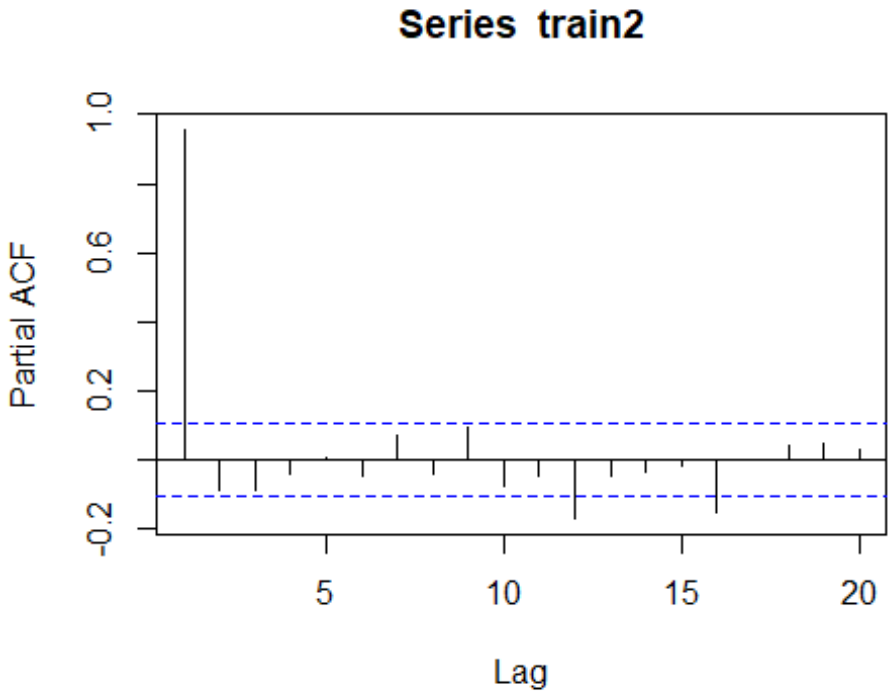

```
pacf(train1, lag.max=20)
```



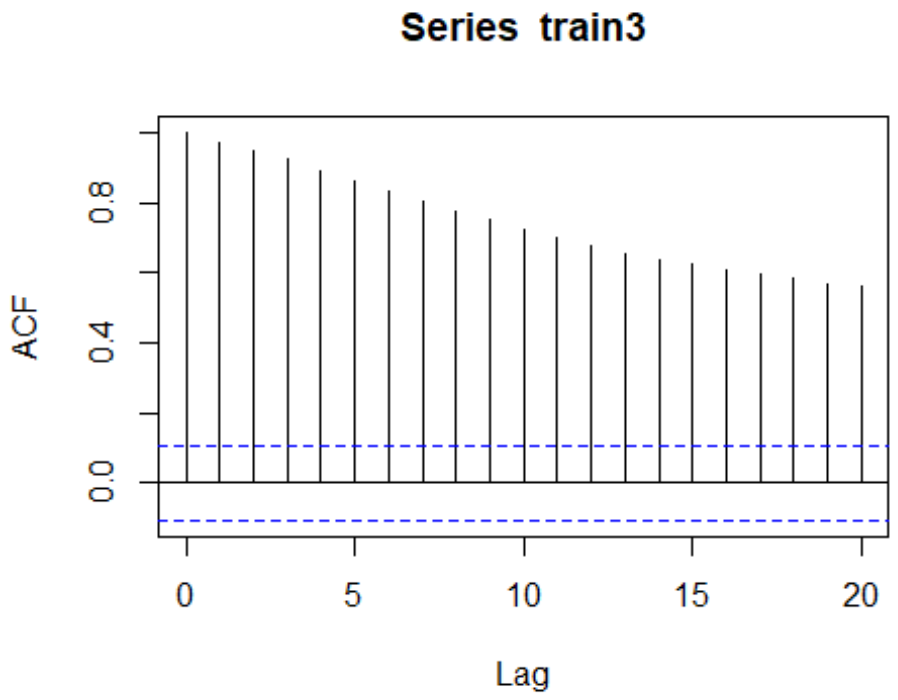

```
acf(train2, lag.max=20)
```



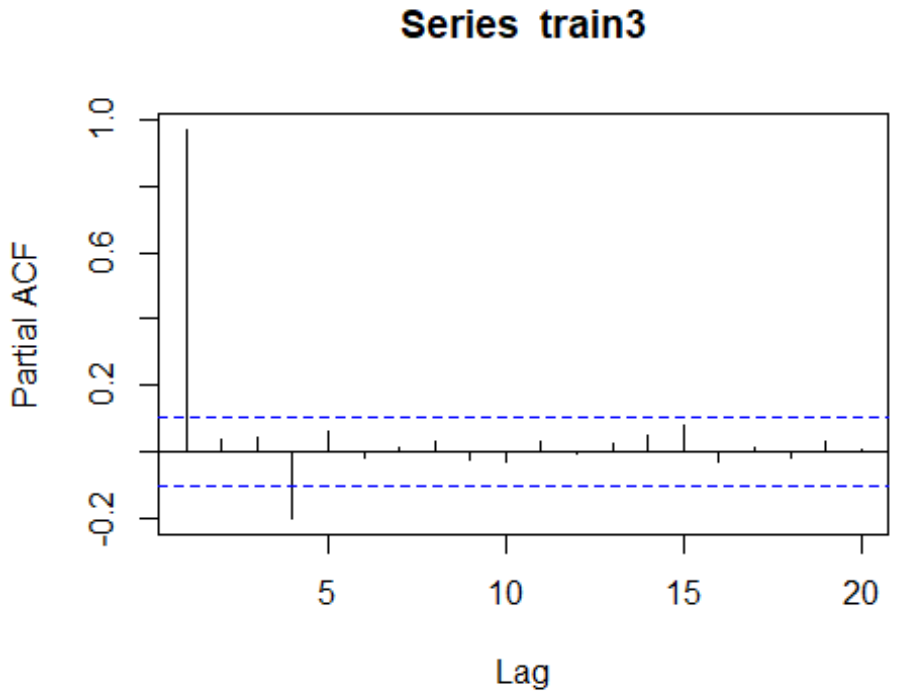

```
pacf(train2, lag.max=20)
```



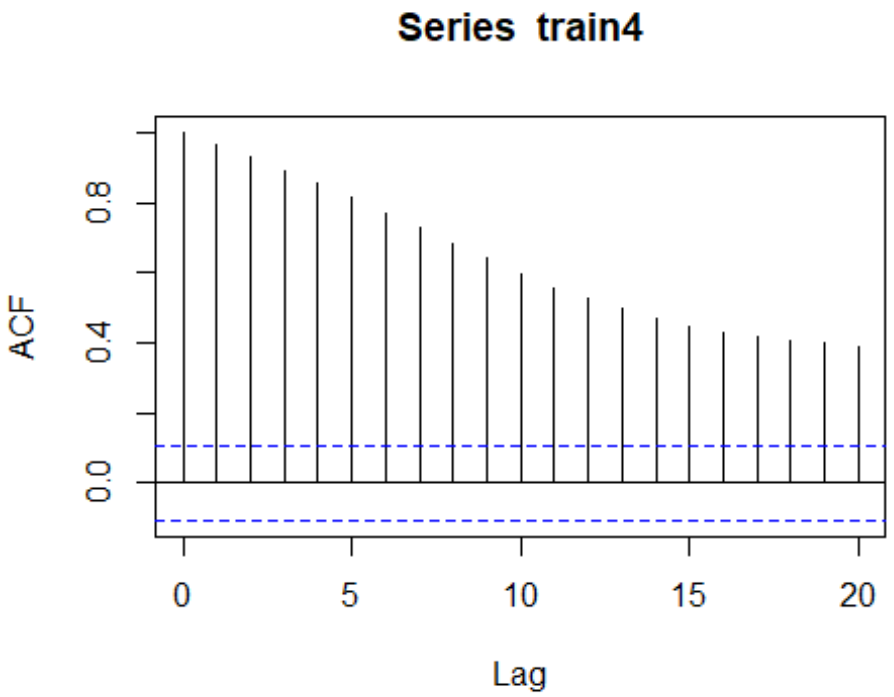

```
acf(train3, lag.max=20)
```



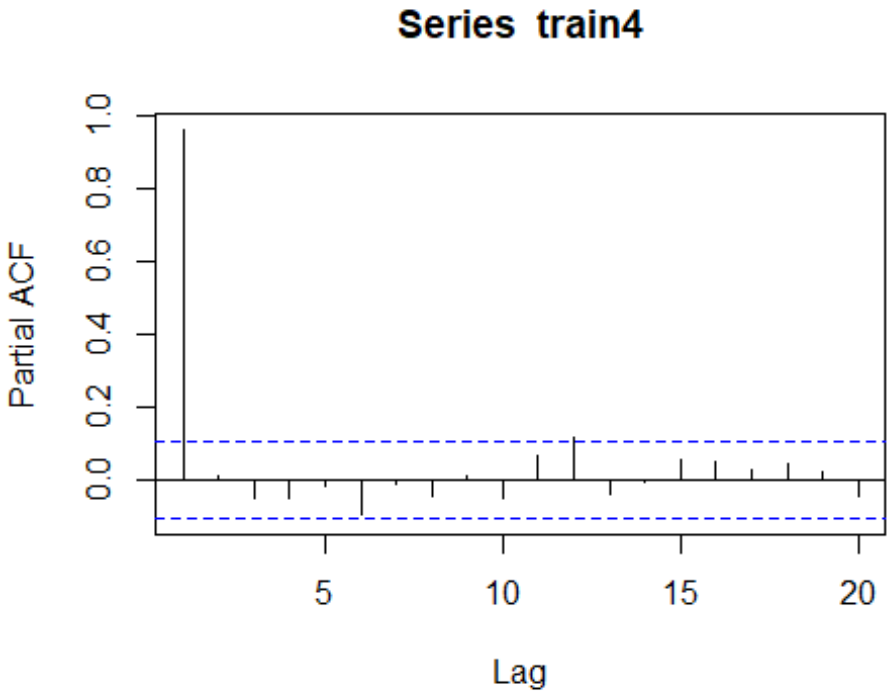

```
pacf(train3, lag.max=20)
```




```
acf(train4, lag.max=20)
```




```
pacf(train4, lag.max=20)
```



```r
#Moreover, we have confirmation that all of our data follows an AR(1)
stationary process -
#(one with a constant mean, variance, and autocorrelation), and we see that
the Settle plot-
#now shows a stationary process.
difftrain1 <- diff(train1,1)
difftrain2 <- diff(train2,1)
difftrain3 <- diff(train3,1)
difftrain4 <- diff(train4,1)
#Perform differencing once (linear trend)

#we can actually have a quick look at your data by using the basic plotting
function
difftrain1
```

```
##   [1]  0.0188540617  0.0000000000  0.0000000000 -0.0005049230 -
0.0010106115
##   [6]  0.0083071592  0.0000000000 -0.0040190961  0.0000000000
0.0000000000
##  [11]  0.0027648629 -0.0030165935 -0.0005036515 -0.0068242398
0.0085859113
##  [16]  0.0000000000  0.0000000000  0.0042654688  0.0099652045 -
0.0057178527
##  [21] -0.0107783974 -0.0022707213  0.0000000000  0.0000000000 -
0.0073520422
##  [26] -0.0068939377  0.0000000000 -0.0082326166 -0.0005167959
0.0000000000
##  [31]  0.0000000000 -0.0059624286 -0.0057366520 -0.0020942416 -
0.0055183427
##  [36]  0.0007902015  0.0000000000  0.0000000000  0.0000000000
0.0000000000
##  [41]  0.0062992334 -0.0102565002  0.0063241317  0.0000000000
0.0000000000
##  [46]  0.0000000000  0.0034089452  0.0049614934 -0.0005211048
0.0023428359
##  [51]  0.0000000000  0.0000000000 -0.0033858608 -0.0026123317
0.0000000000
##  [56] -0.0160847238  0.0037145174  0.0000000000  0.0000000000
0.0000000000
##  [61]  0.0251059211  0.0007744934  0.0043775015  0.0043584225
0.0000000000
##  [66]  0.0000000000  0.0071374252  0.0020299423  0.0060652195
0.0000000000
##  [71] -0.0068259651  0.0000000000  0.0000000000  0.0060698214
0.0087862998
##  [76] -0.0045090257 -0.0108545809 -0.0063653939  0.0000000000
0.0000000000
##  [81] -0.0092379410  0.0168715658 -0.0033007520  0.0048205091 -
0.0048205091
##  [86]  0.0000000000  0.0000000000  0.0188946294  0.0099330342
```

```
0.0054214031
##  [91]  0.0068577295 -0.0026885022  0.0000000000  0.0000000000
0.0000000000
##  [96]  0.0048828222  0.0075215689 -0.0021778593  0.0041097604
0.0000000000
## [101]  0.0000000000 -0.0019319011  0.0036192584  0.0067211007
0.0118907465
## [106]  0.0030685732  0.0000000000  0.0000000000  0.0056404380 -
0.0021114377
## [111] -0.0089644384 -0.0011855366 -0.0235248487  0.0000000000
0.0000000000
## [116]  0.0038787927  0.0069921924 -0.0115999367  0.0118401770
0.0083722532
## [121]  0.0000000000  0.0000000000 -0.0260632513  0.0056077192
0.0014577262
## [126]  0.0000000000 -0.0014577262  0.0000000000  0.0000000000 -
0.0026780296
## [131] -0.0058679875 -0.0014723929  0.0259377042  0.0000000000
0.0000000000
## [136]  0.0000000000  0.0000000000 -0.0064818376 -0.0221608972
0.0156367711
## [141]  0.0024213087  0.0000000000  0.0000000000  0.0127359645
0.0028612323
## [146] -0.0021451563  0.0123311973 -0.0061465915  0.0000000000
0.0000000000
## [151] -0.0116876703  0.0038314223 -0.0040713743 -0.0043290111 -
0.0082285072
## [156]  0.0000000000  0.0000000000 -0.0078068218  0.0014684290
0.0051225871
## [161]  0.0004864996  0.0164017184  0.0000000000  0.0000000000 -
0.0069619775
## [166]  0.0000000000 -0.0048297607  0.0101156932 -0.0154555117
0.0000000000
## [171]  0.0000000000 -0.0246378562  0.0089397169 -0.0039633443
0.0059391414
## [176] -0.0184268162  0.0000000000  0.0000000000  0.0181800549
0.0056601603
## [181] -0.0188264168 -0.0047625106  0.0035114157  0.0000000000
0.0000000000
## [186]  0.0000000000  0.0315452475  0.0084551778 -0.0033734972
0.0055361797
## [191]  0.0000000000  0.0000000000  0.0000000000 -0.0106178604 -
0.0073046344
## [196] -0.0044085303  0.0026964105  0.0000000000  0.0000000000 -
0.0192788967
## [201] -0.0004992511 -0.0070158143 -0.0203207399 -0.0051453677
0.0000000000
## [206]  0.0000000000 -0.0161209940  0.0002620888 -0.0190481950 -
0.0093922601
## [211] -0.0237359390  0.0000000000  0.0000000000  0.0033076105 -
```

```
0.0216976523
## [216]  0.0005622716 -0.0101695792  0.0157749750  0.0000000000
0.0000000000
## [221] -0.0226126085 -0.0083250086  0.0002882260 -0.0072306895 -
0.0031981420
## [226]  0.0000000000  0.0000000000 -0.0117165957 -0.0053175901
0.0000000000
## [231] -0.0101221466  0.0450500276  0.0000000000  0.0000000000 -
0.0257873191
## [236]  0.0005868545 -0.0267554604  0.0003012502 -0.0194653348
0.0000000000
## [241]  0.0000000000  0.0188627436  0.0119833669  0.0032704058
0.0044424773
## [246]  0.0044228290  0.0000000000  0.0000000000 -0.0023564076
0.0120183185
## [251]  0.0032000027  0.0005807201  0.0106829369  0.0000000000
0.0000000000
## [256]  0.0051561271  0.0323288871 -0.0189896798 -0.0048043048
0.0042402890
## [261]  0.0000000000  0.0000000000 -0.0102070634  0.0135863964
0.0050476832
## [266] -0.0067359222 -0.0481708122  0.0000000000  0.0000000000
0.0088261826
## [271]  0.0128057629  0.0000000000  0.0240011521 -0.0045274554
0.0000000000
## [276]  0.0000000000  0.0002835673 -0.0082562747 -0.0187568621 -
0.0191154181
## [281]  0.0000000000  0.0000000000  0.0000000000 -0.0086477354 -
0.0172183683
## [286]  0.0033460107 -0.0045655231  0.0163392440  0.0000000000
0.0000000000
## [291]  0.0000000000 -0.0012012013 -0.0078431775  0.0015130885
0.0063300890
## [296]  0.0000000000  0.0000000000  0.0018012614 -0.0160245303
0.0106109833
## [301] -0.0075677673 -0.0015204503  0.0000000000  0.0000000000
0.0024316121
## [306] -0.0116031836  0.0194653348  0.0241047231 -0.0035346134
0.0000000000
## [311]  0.0000000000 -0.0074041505  0.0056321477  0.0050125418
0.0058651195
## [316] -0.0111733006  0.0000000000  0.0000000000  0.0143845134
0.0095722268
## [321]  0.0000000000 -0.0078249928  0.0112832095  0.0000000000
0.0000000000
## [326]  0.0008626888 -0.0077911229 -0.0125347801  0.0070155213
0.0107200795
## [331]  0.0000000000  0.0000000000  0.0272899255 -0.0076003181
0.0011296245
## [336] -0.0254408462  0.0000000000  0.0000000000
```

```
difftrain2

##   [1]  0.0143091872  0.0000000000  0.0000000000 -0.0162764010 -
0.0098945398
##   [6] -0.0163746349  0.0000000000 -0.0536367958  0.0000000000
0.0000000000
##  [11]  0.0400680666  0.0660742266  0.0161728592 -0.0496555904
0.0118305685
##  [16]  0.0000000000  0.0000000000 -0.0251419131 -0.0240730391
0.0027416056
##  [21] -0.0559512674  0.0032520354  0.0000000000  0.0000000000
0.0200006667
##  [26] -0.0544995008  0.0000000000  0.0022379718 -0.0271920077
0.0000000000
##  [31]  0.0000000000  0.0496648635 -0.0194966632 -0.0206425044 -
0.0148999891
##  [36]  0.0262123270  0.0000000000  0.0000000000  0.0000000000
0.0000000000
##  [41]  0.0240796965  0.0644927661  0.0132948950  0.0000000000
0.0000000000
##  [46]  0.0000000000  0.0342852685 -0.0158314652 -0.0434851119 -
0.0299581875
##  [51]  0.0000000000  0.0000000000  0.0142098306  0.0305685489
0.0000000000
##  [56]  0.0536169696  0.0369233541  0.0000000000  0.0000000000
0.0000000000
##  [61] -0.0224373451  0.0323876760 -0.0133937526 -0.0012550990
0.0000000000
##  [66]  0.0000000000  0.0121705356  0.0660107619  0.0186975100 -
0.0178268091
##  [71]  0.0166862330  0.0000000000  0.0000000000 -0.1014057529
0.0088023203
##  [76] -0.0646428563 -0.0475221871 -0.0035075448  0.0000000000
0.0000000000
##  [81] -0.0354050926  0.0043588881 -0.0208760557 -0.0018521954 -
0.0428014582
##  [86]  0.0000000000  0.0000000000 -0.0124612204  0.0163237204 -
0.0027021827
##  [91] -0.0027095043 -0.0085636958  0.0000000000  0.0000000000
0.0000000000
##  [96]  0.0224207304  0.0163036787 -0.0094465089 -0.0034227073
0.0000000000
## [101]  0.0000000000  0.0229722021 -0.0011175267 -0.0059813262
0.0115565126
## [106] -0.0011125534  0.0000000000  0.0000000000  0.0033339538
0.0165050510
## [111]  0.0101339991 -0.0075908552 -0.0087464114  0.0000000000
0.0000000000
## [116]  0.0166973026  0.0028756310 -0.0199390329 -0.0184779832
0.0026075634
```

```
## [121]  0.0000000000  0.0000000000 -0.0138604954  0.0090124530 -
0.0139283261
## [126] -0.0079924307 -0.0099847220  0.0000000000  0.0000000000
0.0103667659
## [131]  0.0360128939 -0.0059128036  0.0128891492  0.0000000000
0.0000000000
## [136]  0.0000000000  0.0000000000 -0.0132598628  0.0077562716 -
0.0159469354
## [141]  0.0096726945  0.0000000000  0.0000000000 -0.0029662610 -
0.0138345822
## [146]  0.0071281487  0.0041037178  0.0180783378  0.0000000000
0.0000000000
## [151]  0.0061964842 -0.0051001932  0.0003651634 -0.0292667675
0.0292667675
## [156]  0.0000000000  0.0000000000  0.0003650301  0.0148526549
0.0017963001
## [161]  0.0188450021 -0.0242436116  0.0000000000  0.0000000000 -
0.0028912199
## [166]  0.0000000000 -0.0032626456 -0.0102190670 -0.0055177628
0.0000000000
## [171]  0.0000000000  0.0110052468 -0.0032888756  0.0018284883
0.0277445286
## [176] -0.0028469770  0.0000000000  0.0000000000  0.0127480480 -
0.0021134210
## [181] -0.0074323471  0.0155096517 -0.0042061050  0.0000000000
0.0000000000
## [186]  0.0000000000  0.0211997514  0.0020611481  0.0088828736 -
0.0003401939
## [191]  0.0000000000  0.0000000000  0.0000000000 -0.0123247024 -
0.0062197851
## [196]  0.0229580068  0.0033818091  0.0000000000  0.0000000000 -
0.0108622928
## [201] -0.0137459209  0.0020739724  0.0116719485 -0.0137459209
0.0000000000
## [206]  0.0000000000  0.0202096277 -0.0033967424  0.0081328812
0.0006747639
## [211]  0.0190418314  0.0000000000  0.0000000000 -0.0237747679 -
0.0174333590
## [216]  0.0218289704  0.0037043316 -0.0101352219  0.0000000000
0.0000000000
## [221] -0.0074983310  0.0054589013  0.0141894273 -0.0138492333 -
0.0054570395
## [226]  0.0000000000  0.0000000000 -0.0214318608  0.0027913486
0.0000000000
## [231] -0.0115648739  0.0073749237  0.0000000000  0.0000000000 -
0.0105523315
## [236] -0.0142452551  0.0145987994 -0.0113758781 -0.0162194941
0.0000000000
## [241]  0.0000000000  0.0025403752 -0.0069103748 -0.0069584606
0.0174867844
```

```
## [246] -0.0043431121  0.0000000000  0.0000000000 -0.0131436722
0.0040344819
## [251]  0.0156168055  0.0018001805  0.0007191658  0.0000000000
0.0000000000
## [256]  0.0053773207 -0.0053773207 -0.0086643141  0.0208116589
0.0130536339
## [261]  0.0000000000  0.0000000000  0.0024505526  0.0128540940
0.0177904110
## [266]  0.0020325210 -0.0037294500  0.0000000000  0.0000000000 -
0.0047667778
## [271]  0.0098489504  0.0000000000 -0.0173858137  0.0129827584
0.0000000000
## [276]  0.0000000000 -0.0016986585  0.0131736409 -0.0080862974
0.0027027043
## [281] -0.0159840165  0.0000000000  0.0000000000 -0.0141552510 -
0.0083799373
## [286]  0.0038495236  0.0038347615  0.0145080265  0.0000000000
0.0000000000
## [291]  0.0000000000 -0.0324126649 -0.0099680427 -0.0082630253
0.0014419613
## [296]  0.0000000000  0.0000000000  0.0100359265  0.0085227789
0.0003535443
## [301] -0.0042508034 -0.0179087881  0.0000000000  0.0000000000
0.0168432579
## [306]  0.0414188313 -0.0085602303  0.0231145573  0.0003359651
0.0000000000
## [311]  0.0000000000  0.0202833221  0.0143793327 -0.0336554364
0.0251835708
## [316] -0.0158314652  0.0000000000  0.0000000000  0.0281893461
0.0230042093
## [321]  0.0000000000 -0.0003159059 -0.0069752978  0.0000000000
0.0000000000
## [326]  0.0386943748 -0.0003061381  0.0054961970 -0.0190599068 -
0.0191138515
## [331]  0.0000000000  0.0000000000  0.0253019901 -0.0009257831
0.0247001427
## [336] -0.0374391685  0.0000000000  0.0000000000

difftrain3

##   [1] -3.371247e-03  0.000000e+00  0.000000e+00  2.248760e-03  5.484333e-
03
##   [6] -6.549924e-04  0.000000e+00  2.463624e-03  0.000000e+00
0.000000e+00
##  [11]  3.075267e-04  9.258263e-03 -3.427853e-04  8.647882e-03 -1.511773e-
03
##  [16]  0.000000e+00  0.000000e+00 -2.158233e-03 -3.759617e-03  3.423680e-
04
##  [21]  3.909888e-03  4.347422e-03  0.000000e+00  0.000000e+00  3.990518e-
03
```

```
##  [26]  1.314135e-03  0.000000e+00 -4.475122e-03  5.944905e-03
0.000000e+00
##  [31]  0.000000e+00  9.464181e-03 -3.830069e-03 -9.318449e-04  2.309384e-
03
##  [36] -6.326876e-04  0.000000e+00  0.000000e+00  0.000000e+00
0.000000e+00
##  [41] -2.606348e-04  1.117090e-04 -3.618259e-03  0.000000e+00
0.000000e+00
##  [46]  0.000000e+00  6.295538e-03  6.662003e-03  4.747281e-03  6.878411e-
03
##  [51]  0.000000e+00  0.000000e+00  1.530222e-03  2.000328e-03
0.000000e+00
##  [56]  6.229873e-03  6.908739e-03  0.000000e+00  0.000000e+00
0.000000e+00
##  [61] -2.189795e-03  7.554231e-03 -2.678621e-03  5.278939e-03
0.000000e+00
##  [66]  0.000000e+00  8.572190e-03  1.550716e-03  4.929057e-04  7.039527e-
05
##  [71]  1.165225e-02  0.000000e+00  0.000000e+00 -7.297391e-03 -1.025000e-
02
##  [76]  4.601526e-04 -1.203924e-03 -2.355327e-02  0.000000e+00
0.000000e+00
##  [81] -5.556571e-02  3.266982e-02 -9.734343e-03 -2.843433e-02  9.860991e-
03
##  [86]  0.000000e+00  0.000000e+00  1.368923e-02  2.445035e-03  1.313795e-
02
##  [91]  1.362620e-02  4.022747e-04  0.000000e+00  0.000000e+00
0.000000e+00
##  [96] -7.671029e-03 -5.690231e-03  4.731812e-03  1.366247e-02
0.000000e+00
## [101]  0.000000e+00  1.286796e-02 -1.334100e-02 -1.212047e-02 -1.342601e-
02
## [106]  4.470613e-03  0.000000e+00  0.000000e+00  1.042793e-02  2.021205e-
03
## [111] -2.202967e-04  5.858033e-03  1.615041e-02  0.000000e+00
0.000000e+00
## [116]  7.183908e-05 -5.619004e-03 -6.813842e-03  6.907459e-04  8.358424e-
05
## [121]  0.000000e+00  0.000000e+00 -1.060538e-02  2.937720e-04 -1.947850e-
03
## [126] -2.797855e-02 -1.736320e-02  0.000000e+00  0.000000e+00  2.347320e-
02
## [131] -1.660639e-02 -3.101490e-03  1.348438e-02  0.000000e+00
0.000000e+00
## [136]  0.000000e+00  0.000000e+00 -1.133911e-02  1.285140e-02  5.575662e-
03
## [141] -2.130123e-02  0.000000e+00  0.000000e+00  5.129394e-03  1.365157e-
02
## [146] -5.286822e-03  8.633256e-03 -2.518183e-03  0.000000e+00
0.000000e+00
```

```
## [151]  9.140352e-03  9.242313e-03  1.144693e-03 -6.144981e-03 -8.015403e-
03
## [156]  0.000000e+00  0.000000e+00 -1.122986e-04 -1.356822e-02  3.484984e-
03
## [161]  1.131782e-02 -1.197157e-03  0.000000e+00  0.000000e+00 -9.175756e-
03
## [166]  0.000000e+00 -7.432159e-03  1.673259e-03  1.182284e-02
0.000000e+00
## [171]  0.000000e+00  2.587615e-03  1.123532e-04  9.578367e-03  8.384632e-
03
## [176]  3.927980e-03  0.000000e+00  0.000000e+00  5.494204e-04 -8.051662e-
03
## [181]  5.117926e-03 -1.543664e-03 -2.098791e-03  0.000000e+00
0.000000e+00
## [186]  0.000000e+00  4.816983e-03  1.685970e-03 -1.135885e-03 -3.415417e-
03
## [191]  0.000000e+00  0.000000e+00  0.000000e+00 -9.647982e-03  1.196293e-
02
## [196] -7.034890e-03  1.040584e-02  0.000000e+00  0.000000e+00  4.270625e-
03
## [201]  2.219353e-03  7.458668e-03  0.000000e+00  2.377951e-03
0.000000e+00
## [206]  0.000000e+00  1.510303e-03  3.233397e-04 -3.454359e-03  3.202882e-
03
## [211] -2.284153e-03  0.000000e+00  0.000000e+00  1.025816e-03 -4.868471e-
03
## [216]  2.094544e-03 -7.059492e-03  2.539914e-03  0.000000e+00
0.000000e+00
## [221] -1.357239e-02  2.274896e-03 -8.687065e-03  5.383097e-03  7.719028e-
04
## [226]  0.000000e+00  0.000000e+00  2.092166e-03 -5.146503e-03
0.000000e+00
## [231]  9.281235e-03  8.834012e-03  0.000000e+00  0.000000e+00  8.828409e-
03
## [236]  3.330889e-03 -8.113780e-03  8.757189e-03  1.678062e-03
0.000000e+00
## [241]  0.000000e+00 -2.392986e-03  5.278375e-03  1.670428e-03 -3.806953e-
03
## [246] -1.605394e-03  0.000000e+00  0.000000e+00  3.990883e-03  3.195458e-
03
## [251]  7.170262e-03  4.222527e-04 -8.833980e-03  0.000000e+00
0.000000e+00
## [256] -5.088345e-03  4.946365e-03 -2.238847e-03  6.242040e-03  3.916800e-
03
## [261]  0.000000e+00  0.000000e+00  3.726101e-03  3.362643e-03 -1.539808e-
03
## [266] -5.254769e-04 -5.974783e-03  0.000000e+00  0.000000e+00 -3.991316e-
03
## [271]  5.470763e-03  0.000000e+00  1.301511e-03  2.633012e-03
0.000000e+00
```

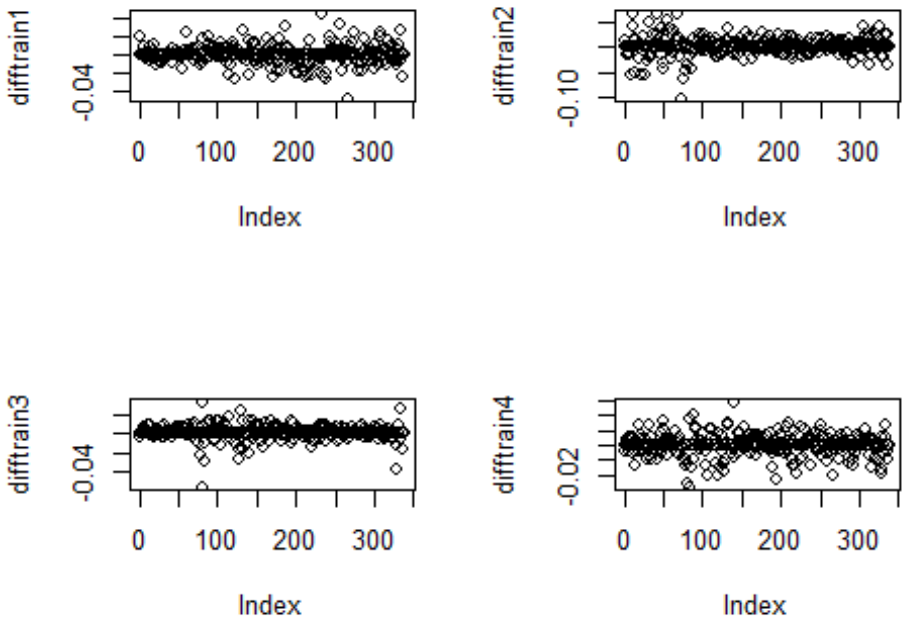
```
## [276]  0.000000e+00  2.241368e-03  1.083859e-03 -1.747366e-04 -1.119038e-
03
## [281]  6.521724e-03  0.000000e+00  0.000000e+00  7.515057e-03  3.104679e-
04
## [286]  5.297570e-03 -4.366744e-03  3.445840e-05  0.000000e+00
0.000000e+00
## [291]  0.000000e+00 -1.310255e-03 -3.456264e-03 -3.190348e-03 -1.529424e-
03
## [296]  0.000000e+00  0.000000e+00  1.946134e-03  3.292840e-03 -4.845800e-
04
## [301]  5.833940e-03  3.441393e-04  0.000000e+00  0.000000e+00 -5.278331e-
03
## [306]  5.347145e-03  1.169108e-03  8.282046e-03  2.161869e-03
0.000000e+00
## [311]  0.000000e+00 -5.131407e-03 -1.402501e-03 -3.360310e-03  2.880957e-
03
## [316] -3.082772e-04  0.000000e+00  0.000000e+00  3.795460e-03 -5.462056e-
04
## [321]  0.000000e+00 -7.127956e-03 -4.688209e-03  0.000000e+00
0.000000e+00
## [326] -1.382218e-04 -1.902489e-03 -3.778694e-02 -1.291975e-02  8.342752e-
03
## [331]  0.000000e+00  0.000000e+00 -6.995951e-03  2.468285e-02 -5.679801e-
04
## [336] -1.574723e-02  0.000000e+00  0.000000e+00

difftrain4

##   [1] -0.0045045121  0.0000000000  0.0000000000  0.0042235738
0.0047652508
##   [6] -0.0044843124  0.0000000000  0.0053228869  0.0000000000
0.0000000000
##  [11] -0.0047612469  0.0055991188  0.0013948949 -0.0039106195 -
0.0140926797
##  [16]  0.0000000000  0.0000000000  0.0132526964  0.0008399832 -
0.0039259725
##  [21]  0.0036460565  0.0061401253  0.0000000000  0.0000000000 -
0.0036236973
##  [26]  0.0041800256  0.0000000000 -0.0094999317 -0.0033745814
0.0000000000
##  [31]  0.0000000000  0.0134305320 -0.0072524725 -0.0092814617
0.0056354052
##  [36] -0.0050704334  0.0000000000  0.0000000000  0.0000000000
0.0000000000
##  [41]  0.0000000000 -0.0093630986 -0.0042851087  0.0000000000
0.0000000000
##  [46]  0.0000000000 -0.0057421921  0.0080298682  0.0155876757
0.0106294707
##  [51]  0.0000000000  0.0000000000  0.0033333364  0.0016625108
0.0000000000
```

```
##  [56] -0.0072242604  0.0033407603  0.0000000000  0.0000000000
0.0000000000
##  [61]  0.0024982664 -0.0022203728  0.0016657416  0.0093871479
0.0000000000
##  [66]  0.0000000000  0.0041135393  0.0013674281 -0.0049315168 -
0.0066134162
##  [71]  0.0057891271  0.0000000000  0.0000000000 -0.0011001101 -
0.0085671382
##  [76] -0.0008329863 -0.0092089391 -0.0129798662  0.0000000000
0.0000000000
##  [81] -0.0125751871 -0.0250449899  0.0178289101 -0.0276082146 -
0.0153041247
##  [86]  0.0000000000  0.0000000000  0.0194646061 -0.0092358768
0.0104214134
##  [91]  0.0026623296  0.0120394935  0.0000000000  0.0000000000
0.0000000000
##  [96]  0.0026235258 -0.0023316828  0.0032046641  0.0000000000
0.0000000000
## [101]  0.0000000000  0.0081112843 -0.0040474181 -0.0040638663 -
0.0137650438
## [106] -0.0199559862  0.0000000000  0.0000000000  0.0104744336 -
0.0005955926
## [111]  0.0059400234  0.0111897516  0.0002927829  0.0000000000
0.0000000000
## [116]  0.0040899853 -0.0117303398  0.0005897965  0.0079283927 -
0.0203877648
## [121]  0.0000000000  0.0000000000 -0.0141291994  0.0069392344 -
0.0045201222
## [126] -0.0176726111 -0.0120613802  0.0000000000  0.0000000000 -
0.0093779993
## [131]  0.0140342238  0.0018564362  0.0141193260  0.0000000000
0.0000000000
## [136]  0.0000000000  0.0000000000 -0.0051948169 -0.0033757896
0.0287855051
## [141] -0.0077961414  0.0000000000  0.0000000000  0.0033057881
0.0080681746
## [146] -0.0059701670  0.0074571561  0.0011880013  0.0000000000
0.0000000000
## [151] -0.0014852223  0.0127013966  0.0032234460 -0.0017569551
0.0020494810
## [156]  0.0000000000  0.0000000000  0.0061233607 -0.0014545457 -
0.0052539525
## [161]  0.0072897247  0.0037697594  0.0000000000  0.0000000000
0.0054842106
## [166]  0.0000000000  0.0065987903 -0.0071746613  0.0071746613
0.0000000000
## [171]  0.0000000000  0.0071235520 -0.0031281129  0.0036958109
0.0011344301
## [176]  0.0005667328  0.0000000000  0.0000000000  0.0000000000 -
0.0002833263
```

```
## [181] -0.0002834065  0.0067796870 -0.0022547924  0.0000000000
0.0000000000
## [186]  0.0000000000  0.0084293840 -0.0152244271 -0.0054123476 -
0.0022876761
## [191]  0.0000000000  0.0000000000  0.0000000000 -0.0234623362
0.0029265458
## [196] -0.0043930368  0.0105110457  0.0000000000  0.0000000000
0.0063694483
## [201] -0.0028901754  0.0000000000  0.0008679300 -0.0052189159
0.0000000000
## [206]  0.0000000000  0.0124225200 -0.0028752176  0.0025880676
0.0142574414
## [211] -0.0082445385  0.0000000000  0.0000000000 -0.0129293570 -
0.0090051444
## [216]  0.0029137550 -0.0128845121  0.0099707571  0.0000000000
0.0000000000
## [221] -0.0185545008  0.0000000000  0.0062231643 -0.0100950976
0.0118660544
## [226]  0.0000000000  0.0000000000 -0.0082914304  0.0100592564
0.0000000000
## [231]  0.0102504924  0.0017467253  0.0000000000  0.0000000000
0.0055112540
## [236]  0.0026000304 -0.0136530806  0.0055417963  0.0029044458
0.0000000000
## [241]  0.0000000000 -0.0008704483  0.0020298688  0.0075036427 -
0.0043221505
## [246] -0.0017341045  0.0000000000  0.0000000000 -0.0026068081
0.0095252560
## [251] -0.0066292216  0.0129293570  0.0059769639  0.0000000000
0.0000000000
## [256] -0.0039806706  0.0042643988 -0.0059752632 -0.0111925404
0.0046069762
## [261]  0.0000000000  0.0000000000 -0.0011497558  0.0083059335 -
0.0045740503
## [266]  0.0000000000 -0.0199688302  0.0000000000  0.0000000000 -
0.0029274026
## [271] -0.0002932121  0.0000000000 -0.0091324836 -0.0029638433
0.0000000000
## [276]  0.0000000000  0.0070985214  0.0055841438  0.0026342764 -
0.0008773213
## [281]  0.0032130888  0.0000000000  0.0000000000  0.0081324875 -
0.0005787037
## [286]  0.0002893937 -0.0075515898 -0.0120253686  0.0000000000
0.0000000000
## [291]  0.0000000000 -0.0091893340 -0.0128879350 -0.0057480116 -
0.0018220473
## [296]  0.0000000000  0.0000000000  0.0063626938  0.0000000000
0.0051212644
## [301]  0.0021011564  0.0035917429  0.0000000000  0.0000000000
0.0005973716
```

```
## [306]  0.0020879948  0.0038661758  0.0109226178  0.0020530877
0.0000000000
## [311]  0.0000000000 -0.0061719520  0.0038252217  0.0046879664
0.0043750981
## [316] -0.0143634303  0.0000000000  0.0000000000  0.0041249322 -
0.0076741817
## [321]  0.0000000000 -0.0032645822 -0.0098581829  0.0000000000
0.0000000000
## [326] -0.0105629472  0.0051445113 -0.0158202265 -0.0182582822 -
0.0059514663
## [331]  0.0000000000  0.0000000000  0.0059514663  0.0139558785 -
0.0043209944
## [336] -0.0093226281  0.0000000000  0.0000000000
```

```
#We can have a look at the plot where the linear trend have been removed
#after apllying differencing to our data

par(mfrow = c(2,2))
plot(difftrain1)
plot(difftrain2)
plot(difftrain3)
plot(difftrain4)
```




```
###performing Dickey-Fuller test to training data
adf.test(train1)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  train1
## Dickey-Fuller = -1.5557, Lag order = 6, p-value = 0.7641
## alternative hypothesis: stationary
```

```
adf.test(train2)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  train2
## Dickey-Fuller = -2.8293, Lag order = 6, p-value = 0.2268
## alternative hypothesis: stationary
```

```
adf.test(train3)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  train3
## Dickey-Fuller = -2.8681, Lag order = 6, p-value = 0.2105
## alternative hypothesis: stationary
```

```
adf.test(train4)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  train4
## Dickey-Fuller = -2.6266, Lag order = 6, p-value = 0.3123
## alternative hypothesis: stationary
```

```
#With a significant p-value of all 4 data, we cannot reject the null
hypothesis of non-stationarity in our series.
###performing Dickey-Fuller test to differenced data

adf.test(difftrain1)

## Warning in adf.test(difftrain1): p-value smaller than printed p-value

##
##  Augmented Dickey-Fuller Test
##
## data:  difftrain1
## Dickey-Fuller = -7.0016, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary

adf.test(difftrain2)

## Warning in adf.test(difftrain2): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  difftrain2
## Dickey-Fuller = -7.035, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

```
adf.test(difftrain3)
```

```
## Warning in adf.test(difftrain3): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  difftrain3
## Dickey-Fuller = -7.1574, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

```
adf.test(difftrain4)
```

```
## Warning in adf.test(difftrain4): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  difftrain4
## Dickey-Fuller = -6.3992, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

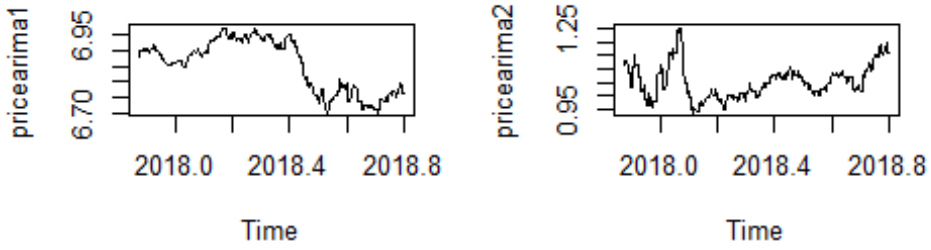
```
#However, when our data is first-differenced, we see that our p-value drops
below 0.05, -
# for all data and we can therefore now reject the null hypothesis of non-
stationarity

###Applying time series function in order to apply arima model
#choosing daily frequency and selecting the number day of that year
#for the start date.
pricearima1 <- ts(train1, start = c(2017,320), frequency = 365)
pricearima2 <- ts(train2, start = c(2017,320), frequency = 365)
pricearima3 <- ts(train3, start = c(2017,320), frequency = 365)
pricearima4 <- ts(train4, start = c(2017,320), frequency = 365)


### Applying auto.arima() function in order to choose suitable model
fittrain1<-auto.arima(pricearima1)
```

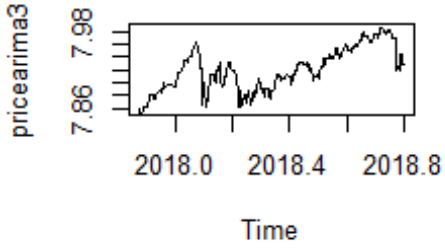
```
## Warning in value[[3L]](cond): The chosen test encountered an error, so no
## seasonal differencing is selected. Check the time series data.
```

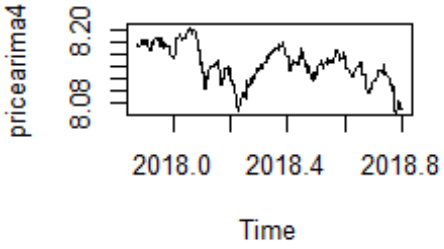
```
fittrain2<-auto.arima(pricearima2)
```

```
## Warning in value[[3L]](cond): The chosen test encountered an error, so no
## seasonal differencing is selected. Check the time series data.

fittrain3<-auto.arima(pricearima3)

## Warning in value[[3L]](cond): The chosen test encountered an error, so no
## seasonal differencing is selected. Check the time series data.

fittrain4<-auto.arima(pricearima4)

## Warning in value[[3L]](cond): The chosen test encountered an error, so no
## seasonal differencing is selected. Check the time series data.

fittrain1

## Series: pricearima1
## ARIMA(0,1,0)
##
## sigma^2 estimated as 9.193e-05:  log likelihood=1091.43
## AIC=-2180.85   AICc=-2180.84   BIC=-2177.03

fittrain2

## Series: pricearima2
## ARIMA(0,1,0)
##
## sigma^2 estimated as 0.0003019:  log likelihood=890.2
## AIC=-1778.39   AICc=-1778.38   BIC=-1774.57

fittrain3

## Series: pricearima3
## ARIMA(0,1,3)
##
## Coefficients:
##           ma1      ma2     ma3
##       -0.0297  -0.0651  0.2316
## s.e.   0.0530   0.0567  0.0538
##
## sigma^2 estimated as 5.157e-05:  log likelihood=1190.9
## AIC=-2373.8   AICc=-2373.68   BIC=-2358.51

fittrain4

## Series: pricearima4
## ARIMA(0,1,0)
##
## sigma^2 estimated as 4.934e-05:  log likelihood=1197.01
## AIC=-2392.02   AICc=-2392   BIC=-2388.19

#Moreover, running the auto.arima function in R for both models returns with
ARIMA(0, 1, 0)
#for Soyabean, natural gas and stoxx 50 except S&P500 whose arima model is
```

```
ARIMA(0,1,3)
#Actually in R auto.arima cannot handle higher frequency like 365 so it
ignores seasonality
#This is the first reason to drop this model


###Plotting time series
par(mfrow = c(2,2))
plot(pricearima1, type='l')
plot(pricearima2, type='l')
plot(pricearima3, type='l')
plot(pricearima4, type='l')
```








```
#Having a look at the time series we can say none of the pattern goes well


exp(train1)

##   [1]  972.00  990.50  990.50  990.50  990.00  989.00  997.25  997.25
##   [9]  993.25  993.25  993.25  996.00  993.00  992.50  985.75  994.25
##  [17]  994.25  994.25  998.50 1008.50 1002.75  992.00  989.75  989.75
##  [25]  989.75  982.50  975.75  975.75  967.75  967.25  967.25  967.25
##  [33]  961.50  956.00  954.00  948.75  949.50  949.50  949.50  949.50
##  [41]  949.50  955.50  945.75  951.75  951.75  951.75  951.75  955.00
##  [49]  959.75  959.25  961.50  961.50  961.50  958.25  955.75  955.75
##  [57]  940.50  944.00  944.00  944.00  944.00  968.00  968.75  973.00
##  [65]  977.25  977.25  977.25  984.25  986.25  992.25  992.25  985.50
```

```
##  [73]  985.50  985.50  991.50 1000.25  995.75  985.00  978.75  978.75
##  [81]  978.75  969.75  986.25  983.00  987.75  983.00  983.00  983.00
##  [89] 1001.75 1011.75 1017.25 1024.25 1021.50 1021.50 1021.50 1021.50
##  [97] 1026.50 1034.25 1032.00 1036.25 1036.25 1036.25 1034.25 1038.00
## [105] 1045.00 1057.50 1060.75 1060.75 1060.75 1066.75 1064.50 1055.00
## [113] 1053.75 1029.25 1029.25 1029.25 1033.25 1040.50 1028.50 1040.75
## [121] 1049.50 1049.50 1049.50 1022.50 1028.25 1029.75 1029.75 1028.25
## [129] 1028.25 1028.25 1025.50 1019.50 1018.00 1044.75 1044.75 1044.75
## [137] 1044.75 1044.75 1038.00 1015.25 1031.25 1033.75 1033.75 1033.75
## [145] 1047.00 1050.00 1047.75 1060.75 1054.25 1054.25 1054.25 1042.00
## [153] 1046.00 1041.75 1037.25 1028.75 1028.75 1028.75 1020.75 1022.25
## [161] 1027.50 1028.00 1045.00 1045.00 1045.00 1037.75 1037.75 1032.75
## [169] 1043.25 1027.25 1027.25 1027.25 1002.25 1011.25 1007.25 1013.25
## [177]  994.75  994.75  994.75 1013.00 1018.75  999.75  995.00  998.50
## [185]  998.50  998.50  998.50 1030.50 1039.25 1035.75 1041.50 1041.50
## [193] 1041.50 1041.50 1030.50 1023.00 1018.50 1021.25 1021.25 1021.25
## [201] 1001.75 1001.25  994.25  974.25  969.25  969.25  969.25  953.75
## [209]  954.00  936.00  927.25  905.50  905.50  905.50  908.50  889.00
## [217]  889.50  880.50  894.50  894.50  894.50  874.50  867.25  867.50
## [225]  861.25  858.50  858.50  858.50  848.50  844.00  844.00  835.50
## [233]  874.00  874.00  874.00  851.75  852.25  829.75  830.00  814.00
## [241]  814.00  814.00  829.50  839.50  842.25  846.00  849.75  849.75
## [249]  849.75  847.75  858.00  860.75  861.25  870.50  870.50  870.50
## [257]  875.00  903.75  886.75  882.50  886.25  886.25  886.25  877.25
## [265]  889.25  893.75  887.75  846.00  846.00  846.00  853.50  864.50
## [273]  864.50  885.50  881.50  881.50  881.50  881.75  874.50  858.25
## [281]  842.00  842.00  842.00  842.00  834.75  820.50  823.25  819.50
## [289]  833.00  833.00  833.00  833.00  832.00  825.50  826.75  832.00
## [297]  832.00  832.00  833.50  820.25  829.00  822.75  821.50  821.50
## [305]  821.50  823.50  814.00  830.00  850.25  847.25  847.25  847.25
## [313]  841.00  845.75  850.00  855.00  845.50  845.50  845.50  857.75
## [321]  866.00  866.00  859.25  869.00  869.00  869.00  869.75  863.00
## [329]  852.25  858.25  867.50  867.50  867.50  891.50  884.75  885.75
## [337]  863.50  863.50  863.50
```

```
exp(train2)
```

```
##   [1] 3.053 3.097 3.097 3.097 3.047 3.017 2.968 2.968 2.813 2.813 2.813
##  [12] 2.928 3.128 3.179 3.025 3.061 3.061 3.061 2.985 2.914 2.922 2.763
##  [23] 2.772 2.772 2.772 2.828 2.678 2.678 2.684 2.612 2.612 2.612 2.745
##  [34] 2.692 2.637 2.598 2.667 2.667 2.667 2.667 2.667 2.732 2.914 2.953
##  [45] 2.953 2.953 2.953 3.056 3.008 2.880 2.795 2.795 2.795 2.835 2.923
##  [56] 2.923 3.084 3.200 3.200 3.200 3.200 3.129 3.232 3.189 3.185 3.185
##  [67] 3.185 3.224 3.444 3.509 3.447 3.505 3.505 3.505 3.167 3.195 2.995
##  [78] 2.856 2.846 2.846 2.846 2.747 2.759 2.702 2.697 2.584 2.584 2.584
##  [89] 2.552 2.594 2.587 2.580 2.558 2.558 2.558 2.558 2.616 2.659 2.634
## [100] 2.625 2.625 2.625 2.686 2.683 2.667 2.698 2.695 2.695 2.695 2.704
## [111] 2.749 2.777 2.756 2.732 2.732 2.732 2.778 2.786 2.731 2.681 2.688
## [122] 2.688 2.688 2.651 2.675 2.638 2.617 2.591 2.591 2.591 2.618 2.714
## [133] 2.698 2.733 2.733 2.733 2.733 2.733 2.697 2.718 2.675 2.701 2.701
```

```
## [144] 2.701 2.693 2.656 2.675 2.686 2.735 2.735 2.735 2.752 2.738 2.739
## [155] 2.660 2.739 2.739 2.739 2.740 2.781 2.786 2.839 2.771 2.771 2.771
## [166] 2.763 2.763 2.754 2.726 2.711 2.711 2.711 2.741 2.732 2.737 2.814
## [177] 2.806 2.806 2.806 2.842 2.836 2.815 2.859 2.847 2.847 2.847 2.847
## [188] 2.908 2.914 2.940 2.939 2.939 2.939 2.939 2.903 2.885 2.952 2.962
## [199] 2.962 2.962 2.930 2.890 2.896 2.930 2.890 2.890 2.890 2.949 2.939
## [210] 2.963 2.965 3.022 3.022 3.022 2.951 2.900 2.964 2.975 2.945 2.945
## [221] 2.945 2.923 2.939 2.981 2.940 2.924 2.924 2.924 2.862 2.870 2.870
## [232] 2.837 2.858 2.858 2.858 2.828 2.788 2.829 2.797 2.752 2.752 2.752
## [243] 2.759 2.740 2.721 2.769 2.757 2.757 2.757 2.721 2.732 2.775 2.780
## [254] 2.782 2.782 2.782 2.797 2.782 2.758 2.816 2.853 2.853 2.853 2.860
## [265] 2.897 2.949 2.955 2.944 2.944 2.944 2.930 2.959 2.959 2.908 2.946
## [276] 2.946 2.946 2.941 2.980 2.956 2.964 2.917 2.917 2.917 2.876 2.852
## [287] 2.863 2.874 2.916 2.916 2.916 2.916 2.823 2.795 2.772 2.776 2.776
## [298] 2.776 2.804 2.828 2.829 2.817 2.767 2.767 2.767 2.814 2.933 2.908
## [309] 2.976 2.977 2.977 2.977 3.038 3.082 2.980 3.056 3.008 3.008 3.008
## [320] 3.094 3.166 3.166 3.165 3.143 3.143 3.143 3.267 3.266 3.284 3.222
## [331] 3.161 3.161 3.161 3.242 3.239 3.320 3.198 3.198 3.198
```

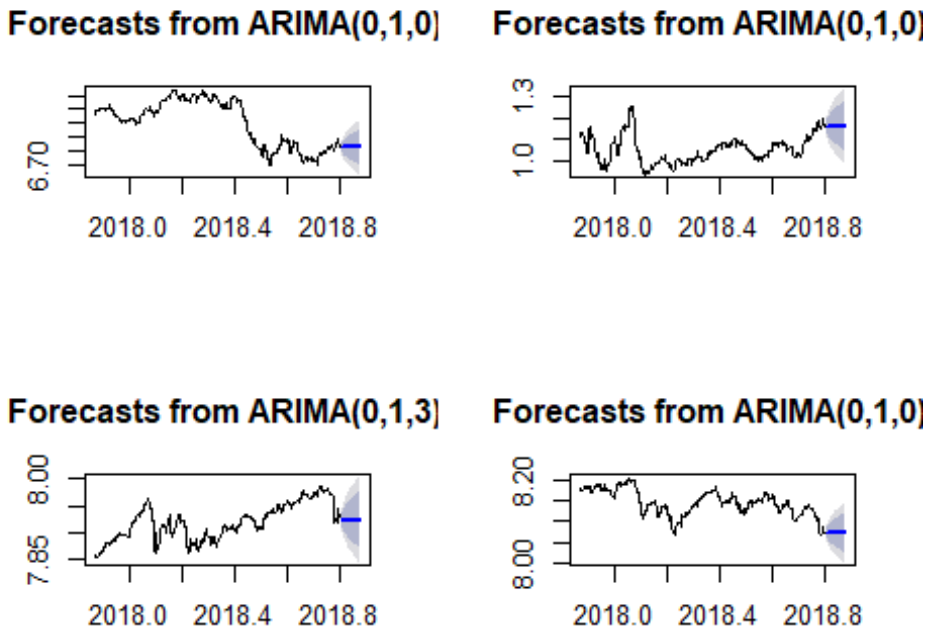
```
exp(train3)
```

```
##   [1] 2585.00 2576.30 2576.30 2576.30 2582.10 2596.30 2594.60 2594.60
##   [9] 2601.00 2601.00 2601.00 2601.80 2626.00 2625.10 2647.90 2643.90
##  [17] 2643.90 2643.90 2638.20 2628.30 2629.20 2639.50 2651.00 2651.00
##  [25] 2651.00 2661.60 2665.10 2665.10 2653.20 2669.02 2669.02 2669.02
##  [33] 2694.40 2684.10 2681.60 2687.80 2686.10 2686.10 2686.10 2686.10
##  [41] 2686.10 2685.40 2685.70 2676.00 2676.00 2676.00 2676.00 2692.90
##  [49] 2710.90 2723.80 2742.60 2742.60 2742.60 2746.80 2752.30 2752.30
##  [57] 2769.50 2788.70 2788.70 2788.70 2788.70 2782.60 2803.70 2796.20
##  [65] 2811.00 2811.00 2811.00 2835.20 2839.60 2841.00 2841.20 2874.50
##  [73] 2874.50 2874.50 2853.60 2824.50 2825.80 2822.40 2756.70 2756.70
##  [81] 2756.70 2607.70 2694.30 2668.20 2593.40 2619.10 2619.10 2619.10
##  [89] 2655.20 2661.70 2696.90 2733.90 2735.00 2735.00 2735.00 2735.00
##  [97] 2714.10 2698.70 2711.50 2748.80 2748.80 2748.80 2784.40 2747.50
## [105] 2714.40 2678.20 2690.20 2690.20 2690.20 2718.40 2723.90 2723.30
## [113] 2739.30 2783.90 2783.90 2783.90 2784.10 2768.50 2749.70 2751.60
## [121] 2751.83 2751.83 2751.83 2722.80 2723.60 2718.30 2643.30 2597.80
## [129] 2597.80 2597.80 2659.50 2615.70 2607.60 2643.00 2643.00 2643.00
## [137] 2643.00 2643.00 2613.20 2647.00 2661.80 2605.70 2605.70 2605.70
## [145] 2619.10 2655.10 2641.10 2664.00 2657.30 2657.30 2657.30 2681.70
## [153] 2706.60 2709.70 2693.10 2671.60 2671.60 2671.60 2671.30 2635.30
## [161] 2644.50 2674.60 2671.40 2671.40 2671.40 2647.00 2647.00 2627.40
## [169] 2631.80 2663.10 2663.10 2663.10 2670.00 2670.30 2696.00 2718.70
## [177] 2729.40 2729.40 2729.40 2730.90 2709.00 2722.90 2718.70 2713.00
## [185] 2713.00 2713.00 2713.00 2726.10 2730.70 2727.60 2718.30 2718.30
## [193] 2718.30 2718.30 2692.20 2724.60 2705.50 2733.80 2733.80 2733.80
## [201] 2745.50 2751.60 2772.20 2772.20 2778.80 2778.80 2778.80 2783.00
## [209] 2783.90 2774.30 2783.20 2776.85 2776.85 2776.85 2779.70 2766.20
## [217] 2772.00 2752.50 2759.50 2759.50 2759.50 2722.30 2728.50 2704.90
## [225] 2719.50 2721.60 2721.60 2721.60 2727.30 2713.30 2713.30 2738.60
```

```
## [233] 2762.90 2762.90 2762.90 2787.40 2796.70 2774.10 2798.50 2803.20
## [241] 2803.20 2803.20 2796.50 2811.30 2816.00 2805.30 2800.80 2800.80
## [249] 2800.80 2812.00 2821.00 2841.30 2842.50 2817.50 2817.50 2817.50
## [257] 2803.20 2817.10 2810.80 2828.40 2839.50 2839.50 2839.50 2850.10
## [265] 2859.70 2855.30 2853.80 2836.80 2836.80 2836.80 2825.50 2841.00
## [273] 2841.00 2844.70 2852.20 2852.20 2852.20 2858.60 2861.70 2861.20
## [281] 2858.00 2876.70 2876.70 2876.70 2898.40 2899.30 2914.70 2902.00
## [289] 2902.10 2902.10 2902.10 2902.10 2898.30 2888.30 2879.10 2874.70
## [297] 2874.70 2874.70 2880.30 2889.80 2888.40 2905.30 2906.30 2906.30
## [305] 2906.30 2891.00 2906.50 2909.90 2934.10 2940.45 2940.45 2940.45
## [313] 2925.40 2921.30 2911.50 2919.90 2919.00 2919.00 2919.00 2930.10
## [321] 2928.50 2928.50 2907.70 2894.10 2894.10 2894.10 2893.70 2888.20
## [329] 2781.10 2745.40 2768.40 2768.40 2768.40 2749.10 2817.80 2816.20
## [337] 2772.20 2772.20 2772.20

exp(train4)

##   [1] 3560 3544 3544 3544 3559 3576 3560 3560 3579 3579 3579 3562 3582
3587
##  [15] 3573 3523 3523 3523 3570 3573 3559 3572 3594 3594 3594 3581 3596
3596
##  [29] 3562 3550 3550 3550 3598 3572 3539 3559 3541 3541 3541 3541 3541
3541
##  [43] 3508 3493 3493 3493 3493 3473 3501 3556 3594 3594 3594 3606 3612
3612
##  [57] 3586 3598 3598 3598 3598 3607 3599 3605 3639 3639 3639 3654 3659
3641
##  [71] 3617 3638 3638 3638 3634 3603 3600 3567 3521 3521 3521 3477 3391
3452
##  [85] 3358 3307 3307 3307 3372 3341 3376 3385 3426 3426 3426 3426 3435
3427
##  [99] 3438 3438 3438 3438 3466 3452 3438 3391 3324 3324 3324 3359 3357
3377
## [113] 3415 3416 3416 3416 3430 3390 3392 3419 3350 3350 3350 3303 3326
3311
## [127] 3253 3214 3214 3214 3184 3229 3235 3281 3281 3281 3281 3281 3264
3253
## [141] 3348 3322 3322 3322 3333 3360 3340 3365 3369 3369 3369 3364 3407
3418
## [155] 3412 3419 3419 3419 3440 3435 3417 3442 3455 3455 3455 3474 3474
3497
## [169] 3472 3497 3497 3497 3522 3511 3524 3528 3530 3530 3530 3530 3529
3528
## [183] 3552 3544 3544 3544 3544 3574 3520 3501 3493 3493 3493 3493 3412
3422
## [197] 3407 3443 3443 3443 3465 3455 3455 3458 3440 3440 3440 3483 3473
3482
## [211] 3532 3503 3503 3503 3458 3427 3437 3393 3427 3427 3427 3364 3364
3385
## [225] 3351 3391 3391 3391 3363 3397 3397 3432 3438 3438 3438 3457 3466
```

```
3419
## [239] 3438 3448 3448 3448 3445 3452 3478 3463 3457 3457 3457 3448 3481
3458
## [253] 3503 3524 3524 3524 3510 3525 3504 3465 3481 3481 3481 3477 3506
3490
## [267] 3490 3421 3421 3421 3411 3410 3410 3379 3369 3369 3369 3393 3412
3421
## [281] 3418 3429 3429 3429 3457 3455 3456 3430 3389 3389 3389 3389 3358
3315
## [295] 3296 3290 3290 3290 3311 3311 3328 3335 3347 3347 3347 3349 3356
3369
## [309] 3406 3413 3413 3413 3392 3405 3421 3436 3387 3387 3387 3401 3375
3375
## [323] 3364 3331 3331 3331 3296 3313 3261 3202 3183 3183 3183 3202 3247
3233
## [337] 3203 3203 3203
```

```r
#As previously mentioned, our data is in logarithmic format. Since we are analysing stock price, -
#this format is necessary to account for compounding returns. However, once we have obtained the -
#forecasts, then we can obtain the real Settle forecast by -
#converting the logarithmic figure to an exponent

#Forecasting the Values From ARIMA with the 26 period
forecastedvalues_ln1=forecast(fittrain1,h=26)
forecastedvalues_ln2=forecast(fittrain2,h=26)
forecastedvalues_ln3=forecast(fittrain3,h=26)
forecastedvalues_ln4=forecast(fittrain4,h=26)

#Let's have a look at the trend of forecasted value using the plot graph
par(mfrow = c(2,2))
plot(forecastedvalues_ln1)
plot(forecastedvalues_ln2)
plot(forecastedvalues_ln3)
plot(forecastedvalues_ln4)
```




```
#this plot will be described below

forecastedvaluesextracted1=as.numeric(forecastedvalues_ln1$mean)
forecastedvaluesextracted2=as.numeric(forecastedvalues_ln2$mean)
forecastedvaluesextracted3=as.numeric(forecastedvalues_ln3$mean)
forecastedvaluesextracted4=as.numeric(forecastedvalues_ln4$mean)
finalforecastvalues1=exp(forecastedvaluesextracted1)
finalforecastvalues2=exp(forecastedvaluesextracted2)
finalforecastvalues3=exp(forecastedvaluesextracted3)
finalforecastvalues4=exp(forecastedvaluesextracted4)
finalforecastvalues1
```

```
##  [1] 863.5 863.5 863.5 863.5 863.5 863.5 863.5 863.5 863.5 863.5 863.5
## [12] 863.5 863.5 863.5 863.5 863.5 863.5 863.5 863.5 863.5 863.5 863.5
## [23] 863.5 863.5 863.5 863.5
```

```
finalforecastvalues2
```

```
##  [1] 3.198 3.198 3.198 3.198 3.198 3.198 3.198 3.198 3.198 3.198 3.198
## [12] 3.198 3.198 3.198 3.198 3.198 3.198 3.198 3.198 3.198 3.198 3.198
## [23] 3.198 3.198 3.198 3.198
```

```
finalforecastvalues3
```

```
##  [1] 2765.265 2761.710 2761.273 2761.273 2761.273 2761.273 2761.273
##  [8] 2761.273 2761.273 2761.273 2761.273 2761.273 2761.273 2761.273
## [15] 2761.273 2761.273 2761.273 2761.273 2761.273 2761.273 2761.273
## [22] 2761.273 2761.273 2761.273 2761.273 2761.273
```

```
finalforecastvalues4
```

```
##  [1] 3203 3203 3203 3203 3203 3203 3203 3203 3203 3203 3203 3203 3203 3203
## [15] 3203 3203 3203 3203 3203 3203 3203 3203 3203 3203 3203 3203
```

```
#Creating a dataframe with forecasted and actual value for referencing
df1<-data.frame(TOTdf$SettleSOY[340:365],finalforecastvalues1)
df2<-data.frame(TOTdf$SettleNG[340:365],finalforecastvalues1)
df3<-data.frame(TOTdf$SettleSP[340:365],finalforecastvalues1)
df4<-data.frame(TOTdf$SettleST[340:365],finalforecastvalues1)
col_headings<-c("Actual Price","Forecasted Price")
names(df1)<-col_headings
names(df2)<-col_headings
names(df3)<-col_headings
names(df4)<-col_headings
attach(df1)
attach(df2)
```

```
## The following objects are masked from df1:
##
##     Actual Price, Forecasted Price
```

```
attach(df3)
```

```
## The following objects are masked from df2:
##
##     Actual Price, Forecasted Price
```

```
## The following objects are masked from df1:
##
##     Actual Price, Forecasted Price
```

```
attach(df4)
```

```
## The following objects are masked from df3:
##
##     Actual Price, Forecasted Price
```

```
## The following objects are masked from df2:
##
##     Actual Price, Forecasted Price
```

```
## The following objects are masked from df1:
##
##     Actual Price, Forecasted Price
```

```
#Top rows of the newly created dataframe to hve a glance
head(df1)
```

```
##   Actual Price Forecasted Price
## 1       863.50            863.5
## 2       858.50            863.5
```

```
## 3       858.50            863.5
## 4       850.25            863.5
## 5       841.75            863.5
## 6       845.00            863.5
```

```
head(df2)
```

```
##   Actual Price Forecasted Price
## 1        3.198            863.5
## 2        3.138            863.5
## 3        3.138            863.5
## 4        3.166            863.5
## 5        3.202            863.5
## 6        3.185            863.5
```

```
head(df3)
```

```
##   Actual Price Forecasted Price
## 1       2772.2            863.5
## 2       2756.5            863.5
## 3       2756.5            863.5
## 4       2664.3            863.5
## 5       2688.2            863.5
## 6       2669.6            863.5
```

```
head(df4)
```

```
##   Actual Price Forecasted Price
## 1         3203            863.5
## 2         3182            863.5
## 3         3182            863.5
## 4         3121            863.5
## 5         3156            863.5
## 6         3109            863.5
```

```
percentage_error1=((df1$`Actual Price`- df1$`Forecasted Price`)/(df1$`Actual
Price`))
percentage_error2=((df2$`Actual Price`- df2$`Forecasted Price`)/(df2$`Actual
Price`))
percentage_error3=((df3$`Actual Price`- df3$`Forecasted Price`)/(df3$`Actual
Price`))
percentage_error4=((df4$`Actual Price`- df4$`Forecasted Price`)/(df4$`Actual
Price`))
percentage_error1
```

```
##  [1]  2.633164e-16 -5.824112e-03 -5.824112e-03 -1.558365e-02 -2.583903e-02
##  [6] -2.189349e-02 -2.189349e-02 -2.189349e-02 -2.920143e-02 -3.599280e-02
## [11] -2.920143e-02  6.329114e-03  1.342474e-02  1.342474e-02  1.342474e-02
## [16]  1.088202e-02  9.747706e-03  4.897724e-03  4.610951e-03  1.342474e-02
## [21]  1.342474e-02  1.342474e-02  9.463722e-03  9.463722e-03  8.041356e-03
## [26]  2.841069e-02
```

```
percentage_error2
```

```
##  [1] -269.0125 -274.1753 -274.1753 -271.7416 -268.6752 -270.1146 -270.1146
##  [8] -270.1146 -269.0125 -269.9445 -263.7961 -265.7593 -261.9415 -261.9415
## [15] -261.9415 -241.0802 -241.8973 -241.8973 -242.7200 -231.1861 -231.1861
## [22] -231.1861 -226.9567 -226.9567 -177.5197 -212.8435
```

```
percentage_error3
```

```
##  [1] 0.6885145 0.6867404 0.6867404 0.6758999 0.6787813 0.6765433 0.6765433
##  [8] 0.6765433 0.6733621 0.6784344 0.6814946 0.6846353 0.6830378 0.6830378
## [15] 0.6830378 0.6848080 0.6870243 0.6934029 0.6925624 0.6892655 0.6892655
## [22] 0.6892655 0.6834445 0.6834445 0.6800193 0.6842201
```

```
percentage_error4
```

```
##  [1] 0.7304090 0.7286298 0.7286298 0.7233259 0.7263942 0.7222580 0.7222580
##  [8] 0.7222580 0.7256117 0.7252625 0.7296493 0.7292255 0.7309969 0.7309969
## [15] 0.7309969 0.7309131 0.7304090 0.7330757 0.7321650 0.7318323 0.7318323
## [22] 0.7318323 0.7288854 0.7288854 0.7299030 0.7286298
```

```
mean(percentage_error1)
```

```
## [1] -0.00156737
```

```
mean(percentage_error2)
```

```
## [1] -251.0727
```

```
mean(percentage_error3)
```

```
## [1] 0.6834642
```

```
mean(percentage_error4)
```

```
## [1] 0.7286641
```

```
#From the above, we see that there is an average of an 0.1% deviation between
the actual price -
#and the forecasted price provided by ARIMA for soyabean where as for other
dataset the error -
#rate is too high. Another reason to drop this model

###Ljung-Box Test
#Note that the method for choosing a specific number of lags for Ljung-Box
can be quite arbitrary.
#In this regard, we will run the Ljung-Box test with lags 5, 10, and 15. To
run this test in R,
#we use the following functions:

Box.test(fittrain1$resid, lag=5, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  fittrain1$resid
## X-squared = 7.699, df = 5, p-value = 0.1736
```

```
Box.test(fittrain2$resid, lag=5, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  fittrain2$resid
## X-squared = 3.6162, df = 5, p-value = 0.6059
```

```
Box.test(fittrain3$resid, lag=5, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  fittrain3$resid
## X-squared = 2.6326, df = 5, p-value = 0.7564
```

```
Box.test(fittrain4$resid, lag=5, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  fittrain4$resid
## X-squared = 3.6141, df = 5, p-value = 0.6062
```

```
Box.test(fittrain1$resid, lag=10, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  fittrain1$resid
## X-squared = 9.3036, df = 10, p-value = 0.5036
```

```
Box.test(fittrain2$resid, lag=10, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  fittrain2$resid
## X-squared = 10.486, df = 10, p-value = 0.3989
```

```
Box.test(fittrain3$resid, lag=10, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
```

```
## data:  fittrain3$resid
## X-squared = 5.3856, df = 10, p-value = 0.864

Box.test(fittrain4$resid, lag=10, type="Ljung-Box")

##
##  Box-Ljung test
##
## data:  fittrain4$resid
## X-squared = 7.8713, df = 10, p-value = 0.6414

Box.test(fittrain1$resid, lag=15, type="Ljung-Box")

##
##  Box-Ljung test
##
## data:  fittrain1$resid
## X-squared = 16.812, df = 15, p-value = 0.3302

Box.test(fittrain2$resid, lag=15, type="Ljung-Box")

##
##  Box-Ljung test
##
## data:  fittrain2$resid
## X-squared = 20.503, df = 15, p-value = 0.1535

Box.test(fittrain3$resid, lag=15, type="Ljung-Box")

##
##  Box-Ljung test
##
## data:  fittrain3$resid
## X-squared = 8.999, df = 15, p-value = 0.8776

Box.test(fittrain4$resid, lag=15, type="Ljung-Box")

##
##  Box-Ljung test
##
## data:  fittrain4$resid
## X-squared = 17.381, df = 15, p-value = 0.2966

#From the above, we see that we have an insignificant p-value at all lags.
This means that there -
#is likely a high degree of randomness exhibited by our residuals and
therefore our ARIMA model -
#is free of autocorrelation.
```

After looking at the plot of forecast from airma model we can come with the below conclusion on out investment data,

Soyabean:

Soyabean forecast is having a little range where risk is low but the return also will not be more as well.

Natural gas:

Natural gas forecast from arima model doesn't looks good because the range is too broad so it is very risky to invest, you may get very good return or you will lose much of your investment

Mutual funds:

Mutual funds cannot perform well when the whole stock market index doesn't perform well, as natural gas S&P500's - forecast also has very broad range which makes risky to invest in the mutual fund.

ETF

ETF on stoxx 50 might perfrm well as its range is small as well as in neutral so the probability of getting return - less than invested is very low.

But, this will not finalize our insights as per the expert's advice we will also apply - artificial neural network mode to our data and then finalize our insights. In addtion to this here ARIMA model - doesn't fit well with our data, so we must look for alternative (ANN).

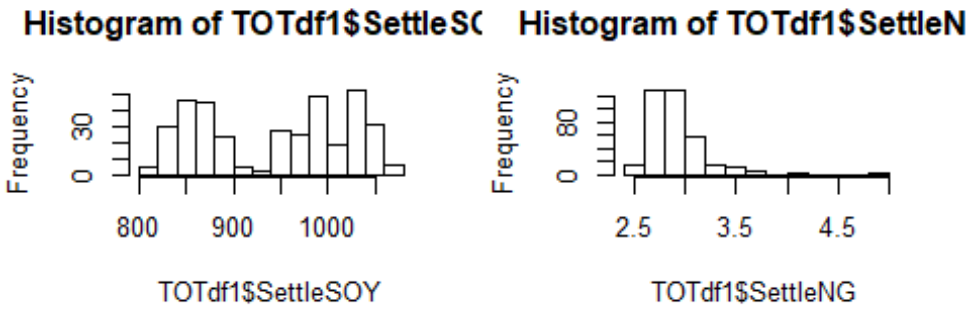
ANN model:

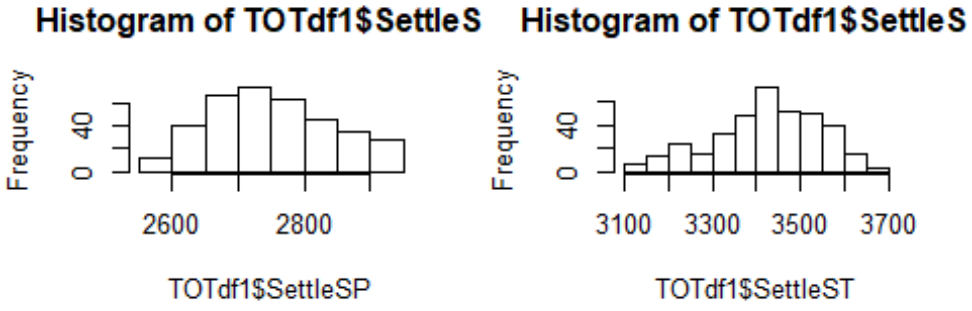
```
#ANN Model:
#Loading required library
library(neuralnet)

##
## Attaching package: 'neuralnet'

## The following object is masked from 'package:dplyr':
##
##     compute

#set.seed() function is used to make sure that we get same result every time
we run neural function
set.seed(1234)
#Plotting a histogram graph for the response variable to be forecasted
#In order to scale dataset to apply ANN the dataset should have all numeric
value,
#as our dataset has "Date" value we will replicate our dataframe without
"Date"
TOTdf1 <- TOTdf[,2:length(TOTdf)]
str(TOTdf1)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    365 obs. of  61 variables:
##  $ HighDAS                       : num  441 427 427 427 455 ...
##  $ LowDAS                        : num  409 409 409 409 424 ...
##  $ MidDAS                        : num  421 421 421 421 442 ...
##  $ LastDAS                       : num  421 420 420 420 442 ...
##  $ BidDAS                        : num  421 420 420 420 442 ...
##  $ AskDAS                        : num  422 421 421 421 442 ...
##  $ VolumeDAS                     : num  19586 20297 20297 20297 17345 ...
##  $ HighZAS                       : num  295 324 324 324 304 ...
##  $ LowZAS                        : num  269 285 285 285 294 ...
##  $ MidZAS                        : num  288 295 295 295 300 ...
##  $ LastZAS                       : num  288 294 294 294 300 ...
##  $ BidZAS                        : num  287 294 294 294 300 ...
##  $ AskZAS                        : num  289 295 295 295 300 ...
##  $ VolumeZAS                     : num  41221 88068 88068 88068 19486 ...
##  $ ValueCAEU                     : num  1.5 1.51 1.51 1.51 1.51 ...
##  $ ValueEUCY                     : num  7.81 7.82 7.82 7.82 7.82 ...
##  $ OpenSOY                       : num  976 973 973 973 990 ...
##  $ HighSOY                       : num  978 992 992 992 992 ...
##  $ LowSOY                        : num  970 972 972 972 983 ...
##  $ LastSOY                       : num  972 990 990 990 989 ...
##  $ ChangeSOY                     : num  4.25 18.5 18.5 18.5 0.5 1 8.25 8.25
4 4 ...
##  $ SettleSOY                     : num  972 990 990 990 990 ...
##  $ VolumeSOY                     : num  85723 115931 115931 115931 88069
...
##  $ Previous_Day_Open_InterestSOY: num  323185 327103 327103 327103 319186
...
##  $ OpenNG                        : num  3.09 3.06 3.06 3.06 3.07 ...
##  $ HighNG                        : num  3.11 3.13 3.13 3.13 3.08 ...
##  $ LowNG                         : num  3.04 3.06 3.06 3.06 3.02 ...
##  $ LastNG                        : num  3.07 3.13 3.13 3.13 3.04 ...
##  $ ChangeNG                      : num  0.027 0.044 0.044 0.044 0.05 0.03
0.049 0.049 0.155 0.155 ...
##  $ SettleNG                      : num  3.05 3.1 3.1 3.1 3.05 ...
##  $ VolumeNG                      : num  1036 928 928 928 938 ...
##  $ Previous_Day_Open_InterestNG : num  1219 1260 1260 1260 1349 ...
##  $ Short_VolumeCIS               : num  716540 144234 144234 144234 222817
...
##  $ Total_VolumeCIS               : num  1009369 321193 321193 321193 402545
...
##  $ Short_VolumeGOO               : num  9973 13143 13143 13143 13032 ...
##  $ Total_VolumeGOO               : num  20168 27828 27828 27828 22268 ...
##  $ HighRBC                       : num  67.1 66.9 66.9 66.9 67.2 ...
##  $ LowRBC                        : num  66.3 66.3 66.3 66.3 66.7 ...
##  $ LastRBC                       : num  67.1 66.9 66.9 66.9 67.2 ...
##  $ Previous_Day_PriceRBC         : num  66.3 67.1 67.1 67.1 66.9 ...
##  $ VolumeRBC                     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ OpenKBC                       : num  70.5 69.2 69.2 69.2 67.1 ...
##  $ HighKBC                       : num  70.5 69.3 69.3 69.3 67.4 ...
```

```
##  $ LowKBC                          : num  68.1 67.2 67.2 67.2 66.8 ...
##  $ LastKBC                         : num  69.6 67.4 67.4 67.4 67 ...
##  $ VolumeKBC                       : num  1064491 1482100 1482100 1482100
1045550 ...
##  $ TurnoverKBC                     : num  7.37e+07 1.01e+08 1.01e+08 1.01e+08
7.00e+07 ...
##  $ OpenSP                          : num  2564 2582 2582 2582 2574 ...
##  $ HighSP                          : num  2589 2586 2586 2586 2583 ...
##  $ LowSP                           : num  2562 2575 2575 2575 2568 ...
##  $ LastSP                          : num  2585 2576 2576 2576 2582 ...
##  $ ChangeSP                        : num  19.9 8.7 8.7 8.7 5.8 14.2 1.7 1.7
6.4 6.4 ...
##  $ SettleSP                        : num  2585 2576 2576 2576 2582 ...
##  $ VolumeSP                        : num  2397 2682 2682 2682 2646 ...
##  $ Previous_Day_Open_InterestSP    : num  64596 65063 65063 65063 65885 ...
##  $ OpenST                          : num  3552 3563 3563 3563 3526 ...
##  $ HighST                          : num  3575 3575 3575 3575 3564 ...
##  $ LowST                           : num  3549 3537 3537 3537 3523 ...
##  $ SettleST                        : num  3560 3544 3544 3544 3559 ...
##  $ VolumeST                        : num  992582 951137 951137 951137 916183
...
##  $ Prev._Day_Open_InterestST       : num  3819776 3715466 3715466 3715466
3656381 ...
```

```r
par(mfrow = c(2,2))
hist(TOTdf1$SettleSOY)
hist(TOTdf1$SettleNG)
hist(TOTdf1$SettleSP)
hist(TOTdf1$SettleST)
```






```
#Checking the dimension of data frame
dim(TOTdf1)

## [1] 365  61

#Checking top 3 rows of data
head(TOTdf1,2,range)

## # A tibble: 2 x 61
##   HighDAS LowDAS MidDAS LastDAS BidDAS AskDAS VolumeDAS HighZAS LowZAS
##     <dbl>  <dbl>  <dbl>   <dbl>  <dbl>  <dbl>     <dbl>   <dbl>  <dbl>
## 1    441.   409.   421.    421.   421.   422.    19586.    295    269
## 2    427.   409.   421.    420.   420.   421.    20297.    324.   285.
## # ... with 52 more variables: MidZAS <dbl>, LastZAS <dbl>, BidZAS <dbl>,
## #   AskZAS <dbl>, VolumeZAS <dbl>, ValueCAEU <dbl>, ValueEUCY <dbl>,
## #   OpenSOY <dbl>, HighSOY <dbl>, LowSOY <dbl>, LastSOY <dbl>,
## #   ChangeSOY <dbl>, SettleSOY <dbl>, VolumeSOY <dbl>,
## #   Previous_Day_Open_InterestSOY <dbl>, OpenNG <dbl>, HighNG <dbl>,
## #   LowNG <dbl>, LastNG <dbl>, ChangeNG <dbl>, SettleNG <dbl>,
## #   VolumeNG <dbl>, Previous_Day_Open_InterestNG <dbl>,
## #   Short_VolumeCIS <dbl>, Total_VolumeCIS <dbl>, Short_VolumeGOO <dbl>,
## #   Total_VolumeGOO <dbl>, HighRBC <dbl>, LowRBC <dbl>, LastRBC <dbl>,
## #   Previous_Day_PriceRBC <dbl>, VolumeRBC <dbl>, OpenKBC <dbl>,
## #   HighKBC <dbl>, LowKBC <dbl>, LastKBC <dbl>, VolumeKBC <dbl>,
## #   TurnoverKBC <dbl>, OpenSP <dbl>, HighSP <dbl>, LowSP <dbl>,
## #   LastSP <dbl>, ChangeSP <dbl>, SettleSP <dbl>, VolumeSP <dbl>,
## #   Previous_Day_Open_InterestSP <dbl>, OpenST <dbl>, HighST <dbl>,
```

```
## #   LowST <dbl>, SettleST <dbl>, VolumeST <dbl>,
## #   Prev._Day_Open_InterestST <dbl>

#Fetching the minimum and maximum value for each of the variable
apply(TOTdf1,2,range)

##      HighDAS LowDAS   MidDAS LastDAS  BidDAS  AskDAS VolumeDAS HighZAS
## [1,]  141.83  131.0  135.785  135.67  135.68  135.89  1223.988  112.07
## [2,] 1568.70 1281.4 1427.750 1424.60 1424.80 1430.70 98420.516  799.26
##      LowZAS  MidZAS LastZAS BidZAS AskZAS  VolumeZAS ValueCAEU ValueEUCY
## [1,]  96.60 107.900  107.77 107.81 107.99   3381.115    1.4795    7.4174
## [2,] 638.17 754.395  753.57 753.80 754.99 173274.421    1.6105    8.0958
##      OpenSOY HighSOY  LowSOY LastSOY ChangeSOY SettleSOY VolumeSOY
## [1,]  813.75   823.5  810.50     812      0.25    814.00        64
## [2,] 1066.00  1071.0 1060.75    1067     41.75   1066.75    327585
##      Previous_Day_Open_InterestSOY OpenNG HighNG LowNG LastNG ChangeNG
## [1,]                            62  2.535  2.590 2.525  2.570    0.001
## [2,]                        445389  4.675  4.905 4.055  4.685    0.799
##      SettleNG VolumeNG Previous_Day_Open_InterestNG Short_VolumeCIS
## [1,]    2.552      158                          496           29760
## [2,]    4.837    15852                         4095          716540
##      Total_VolumeCIS Short_VolumeGOO Total_VolumeGOO HighRBC LowRBC
## [1,]           94746            2920            4377   61.76  61.02
## [2,]         1009369           50780           69933   70.50  70.00
##      LastRBC Previous_Day_PriceRBC VolumeRBC OpenKBC HighKBC LowKBC
## [1,]   61.49                61.49         0   59.74   60.36  59.22
## [2,]   70.50                70.50      1300   78.32   78.80  77.42
##      LastKBC VolumeKBC TurnoverKBC OpenSP HighSP LowSP LastSP ChangeSP
## [1,]   59.56    243041    15258877 2564.3 2583.1  2531 2576.3      0.1
## [2,]   77.76   1950851   129293609 2931.0 2941.8  2931 2933.0    149.0
##      SettleSP VolumeSP Previous_Day_Open_InterestSP OpenST HighST LowST
## [1,]  2576.30        0                        28883   3096   3148  3080
## [2,]  2940.45    31100                        98999   3675   3681  3653
##      SettleST VolumeST Prev._Day_Open_InterestST
## [1,]     3109   341766                         0
## [2,]     3659  3346136                   4354304

### we can see scale of each feature or variable is not same
###We need to normalize each variable in the interval [0,1] in order to apply
neural network model
###Normalization is the most important to apply in order to make sure that
each variable is scaled properly and
###none of the features or variables over dominates.
#fetches minimum value of each variable in the dataframe
minval <- apply(TOTdf1,2,min)

#fetches maximum value of each variable in the dataframe
maxval <- apply(TOTdf1,2,max)

###Creating index to break down dataframe into training and testing data.
```

```r
index <- sample(1:nrow(TOTdf1),160)

###Subsetting data that falls within index to create training data
training <- TOTdf1[index,]

#Checking the dimension of training data
dim(training)
```

```
## [1] 160  61
```

```r
###Subsetting data that falls beneath the index to create testing data
testing <- TOTdf1[-index,]

#Checking the dimension of testing data
dim(testing)
```

```
## [1] 205  61
```

```r
###Scaling provides us mean = 0 and standard deviation = 1 for each variable.
newdf <- as.data.frame(scale(TOTdf1,center=minval,scale=maxval - minval))
### we now creating index in order to divide the scaled dataframe into
testing and training data
ind <- sample(1:nrow(newdf),160)
###Generating training data from scaled dataframe
traindfm <- newdf[ind,]
#Checking the dimension of the scaled training data
dim(traindfm)
```

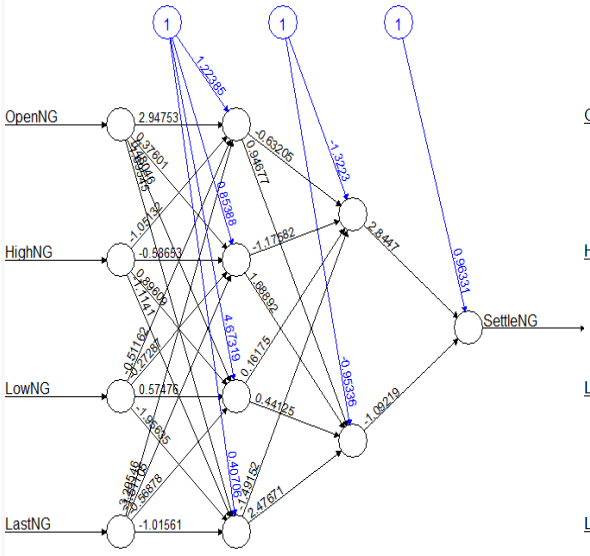
```
## [1] 160  61
```

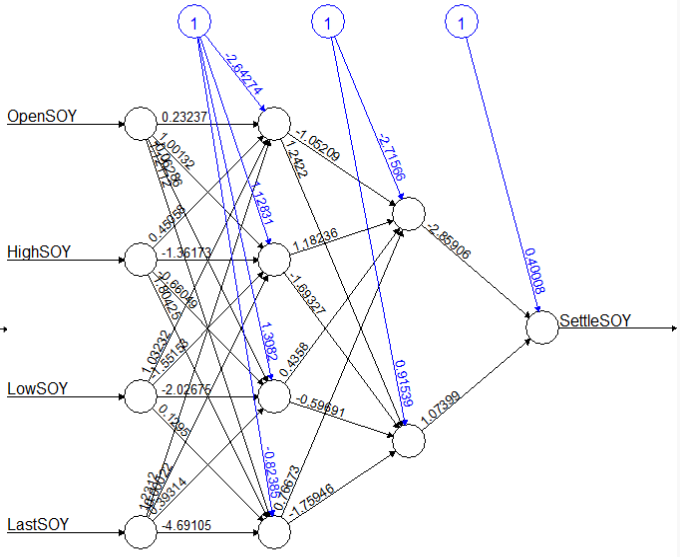
```r
###Generating testing data from scaled dataframe
testdfm <- newdf[-ind,]
#Checking the dimension of the scaled testing data
dim(testdfm)
```

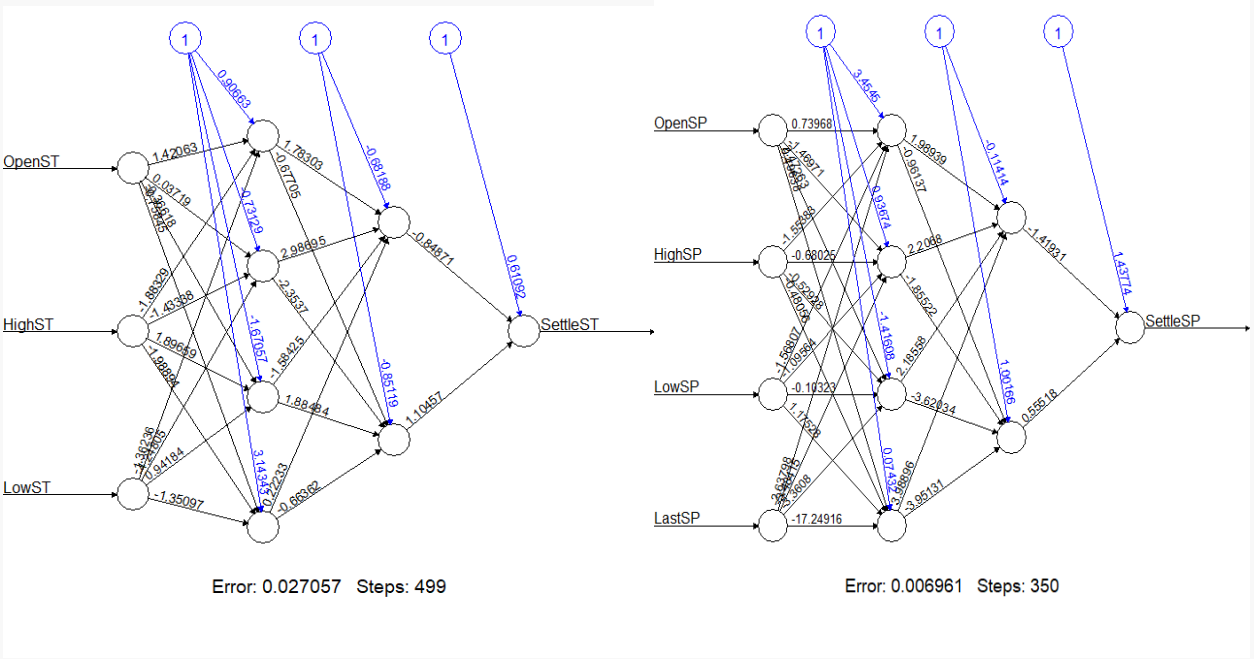
```
## [1] 205  61
```

```r
###We will consider some configuration for neural network like follows,
###input layer (predicting variable – x1,x2,….,xn) -> number of hidden layers
-> output layer(response variable – y)
#Fetching names of each variable in the dataset
varbls <- colnames(newdf)
###Creating formula in following format,
###y ~ x1 + x2 + ……. + xn
predictvarbls1 <- varbls[c(17,18,19,20)]
predictvarbls2 <- varbls[c(25,26,27,28)]
predictvarbls3 <- varbls[c(48,49,50,51)]
predictvarbls4 <- varbls[c(56,57,58)]

predictvarbls1 <- paste(predictvarbls1,collapse = "+")
predictvarbls2 <- paste(predictvarbls2,collapse = "+")
```

```
predictvarbls3 <- paste(predictvarbls3,collapse = "+")
predictvarbls4 <- paste(predictvarbls4,collapse = "+")

form1 <- as.formula(paste("SettleSOY~",predictvarbls1,collapse = "+"))
form2 <- as.formula(paste("SettleNG~",predictvarbls2,collapse = "+"))
form3 <- as.formula(paste("SettleSP~",predictvarbls3,collapse = "+"))
form4 <- as.formula(paste("SettleST~",predictvarbls4,collapse = "+"))
###Applying neural network model
neuralmode1 <- neuralnet(formula = form1,hidden = c(4,2),
linear.output=T,data=traindfm)
neuralmode2 <- neuralnet(formula = form2,hidden = c(4,2),
linear.output=T,data=traindfm)
neuralmode3 <- neuralnet(formula = form3,hidden = c(4,2),
linear.output=T,data=traindfm)
neuralmode4 <- neuralnet(formula = form4,hidden = c(4,2),
linear.output=T,data=traindfm)
#Plotting neural network
par(mfrow = c(2,2))
plot(neuralmode1)
plot(neuralmode2)
plot(neuralmode3)
plot(neuralmode4)
```







Note:The above graphs are manually copied, it's working properly in RStudio but it's – not at all printed in the markdown, by look to fix it I lost all time and went beyond the – due date, sorry for that.

```
##By having a look at the neural network graph we can see there are four
layers, where as in first layer predicting variables are available, seconf
and third layers are occupied by hidden layer (4,2) and the final layer is
occupied by the response variable.
###Prediction for test data set
#predictionm <- compute(neuralmode, testdfm[,c(-22,-30,-53,-59)])
predictionm1 <- compute(neuralmode1, testdfm[,c(17,18,19,20)])
predictionm2 <- compute(neuralmode2, testdfm[,c(25,26,27,28)])
predictionm3 <- compute(neuralmode3, testdfm[,c(48,49,50,51)])
predictionm4 <- compute(neuralmode4, testdfm[,c(56,57,58)])
###Structure of prediction of test dataset
str(predictionm1)
```

```
## List of 2
##  $ neurons   :List of 3
##   ..$ : num [1:205, 1:5] 1 1 1 1 1 1 1 1 1 1 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:205] "2" "3" "4" "6" ...
##   .. .. ..$ : chr [1:5] "1" "OpenSOY" "HighSOY" "LowSOY" ...
##   ..$ : num [1:205, 1:5] 1 1 1 1 1 1 1 1 1 1 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:205] "2" "3" "4" "6" ...
```

```
##   .. .. ..$ : NULL
##   ..$ : num [1:205, 1:3] 1 1 1 1 1 1 1 1 1 1 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:205] "2" "3" "4" "6" ...
##   .. .. ..$ : NULL
##  $ net.result: num [1:205, 1] 0.692 0.692 0.692 0.703 0.725 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:205] "2" "3" "4" "6" ...
##   .. ..$ : NULL
```

```
str(predictionm2)
```

```
## List of 2
##  $ neurons   :List of 3
##   ..$ : num [1:205, 1:5] 1 1 1 1 1 1 1 1 1 1 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:205] "2" "3" "4" "6" ...
##   .. .. ..$ : chr [1:5] "1" "OpenNG" "HighNG" "LowNG" ...
##   ..$ : num [1:205, 1:5] 1 1 1 1 1 1 1 1 1 1 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:205] "2" "3" "4" "6" ...
##   .. .. ..$ : NULL
##   ..$ : num [1:205, 1:3] 1 1 1 1 1 1 1 1 1 1 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:205] "2" "3" "4" "6" ...
##   .. .. ..$ : NULL
##  $ net.result: num [1:205, 1] 0.245 0.245 0.245 0.204 0.127 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:205] "2" "3" "4" "6" ...
##   .. ..$ : NULL
```

```
str(predictionm3)
```

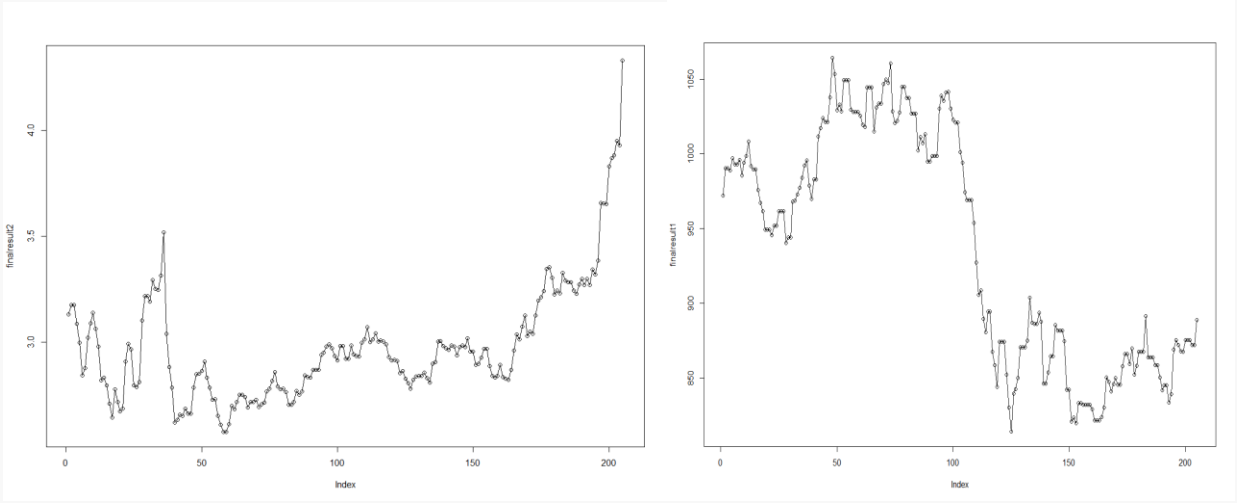
```
## List of 2
##  $ neurons   :List of 3
##   ..$ : num [1:205, 1:5] 1 1 1 1 1 1 1 1 1 1 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:205] "2" "3" "4" "6" ...
##   .. .. ..$ : chr [1:5] "1" "OpenSP" "HighSP" "LowSP" ...
##   ..$ : num [1:205, 1:5] 1 1 1 1 1 1 1 1 1 1 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:205] "2" "3" "4" "6" ...
##   .. .. ..$ : NULL
##   ..$ : num [1:205, 1:3] 1 1 1 1 1 1 1 1 1 1 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:205] "2" "3" "4" "6" ...
##   .. .. ..$ : NULL
##  $ net.result: num [1:205, 1] 0.0306 0.0306 0.0306 0.0564 0.0631 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:205] "2" "3" "4" "6" ...
##   .. ..$ : NULL
```

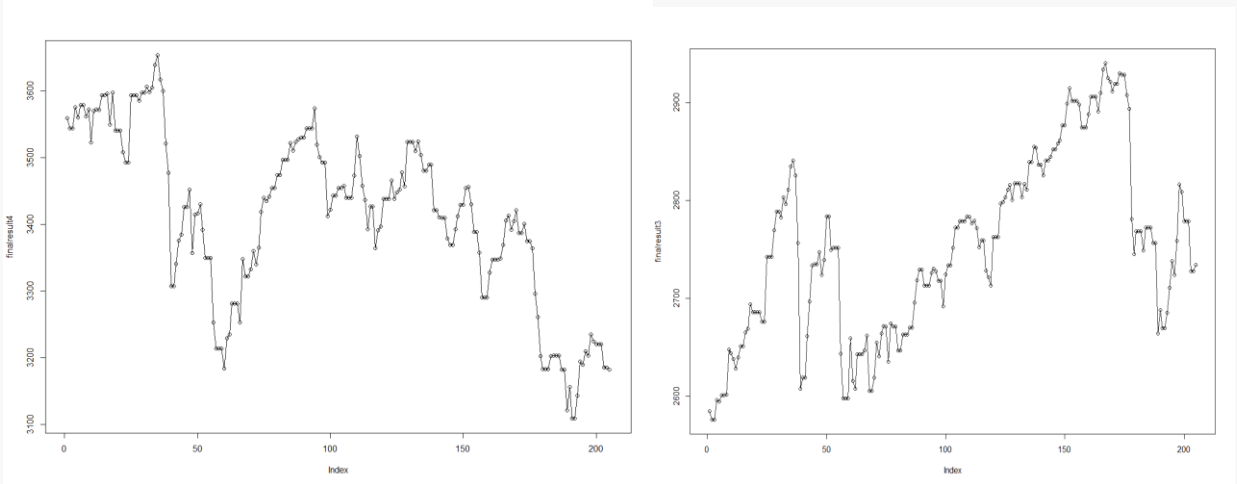
```r
str(predictionm4)
```

```
## List of 2
##  $ neurons   :List of 3
##   ..$ : num [1:205, 1:4] 1 1 1 1 1 1 1 1 1 1 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:205] "2" "3" "4" "6" ...
##   .. .. ..$ : chr [1:4] "1" "OpenST" "HighST" "LowST"
##   ..$ : num [1:205, 1:5] 1 1 1 1 1 1 1 1 1 1 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:205] "2" "3" "4" "6" ...
##   .. .. ..$ : NULL
##   ..$ : num [1:205, 1:3] 1 1 1 1 1 1 1 1 1 1 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:205] "2" "3" "4" "6" ...
##   .. .. ..$ : NULL
##  $ net.result: num [1:205, 1] 0.812 0.812 0.812 0.84 0.853 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:205] "2" "3" "4" "6" ...
##   .. ..$ : NULL
```

```r
predcm1 <- predictionm1$net.result
predcm2 <- predictionm2$net.result
predcm3 <- predictionm3$net.result
predcm4 <- predictionm4$net.result
predictionm1 <- predcm1*(max(testdfm$SettleSOY)-
min(testdfm$SettleSOY))+min(testdfm$SettleSOY)
predictionm2 <- predcm2*(max(testdfm$SettleNG)-
min(testdfm$SettleNG))+min(testdfm$SettleNG)
predictionm3 <- predcm3*(max(testdfm$SettleSP)-
min(testdfm$SettleSP))+min(testdfm$SettleSP)
predictionm4 <- predcm4*(max(testdfm$SettleST)-
min(testdfm$SettleST))+min(testdfm$SettleST)

actualval1 <- (testdfm$SettleSOY)*(max(testdfm$SettleSOY)-
min(testdfm$SettleSOY))+min(testdfm$SettleSOY)
actualval2 <- (testdfm$SettleNG)*(max(testdfm$SettleNG)-
min(testdfm$SettleNG))+min(testdfm$SettleNG)
actualval3 <- (testdfm$SettleSP)*(max(testdfm$SettleSP)-
min(testdfm$SettleSP))+min(testdfm$SettleSP)
actualval4 <- (testdfm$SettleST)*(max(testdfm$SettleST)-
min(testdfm$SettleST))+min(testdfm$SettleST)

#Mean Square Error
MSE1 <- sum((predictionm1-actualval1)^2)/nrow(testdfm)
MSE2 <- sum((predictionm2-actualval2)^2)/nrow(testdfm)
MSE3 <- sum((predictionm2-actualval3)^2)/nrow(testdfm)
MSE4 <- sum((predictionm2-actualval4)^2)/nrow(testdfm)

MSE1
```

```
## [1] 0.0002134710239

MSE2

## [1] 0.000074251808

MSE3

## [1] 0.1906089505

MSE4

## [1] 0.2810563307

##We could see the error level is too low for commodity where ANN suits best
-
#for commodity forecasting and eventhough there is a significant error for
stock -
#but still ANN fits well when compared to ARIMA mode, so we can say ANN
outperforms -
#ARIMA model in terms of forecasting.


final1 <- (testdfm$SettleSOY-predictionm1)
final2 <- (testdfm$SettleNG-predictionm2)
final3 <- (testdfm$SettleSP-predictionm3)
final4 <- (testdfm$SettleST-predictionm4)

finalresult1 <- (testing$SettleSOY + final1)
finalresult2 <- (testing$SettleNG + final2)
finalresult3 <- (testing$SettleSP + final3)
finalresult4 <- (testing$SettleST + final4)

#Let's forecast how the commodity and stock performs,
par(mfrow = c(2,2))
plot(finalresult1,type = "o")
plot(finalresult2,type = "o")
plot(finalresult3,type = "o")
plot(finalresult4,type = "o")
#discussion below
```





Note:The above graphs are manually copied, it's working properly in RStudio but it's – not at all printed in the markdown, by look to fix it I lost all time and went beyond the – due date, sorry for that.

After looking into the graph let's forecast and explain our insights,

Insight1: Soybean is now going to perform very well, from the above graph we coul see that soy price will go down intially which - will be the best time to buy and and reach peak soon, so it will be the best time to sell, so it will be best to - invest in the spot market so that we can sell easily.

Insight2: Natual gas is going to perform very as per the forecast so it will be better to invest a significant amount of money - in it than others, in the mid of forecasting range it gonna reach its lowest price when we could buy and at the end of- forecast range it gonna reach highest price whenwe could sell to get the higher returns.

Insight3: S&P500 is predicted to be performs well as per the forecast, where initially it goes down where as price mutual fund- that depends on S&P 500 also will decrease which is the best time to buy mutual funds and it gonna perfrom well - where it gradually increase and reach the highst poin near the end of the foreccast range which will be the best time to sell.

Insight4: Looking into the forecast Stoxx50 will not going to perform well so it better to avoid investing in ETF's that depends- on stoxx50.Because the price seems to be higher initiall and there is no hope that will increase after have a look at the foresat- graph.

So, these are the insights of the portfolio manager on the investments customer interested in. But as no prediction model is the best our forecast might be subjectable to vary from the actual so as i told before it's just a prediction.

References:

[1] https://www.investopedia.com/terms/p/portfoliomanager.asp.

[2} Zangeneh, L., 2014. Investigating the Challenges of Data, Pricing and Modelling to Enable Agent Based Simulation of the Credit Default Swap Market (Doctoral dissertation, UCL (University College London)).

[3] Kohzadi, N., Boyd, M.S., Kermanshahi, B. and Kaastra, I., 1996. A comparison of artificial neural network and time series models for forecasting commodity prices. Neurocomputing, 10(2), pp.169-181.

[4] Indro, D.C., Jiang, C.X., Patuwo, B.E. and Zhang, G.P., 1999. Predicting mutual fund performance using artificial neural networks. Omega, 27(3), pp.373-380.

[5] J. T. Yao, C. L. Tan, and H. L. Poh, "Neural networks for technical analysis: a study on KLCI," International Journal of Theoretical and Applied Finance, vol. 2, no. 2, pp. 221–241, 1999.

[6] J. V. Hansen, J. B. Mcdonald, and R. D. Nelson, "Time series prediction with genetic-algorithm designed neural networks: an empirical comparison with modern statistical models," Computational Intelligence, vol. 15, no. 3, pp. 171–184, 1999.

[7] Y. B. Wijaya, S. Kom, and T. A. Napitupulu, "Stock price prediction: Comparison of Arima and artificial neural network methods—an Indonesia stock's case," in Proceedings of the 2nd International Conference on Advances in Computing, Control and Telecommunication Technologies (ACT '10), pp. 176–179, Jakarta, Indonesia, December 2010.

[8] Comparison of ARIMA and Artificial Neural Networks Models for Stock Price Prediction Ayodele Ariyo Adebiyi,1 Aderemi Oluyinka Adewumi,1 and Charles Korede Ayo2, 2014

[9] KNOEPFEL, I. and BRABECK-LETMATHE, P., 2012. Commodities. Sustainable Investing for Institutional Investors: Risks, Regulations and Strategies. pp. 201-210.

[10] https://www.investopedia.com

[11] Dunis, Christian & Laws, Jason & Evans, Ben & John, Liverpool. (2006). Modelling and trading the soybean-oil crush spread with recurrent and higher order networks: A comparative analysis. Neural Network World. 16. 10.4018/978-1-59904-897-0.ch016.

[12] DE LIMA, D.P., FIORIOLLI, J.C., PADULA, A.D. and PUMI, G., 2018. The impact of Chinese imports of soybean on port infrastructure in Brazil: A study based on the concept of the "Bullwhip Effect". Journal of Commodity Markets, 9, pp. 55-76.

[13] Ghosh, A. (1993), Hedging with stock index futures: Estimation and forecasting with error correction model. J. Fut. Mark., 13: 743-752. doi:10.1002/fut.3990130703

[14] TUNARU, R.S., 2018. Dividend derivatives. Quantitative Finance, 18(1), pp. 63-81.