DATA 624 Spring 2019: Project-1

Ahmed Sajjad, Harpreet Shoker, Jagruti Solao, Chad Smith, Todd Weigel

March 19, 2019

#### Introduction

Data given is a de-identified Excel spreadsheet. Assignment is to perform the appropriate analysis to forecast several series for 140 periods.We are having 1622 periods for our analysis.We need to create model and forecast values for the following series.

S01 – Forecast Var01, Var02 S02 – Forecast Var02, Var03 S03 – Forecast Var05, Var07 S04 – Forecast Var01, Var02 S05 – Forecast Var02, Var03 S06 – Forecast Var05, Var07

Loading all the required libraries.

library(utils)  
library('readxl')  
library('xlsx')  
library(tidyr)  
library(dplyr)  
library(ggplot2)  
library(forecast)  
library(fma)  
library(fpp2)  
library(tseries)  
library(gridExtra)  
library(ggcorrplot)  
library(astsa)  
library(janitor)  
library(timeDate)

#### Exploratory Data Analysis and Data Cleaning

Importing data from Excel, and dates are in a numeric format. We are using as.Date to import these, we simply need to set the origin date and for Excel on Windows, the origin date is December 30, 1899 for dates after 1900.

project\_in\_df <- data.frame(read\_excel("Project1data.xls", sheet = "Set for Class"))  
project\_in\_df = mutate(project\_in\_df, datetime=as.Date(SeriesInd, origin="1899-12-30"))  
project\_in\_df = project\_in\_df[c(1, 8, 2, 3, 4, 5, 6, 7)]  
  
head(project\_in\_df)

## SeriesInd datetime group Var01 Var02 Var03 Var05 Var07  
## 1 40669 2011-05-06 S03 30.64286 123432400 30.34 30.49 30.57286  
## 2 40669 2011-05-06 S02 10.28000 60855800 10.05 10.17 10.28000  
## 3 40669 2011-05-06 S01 26.61000 10369300 25.89 26.20 26.01000  
## 4 40669 2011-05-06 S06 27.48000 39335700 26.82 27.02 27.32000  
## 5 40669 2011-05-06 S05 69.26000 27809100 68.19 68.72 69.15000  
## 6 40669 2011-05-06 S04 17.20000 16587400 16.88 16.94 17.10000

nrow(project\_in\_df)

## [1] 10572

# Create separate dataframes for each group  
group\_S01\_df = filter(project\_in\_df, group == 'S01')  
group\_S02\_df = filter(project\_in\_df, group == 'S02')  
group\_S03\_df = filter(project\_in\_df, group == 'S03')  
group\_S04\_df = filter(project\_in\_df, group == 'S04')  
group\_S05\_df = filter(project\_in\_df, group == 'S05')  
group\_S06\_df = filter(project\_in\_df, group == 'S06')  
  
  
#remove forecast cells for now  
#project\_in\_df <- project\_in\_df[1:9732,]  
group\_S01\_df <- slice(group\_S01\_df, 1:1622)   
group\_S02\_df <- slice(group\_S02\_df, 1:1622)   
group\_S03\_df <- slice(group\_S03\_df, 1:1622)   
group\_S04\_df <- slice(group\_S04\_df, 1:1622)   
group\_S05\_df <- slice(group\_S05\_df, 1:1622)   
group\_S06\_df <- slice(group\_S06\_df, 1:1622)  
  
  
#Change numeric sequence to actual dates for graphing  
#note, we put back to sequence before writing the files back out to excel  
group\_S01\_df[,1] <- excel\_numeric\_to\_date(group\_S01\_df[,1])  
group\_S02\_df[,1] <- excel\_numeric\_to\_date(group\_S02\_df[,1])  
group\_S03\_df[,1] <- excel\_numeric\_to\_date(group\_S03\_df[,1])  
group\_S04\_df[,1] <- excel\_numeric\_to\_date(group\_S04\_df[,1])  
group\_S05\_df[,1] <- excel\_numeric\_to\_date(group\_S05\_df[,1])  
group\_S06\_df[,1] <- excel\_numeric\_to\_date(group\_S06\_df[,1])  
  
#check for rows with NA's  
group\_S01\_df[rowSums(is.na(group\_S01\_df))>0,]

## SeriesInd datetime group Var01 Var02 Var03 Var05 Var07  
## 1537 2017-06-11 2017-06-11 S01 NA 7329600 NA NA NA  
## 1538 2017-06-12 2017-06-12 S01 NA 6121400 NA NA NA  
## 1607 2017-09-19 2017-09-19 S01 58.83 6337000 NA NA NA  
## 1608 2017-09-22 2017-09-22 S01 59.28 3690900 NA NA NA

group\_S02\_df[rowSums(is.na(group\_S02\_df))>0,]

## SeriesInd datetime group Var01 Var02 Var03 Var05 Var07  
## 1537 2017-06-11 2017-06-11 S02 NA 38160300 NA NA NA  
## 1538 2017-06-12 2017-06-12 S02 NA 45801300 NA NA NA  
## 1607 2017-09-19 2017-09-19 S02 13.26 19465000 NA NA NA  
## 1608 2017-09-22 2017-09-22 S02 13.20 16234300 NA NA NA

group\_S03\_df[rowSums(is.na(group\_S03\_df))>0,]

## SeriesInd datetime group Var01 Var02 Var03 Var05 Var07  
## 1537 2017-06-11 2017-06-11 S03 NA 42343600 NA NA NA  
## 1538 2017-06-12 2017-06-12 S03 NA 50074700 NA NA NA  
## 1607 2017-09-19 2017-09-19 S03 95.43 32026000 NA NA NA  
## 1608 2017-09-22 2017-09-22 S03 97.19 38018600 NA NA NA

group\_S04\_df[rowSums(is.na(group\_S04\_df))>0,]

## SeriesInd datetime group Var01 Var02 Var03 Var05 Var07  
## 1537 2017-06-11 2017-06-11 S04 NA 9098800 NA NA NA  
## 1538 2017-06-12 2017-06-12 S04 NA 11188200 NA NA NA  
## 1607 2017-09-19 2017-09-19 S04 36.72 34330700 NA NA NA  
## 1608 2017-09-22 2017-09-22 S04 36.95 7785800 NA NA NA

group\_S05\_df[rowSums(is.na(group\_S05\_df))>0,]

## SeriesInd datetime group Var01 Var02 Var03 Var05 Var07  
## 795 2014-07-01 2014-07-01 S05 NA NA NA NA NA  
## 1537 2017-06-11 2017-06-11 S05 NA 16610900 NA NA NA  
## 1538 2017-06-12 2017-06-12 S05 NA 19331600 NA NA NA  
## 1607 2017-09-19 2017-09-19 S05 90.4 13191900 NA NA NA  
## 1608 2017-09-22 2017-09-22 S05 89.9 11766100 NA NA NA

group\_S06\_df[rowSums(is.na(group\_S06\_df))>0,]

## SeriesInd datetime group Var01 Var02 Var03 Var05 Var07  
## 20 2011-06-03 2011-06-03 S06 NA NA NA NA NA  
## 1537 2017-06-11 2017-06-11 S06 NA 19885500 NA NA NA  
## 1538 2017-06-12 2017-06-12 S06 NA 32570900 NA NA NA  
## 1607 2017-09-19 2017-09-19 S06 49.21 13222800 NA NA NA  
## 1608 2017-09-22 2017-09-22 S06 48.88 10644000 NA NA NA

From the above results can see missing values in all the variables,we need to impute NA’s with some value. Var01 has 14,Var02 has 2,Var03 has 26,Var05 has 26 and Var07 has 26 missing values.

Here we will be imputing these values with the next value in sequence, e.g., if row 1600 is NA for Var01, take value from row 1601 hopefully, since these are stock values, the value from the next day for the few missing values we have, should be close enough

removeNAs <- function(dfTs)  
{  
  
 while(nrow(dfTs[rowSums(is.na(dfTs))>0,]) > 0)  
 {   
   
 dfTs <- transmute(dfTs,   
 SeriesInd = SeriesInd,  
 Var01 = if\_else(is.na(Var01), lead(Var01), Var01),  
 Var02 = if\_else(is.na(Var02), lead(Var02), Var02),  
 Var03 = if\_else(is.na(Var03), lead(Var03), Var03),  
 Var05 = if\_else(is.na(Var05), lead(Var05), Var05),  
 Var07 = if\_else(is.na(Var07), lead(Var07), Var07))  
 }  
 print(dfTs[rowSums(is.na(dfTs))>0,])  
 return(dfTs)  
}  
  
group\_S01\_df <- removeNAs(group\_S01\_df)

## [1] SeriesInd Var01 Var02 Var03 Var05 Var07   
## <0 rows> (or 0-length row.names)

group\_S02\_df <- removeNAs(group\_S02\_df)

## [1] SeriesInd Var01 Var02 Var03 Var05 Var07   
## <0 rows> (or 0-length row.names)

group\_S03\_df <- removeNAs(group\_S03\_df)

## [1] SeriesInd Var01 Var02 Var03 Var05 Var07   
## <0 rows> (or 0-length row.names)

group\_S04\_df <- removeNAs(group\_S04\_df)

## [1] SeriesInd Var01 Var02 Var03 Var05 Var07   
## <0 rows> (or 0-length row.names)

group\_S05\_df <- removeNAs(group\_S05\_df)

## [1] SeriesInd Var01 Var02 Var03 Var05 Var07   
## <0 rows> (or 0-length row.names)

group\_S06\_df <- removeNAs(group\_S06\_df)

## [1] SeriesInd Var01 Var02 Var03 Var05 Var07   
## <0 rows> (or 0-length row.names)

summary(group\_S01\_df)

## SeriesInd Var01 Var02 Var03 Var05 Var07   
## Min. :2011-05-06 Min. :23.01 Min. : 1339900 Min. :22.28 Min. :22.55 Min. :22.50   
## 1st Qu.:2012-12-10 1st Qu.:29.85 1st Qu.: 5347550 1st Qu.:29.34 1st Qu.:29.60 1st Qu.:29.61   
## Median :2014-07-25 Median :35.66 Median : 7895050 Median :35.10 Median :35.36 Median :35.41   
## Mean :2014-07-23 Mean :39.43 Mean : 8907092 Mean :38.69 Mean :39.05 Mean :39.09   
## 3rd Qu.:2016-02-29 3rd Qu.:48.74 3rd Qu.:11321675 3rd Qu.:47.91 3rd Qu.:48.31 3rd Qu.:48.30   
## Max. :2017-10-13 Max. :62.31 Max. :48477500 Max. :61.59 Max. :62.29 Max. :62.14

summary(group\_S02\_df)

## SeriesInd Var01 Var02 Var03 Var05 Var07   
## Min. :2011-05-06 Min. : 9.03 Min. : 7128800 Min. : 8.82 Min. : 8.99 Min. : 8.92   
## 1st Qu.:2012-12-10 1st Qu.:12.18 1st Qu.: 27880300 1st Qu.:11.81 1st Qu.:12.01 1st Qu.:12.00   
## Median :2014-07-25 Median :14.07 Median : 39767500 Median :13.76 Median :13.91 Median :13.91   
## Mean :2014-07-23 Mean :13.99 Mean : 50633098 Mean :13.68 Mean :13.85 Mean :13.84   
## 3rd Qu.:2016-02-29 3rd Qu.:15.84 3rd Qu.: 59050900 3rd Qu.:15.52 3rd Qu.:15.72 3rd Qu.:15.67   
## Max. :2017-10-13 Max. :39.36 Max. :480879500 Max. :38.28 Max. :39.33 Max. :38.40

summary(group\_S03\_df)

## SeriesInd Var01 Var02 Var03 Var05 Var07   
## Min. :2011-05-06 Min. : 28.00 Min. : 13046400 Min. : 27.18 Min. : 27.48 Min. : 27.44   
## 1st Qu.:2012-12-10 1st Qu.: 54.12 1st Qu.: 55186050 1st Qu.: 52.89 1st Qu.: 53.34 1st Qu.: 53.53   
## Median :2014-07-25 Median : 76.24 Median : 85595400 Median : 74.92 Median : 75.66 Median : 75.76   
## Mean :2014-07-23 Mean : 77.65 Mean : 99387244 Mean : 76.15 Mean : 76.95 Mean : 76.91   
## 3rd Qu.:2016-02-29 3rd Qu.: 99.52 3rd Qu.:127036525 3rd Qu.: 97.57 3rd Qu.: 98.53 3rd Qu.: 98.51   
## Max. :2017-10-13 Max. :134.54 Max. :470249500 Max. :131.40 Max. :134.46 Max. :133.00

summary(group\_S04\_df)

## SeriesInd Var01 Var02 Var03 Var05 Var07   
## Min. :2011-05-06 Min. :11.80 Min. : 3468900 Min. :11.09 Min. :11.30 Min. :11.09   
## 1st Qu.:2012-12-10 1st Qu.:15.99 1st Qu.: 12918725 1st Qu.:15.66 1st Qu.:15.83 1st Qu.:15.80   
## Median :2014-07-25 Median :23.45 Median : 17000800 Median :22.77 Median :23.15 Median :23.28   
## Mean :2014-07-23 Mean :26.46 Mean : 20757818 Mean :25.83 Mean :26.15 Mean :26.14   
## 3rd Qu.:2016-02-29 3rd Qu.:36.42 3rd Qu.: 24015700 3rd Qu.:35.59 3rd Qu.:35.98 3rd Qu.:35.93   
## Max. :2017-10-13 Max. :52.62 Max. :233872100 Max. :51.64 Max. :52.28 Max. :52.37

summary(group\_S05\_df)

## SeriesInd Var01 Var02 Var03 Var05 Var07   
## Min. :2011-05-06 Min. : 56.99 Min. : 4156600 Min. : 55.94 Min. : 56.85 Min. : 56.57   
## 1st Qu.:2012-12-10 1st Qu.: 78.86 1st Qu.: 11199775 1st Qu.: 77.25 1st Qu.: 77.95 1st Qu.: 78.11   
## Median :2014-07-25 Median : 85.95 Median : 14575700 Median : 84.88 Median : 85.33 Median : 85.44   
## Mean :2014-07-23 Mean : 84.21 Mean : 16787776 Mean : 82.97 Mean : 83.59 Mean : 83.63   
## 3rd Qu.:2016-02-29 3rd Qu.: 90.99 3rd Qu.: 20014325 3rd Qu.: 89.71 3rd Qu.: 90.37 3rd Qu.: 90.41   
## Max. :2017-10-13 Max. :104.76 Max. :118023500 Max. :103.95 Max. :104.42 Max. :104.38

summary(group\_S06\_df)

## SeriesInd Var01 Var02 Var03 Var05 Var07   
## Min. :2011-05-06 Min. : 23.41 Min. : 4297000 Min. : 22.58 Min. : 22.91 Min. : 22.88   
## 1st Qu.:2012-12-10 1st Qu.: 30.56 1st Qu.: 15427725 1st Qu.: 29.99 1st Qu.: 30.32 1st Qu.: 30.26   
## Median :2014-07-25 Median : 37.14 Median : 21795200 Median : 36.67 Median : 36.94 Median : 36.98   
## Mean :2014-07-23 Mean : 40.21 Mean : 25733544 Mean : 39.51 Mean : 39.86 Mean : 39.87   
## 3rd Qu.:2016-02-29 3rd Qu.: 50.74 3rd Qu.: 31506875 3rd Qu.: 50.06 3rd Qu.: 50.45 3rd Qu.: 50.43   
## Max. :2017-10-13 Max. :195.18 Max. :144985700 Max. :189.36 Max. :195.00 Max. :189.72

From the above data summaries we can see all the null values are removed with apropiate data

# Select relevant columns for each group  
group\_S01\_df = select (group\_S01\_df, matches("SeriesInd|datetime|group|Var01|Var02"))  
group\_S02\_df = select (group\_S02\_df, matches("SeriesInd|datetime|group|Var02|Var03"))  
group\_S03\_df = select (group\_S03\_df, matches("SeriesInd|datetime|group|Var05|Var07"))  
group\_S04\_df = select (group\_S04\_df, matches("SeriesInd|datetime|group|Var01|Var02"))  
group\_S05\_df = select (group\_S05\_df, matches("SeriesInd|datetime|group|Var02|Var03"))  
group\_S06\_df = select (group\_S06\_df, matches("SeriesInd|datetime|group|Var05|Var07"))  
  
# Check number of rows  
print (c(nrow(group\_S01\_df), nrow(group\_S02\_df), nrow(group\_S03\_df), nrow(group\_S04\_df), nrow(group\_S05\_df), nrow(group\_S06\_df)))

## [1] 1622 1622 1622 1622 1622 1622

# Verify dataframes  
head(project\_in\_df, 20)

## SeriesInd datetime group Var01 Var02 Var03 Var05 Var07  
## 1 40669 2011-05-06 S03 30.64286 123432400 30.34000 30.49000 30.57286  
## 2 40669 2011-05-06 S02 10.28000 60855800 10.05000 10.17000 10.28000  
## 3 40669 2011-05-06 S01 26.61000 10369300 25.89000 26.20000 26.01000  
## 4 40669 2011-05-06 S06 27.48000 39335700 26.82000 27.02000 27.32000  
## 5 40669 2011-05-06 S05 69.26000 27809100 68.19000 68.72000 69.15000  
## 6 40669 2011-05-06 S04 17.20000 16587400 16.88000 16.94000 17.10000  
## 7 40670 2011-05-07 S03 30.79857 150476200 30.46428 30.65714 30.62571  
## 8 40670 2011-05-07 S02 11.24000 215620200 10.40000 10.45000 10.96000  
## 9 40670 2011-05-07 S01 26.30000 10943800 25.70000 25.95000 25.86000  
## 10 40670 2011-05-07 S06 28.24000 55416000 27.24000 27.27000 28.07000  
## 11 40670 2011-05-07 S05 69.45000 30174700 68.80000 69.19000 69.42000  
## 12 40670 2011-05-07 S04 17.23000 11718100 17.00000 17.22000 17.23000  
## 13 40671 2011-05-08 S03 30.74714 138040000 30.10714 30.62571 30.13857  
## 14 40671 2011-05-08 S02 11.46000 200070600 11.13000 11.21000 11.37000  
## 15 40671 2011-05-08 S01 26.03000 8933800 25.56000 25.90000 25.67000  
## 16 40671 2011-05-08 S06 28.33000 33237000 27.79000 28.03000 28.11000  
## 17 40671 2011-05-08 S05 70.60000 35044700 69.34000 69.45000 70.02000  
## 18 40671 2011-05-08 S04 17.30000 16422000 17.07000 17.17000 17.17000  
## 19 40672 2011-05-09 S03 30.28572 119282800 29.86429 30.25000 30.08286  
## 20 40672 2011-05-09 S02 11.69000 130201700 11.32000 11.46000 11.66000

head(group\_S01\_df, 20)

## SeriesInd Var01 Var02  
## 1 2011-05-06 26.61 10369300  
## 2 2011-05-07 26.30 10943800  
## 3 2011-05-08 26.03 8933800  
## 4 2011-05-09 25.84 10775400  
## 5 2011-05-10 26.34 12875600  
## 6 2011-05-13 26.49 11677000  
## 7 2011-05-14 26.03 21165300  
## 8 2011-05-15 25.16 18809200  
## 9 2011-05-16 25.00 22908400  
## 10 2011-05-17 24.77 20359100  
## 11 2011-05-21 24.78 11783800  
## 12 2011-05-22 24.61 13253500  
## 13 2011-05-23 24.88 18407800  
## 14 2011-05-24 24.17 21806400  
## 15 2011-05-27 23.82 20192100  
## 16 2011-05-28 23.90 19646700  
## 17 2011-05-29 23.66 19535100  
## 18 2011-05-30 23.51 20198600  
## 19 2011-05-31 23.37 23137300  
## 20 2011-06-03 23.08 13199000

head(group\_S02\_df, 20)

## SeriesInd Var02 Var03  
## 1 2011-05-06 60855800 10.05  
## 2 2011-05-07 215620200 10.40  
## 3 2011-05-08 200070600 11.13  
## 4 2011-05-09 130201700 11.32  
## 5 2011-05-10 130463000 11.46  
## 6 2011-05-13 170626200 11.78  
## 7 2011-05-14 162995900 11.72  
## 8 2011-05-15 154527100 11.47  
## 9 2011-05-16 116531200 11.51  
## 10 2011-05-17 96149800 11.55  
## 11 2011-05-21 65934000 11.46  
## 12 2011-05-22 71649500 11.50  
## 13 2011-05-23 121451400 11.01  
## 14 2011-05-24 161530100 10.41  
## 15 2011-05-27 121621500 10.61  
## 16 2011-05-28 108250500 11.07  
## 17 2011-05-29 105091600 11.22  
## 18 2011-05-30 203320000 11.27  
## 19 2011-05-31 159741200 10.70  
## 20 2011-06-03 82748200 10.93

head(group\_S03\_df, 20)

## SeriesInd Var05 Var07  
## 1 2011-05-06 30.49000 30.57286  
## 2 2011-05-07 30.65714 30.62571  
## 3 2011-05-08 30.62571 30.13857  
## 4 2011-05-09 30.25000 30.08286  
## 5 2011-05-10 30.04286 30.28286  
## 6 2011-05-13 30.40000 30.01571  
## 7 2011-05-14 29.88428 29.67429  
## 8 2011-05-15 29.69571 30.09286  
## 9 2011-05-16 30.01571 29.91857  
## 10 2011-05-17 30.13286 29.41857  
## 11 2011-05-21 29.76143 30.72000  
## 12 2011-05-22 30.70143 30.24714  
## 13 2011-05-23 30.29714 29.72429  
## 14 2011-05-24 29.54000 28.25000  
## 15 2011-05-27 28.93000 29.01000  
## 16 2011-05-28 29.42143 29.42000  
## 17 2011-05-29 29.55000 29.69714  
## 18 2011-05-30 29.27571 28.47000  
## 19 2011-05-31 28.72571 27.43714  
## 20 2011-06-03 27.48143 27.81857

head(group\_S04\_df, 20)

## SeriesInd Var01 Var02  
## 1 2011-05-06 17.20 16587400  
## 2 2011-05-07 17.23 11718100  
## 3 2011-05-08 17.30 16422000  
## 4 2011-05-09 16.90 31816300  
## 5 2011-05-10 16.76 15470000  
## 6 2011-05-13 16.83 16181900  
## 7 2011-05-14 16.86 15672400  
## 8 2011-05-15 16.98 16955600  
## 9 2011-05-16 17.23 16715600  
## 10 2011-05-17 17.25 18415000  
## 11 2011-05-21 16.96 15182600  
## 12 2011-05-22 16.68 14419500  
## 13 2011-05-23 16.58 21858400  
## 14 2011-05-24 16.21 25132800  
## 15 2011-05-27 16.11 19683700  
## 16 2011-05-28 16.17 43979400  
## 17 2011-05-29 16.49 41701000  
## 18 2011-05-30 15.96 30159500  
## 19 2011-05-31 15.67 39664600  
## 20 2011-06-03 15.30 29865700

head(group\_S05\_df, 20)

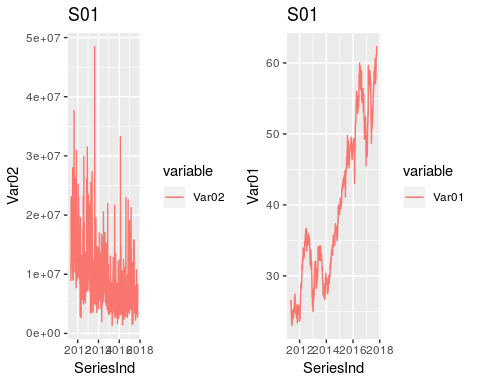
## SeriesInd Var02 Var03  
## 1 2011-05-06 27809100 68.19  
## 2 2011-05-07 30174700 68.80  
## 3 2011-05-08 35044700 69.34  
## 4 2011-05-09 27192100 69.42  
## 5 2011-05-10 24891800 69.22  
## 6 2011-05-13 30685000 69.65  
## 7 2011-05-14 31496700 69.52  
## 8 2011-05-15 24884400 69.26  
## 9 2011-05-16 18630800 69.35  
## 10 2011-05-17 29411900 68.65  
## 11 2011-05-21 26081900 68.42  
## 12 2011-05-22 34629500 67.93  
## 13 2011-05-23 39114000 66.50  
## 14 2011-05-24 39085500 66.00  
## 15 2011-05-27 29305100 65.69  
## 16 2011-05-28 34083300 65.50  
## 17 2011-05-29 35723500 65.00  
## 18 2011-05-30 37349800 64.57  
## 19 2011-05-31 40880500 64.02  
## 20 2011-06-03 37703000 65.35

head(group\_S06\_df, 20)

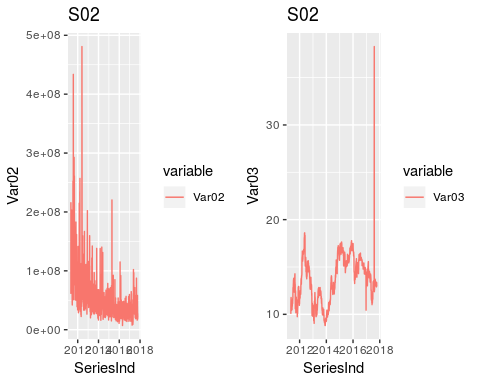
## SeriesInd Var05 Var07  
## 1 2011-05-06 27.02 27.32  
## 2 2011-05-07 27.27 28.07  
## 3 2011-05-08 28.03 28.11  
## 4 2011-05-09 28.12 29.13  
## 5 2011-05-10 28.90 28.86  
## 6 2011-05-13 29.09 28.80  
## 7 2011-05-14 28.47 28.08  
## 8 2011-05-15 27.99 28.58  
## 9 2011-05-16 28.50 28.99  
## 10 2011-05-17 28.82 28.08  
## 11 2011-05-21 27.93 28.28  
## 12 2011-05-22 28.02 27.82  
## 13 2011-05-23 27.98 28.00  
## 14 2011-05-24 27.97 27.26  
## 15 2011-05-27 27.45 27.66  
## 16 2011-05-28 27.45 26.99  
## 17 2011-05-29 26.90 28.20  
## 18 2011-05-30 28.28 28.45  
## 19 2011-05-31 28.44 28.43  
## 20 2011-06-03 28.96 28.79

Lets start looking at the data. First we are making line plots for all the groups and variables we will be forecasting.

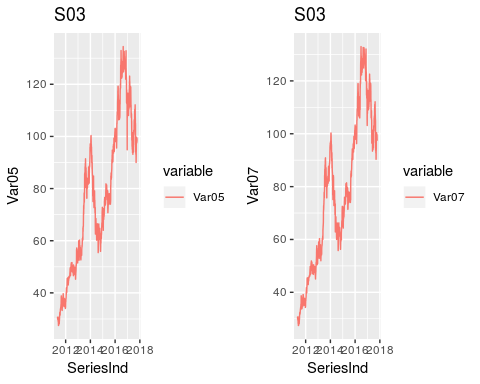
p1 <- ggplot(group\_S01\_df, aes(SeriesInd, y = Var02, color = variable)) +   
 ggtitle("S01") +  
 geom\_line(aes(y = Var02, col = "Var02"))  
  
p2 <- ggplot(group\_S01\_df, aes(SeriesInd, y = Var01, color = variable)) +   
 ggtitle("S01") +   
 geom\_line(aes(y = Var01, col = "Var01"))  
  
p3 <- ggplot(group\_S02\_df, aes(SeriesInd, y = Var02,color = variable))+  
 ggtitle("S02") +   
 geom\_line(aes(y = Var02, col = "Var02"))  
  
  
p4 <- ggplot(group\_S02\_df, aes(SeriesInd, y = Var03,color = variable))+  
 ggtitle("S02") +   
 geom\_line(aes(y = Var03, col = "Var03"))  
  
p5 <- ggplot(group\_S03\_df, aes(SeriesInd, y = Var05, color = variable)) +   
 geom\_line(aes(y = Var05, col = "Var05")) +  
 ggtitle("S03")   
  
p6 <- ggplot(group\_S03\_df, aes(SeriesInd, y = Var07, color = variable)) +   
 geom\_line(aes(y = Var07, col = "Var07")) +  
 ggtitle("S03")   
   
p7 <- ggplot(group\_S04\_df, aes(SeriesInd, y = Var01, color = variable)) +   
 geom\_line(aes(y = Var01, col = "Var01")) +   
 ggtitle("S04")   
p8 <- ggplot(group\_S04\_df, aes(SeriesInd, y = Var02,color = variable))+  
 ggtitle("S04") +   
 geom\_line(aes(y = Var02, col = "Var02"))  
  
  
p9 <- ggplot(group\_S05\_df, aes(SeriesInd, y = Var03, color = variable)) +   
 ggtitle("S05") +  
 geom\_line(aes(y = Var03, col = "Var03"))  
  
p10 <- ggplot(group\_S05\_df, aes(SeriesInd, y = Var02,color = variable))+  
 ggtitle("S05") +   
 geom\_line(aes(y = Var02, col = "Var02"))  
  
p11 <- ggplot(group\_S06\_df, aes(SeriesInd, y = Var05, color = variable)) +   
 geom\_line(aes(y = Var05, col = "Var05")) +  
 ggtitle("S06")  
  
p12 <- ggplot(group\_S06\_df, aes(SeriesInd, y = Var07, color = variable)) +   
 ggtitle("S06") +  
 geom\_line(aes(y = Var07, col = "Var07"))  
grid.arrange(p1, p2, nrow = 1)



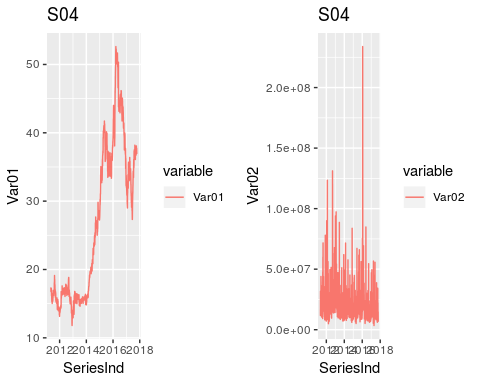
grid.arrange(p3, p4, nrow = 1)



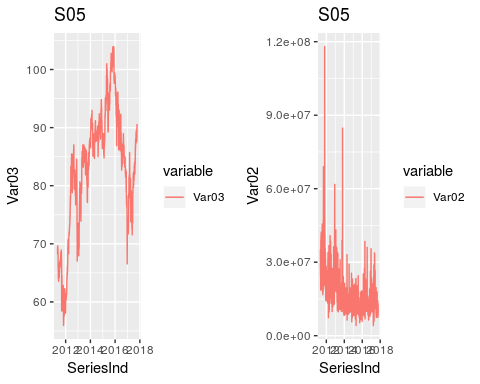
grid.arrange(p5, p6, nrow = 1)



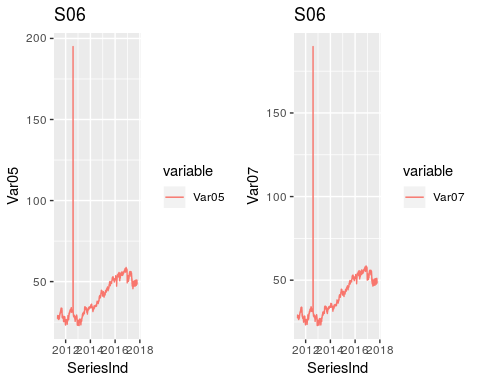
grid.arrange(p7,p8, nrow = 1)



grid.arrange(p9,p10, nrow = 1)



grid.arrange(p11,p12, nrow = 1)



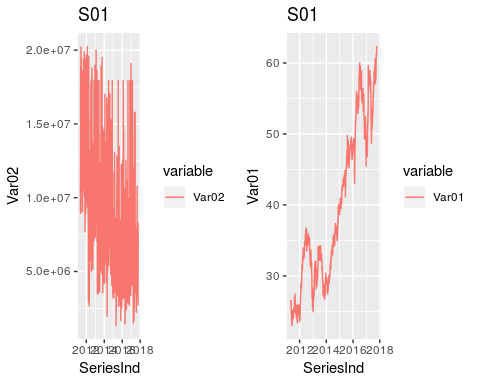
Comparing to other variable Var02 seems to be noisy than any other variable and having outliers. These outliers needs to be fixed before producing forecasts S03 and S06 variables Var05 and Var07 seems to be quite similar Also we see in S02 - Var03 and S06-Var05 and Var07 plot some outlier values that also needs to be fixed before forecasting. We can also observe some seasonality and trend pattern in Var01,Var03,Var05 and var07

Removing outliers We are using here IQR to fix the outliers in our data For missing values that lie outside the 1.5\*IQR limits, we are capping it by replacing those observations outside the lower limit with the value of 5th %ile and those that lie above the upper limit, with the value of 95th %ile.

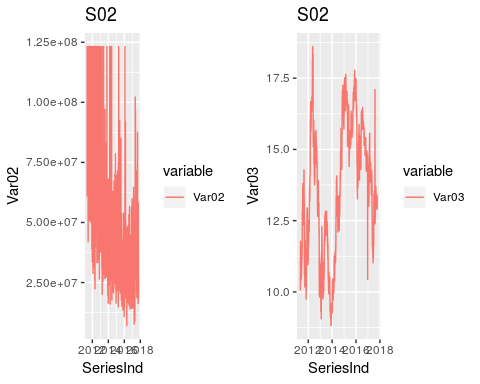
Remove\_Outlier <- function(x){  
 repeat{  
 qnt <- quantile(x, probs=c(.25, .75), na.rm = T)  
 caps <- quantile(x, probs=c(.05, .95), na.rm = T)  
 H <- 1.5 \* IQR(x, na.rm = T)  
 x[x < (qnt[1] - H)] <- caps[1]  
 x[x > (qnt[2] + H)] <- caps[2]  
   
 return(x)  
 if(x < (qnt[1] - H)){break}  
 if(x > (qnt[1] + H)){break}  
 }  
   
}  
group\_S01\_df$Var01=Remove\_Outlier(group\_S01\_df$Var01)  
group\_S01\_df$Var02=Remove\_Outlier(group\_S01\_df$Var02)  
group\_S02\_df$Var03=Remove\_Outlier(group\_S02\_df$Var03)  
group\_S02\_df$Var02=Remove\_Outlier(group\_S02\_df$Var02)  
group\_S04\_df$Var01=Remove\_Outlier(group\_S04\_df$Var01)  
group\_S04\_df$Var02=Remove\_Outlier(group\_S04\_df$Var02)  
group\_S05\_df$Var03=Remove\_Outlier(group\_S05\_df$Var03)  
group\_S05\_df$Var02=Remove\_Outlier(group\_S05\_df$Var02)  
group\_S06\_df$Var05=Remove\_Outlier(group\_S06\_df$Var05)  
group\_S06\_df$Var07=Remove\_Outlier(group\_S06\_df$Var07)

Lets plot again plots for the variables to check if the outliers are fixed

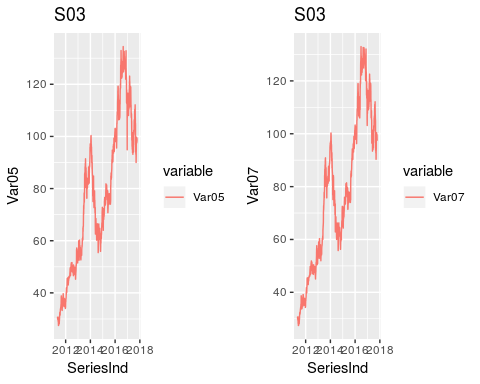
p1 <- ggplot(group\_S01\_df, aes(SeriesInd, y = Var02, color = variable)) +   
 ggtitle("S01") +  
 geom\_line(aes(y = Var02, col = "Var02"))  
  
p2 <- ggplot(group\_S01\_df, aes(SeriesInd, y = Var01, color = variable)) +   
 ggtitle("S01") +   
 geom\_line(aes(y = Var01, col = "Var01"))  
  
p3 <- ggplot(group\_S02\_df, aes(SeriesInd, y = Var02,color = variable))+  
 ggtitle("S02") +   
 geom\_line(aes(y = Var02, col = "Var02"))  
  
  
p4 <- ggplot(group\_S02\_df, aes(SeriesInd, y = Var03,color = variable))+  
 ggtitle("S02") +   
 geom\_line(aes(y = Var03, col = "Var03"))  
  
p5 <- ggplot(group\_S03\_df, aes(SeriesInd, y = Var05, color = variable)) +   
 geom\_line(aes(y = Var05, col = "Var05")) +  
 ggtitle("S03")   
  
p6 <- ggplot(group\_S03\_df, aes(SeriesInd, y = Var07, color = variable)) +   
 geom\_line(aes(y = Var07, col = "Var07")) +  
 ggtitle("S03")   
   
p7 <- ggplot(group\_S04\_df, aes(SeriesInd, y = Var01, color = variable)) +   
 geom\_line(aes(y = Var01, col = "Var01")) +   
 ggtitle("S04")   
p8 <- ggplot(group\_S04\_df, aes(SeriesInd, y = Var02,color = variable))+  
 ggtitle("S04") +   
 geom\_line(aes(y = Var02, col = "Var02"))  
  
  
p9 <- ggplot(group\_S05\_df, aes(SeriesInd, y = Var03, color = variable)) +   
 ggtitle("S05") +  
 geom\_line(aes(y = Var03, col = "Var03"))  
  
p10 <- ggplot(group\_S05\_df, aes(SeriesInd, y = Var02,color = variable))+  
 ggtitle("S05") +   
 geom\_line(aes(y = Var02, col = "Var02"))  
  
p11 <- ggplot(group\_S06\_df, aes(SeriesInd, y = Var05, color = variable)) +   
 geom\_line(aes(y = Var05, col = "Var05")) +  
 ggtitle("S06")  
  
p12 <- ggplot(group\_S06\_df, aes(SeriesInd, y = Var07, color = variable)) +   
 ggtitle("S06") +  
 geom\_line(aes(y = Var07, col = "Var07"))  
grid.arrange(p1, p2, nrow = 1)



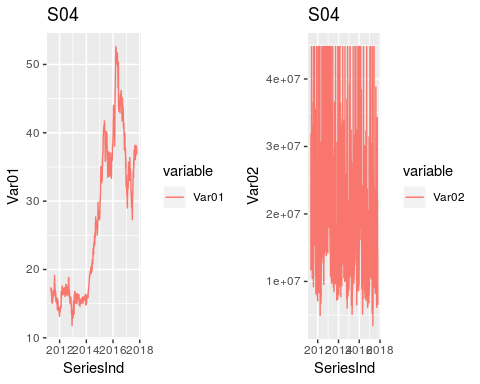
grid.arrange(p3, p4, nrow = 1)



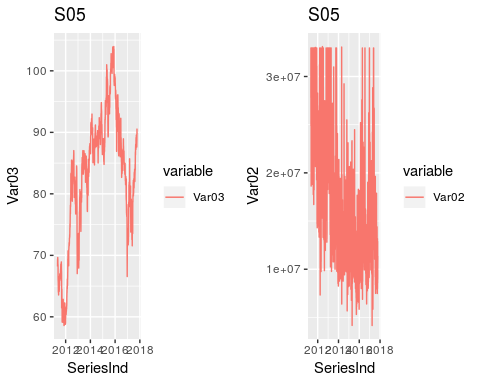
grid.arrange(p5, p6, nrow = 1)



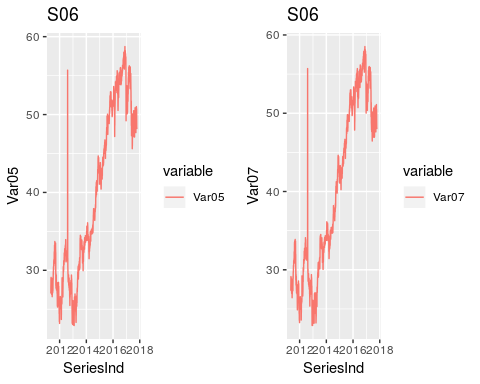
grid.arrange(p7,p8, nrow = 1)



grid.arrange(p9,p10, nrow = 1)



grid.arrange(p11,p12, nrow = 1)



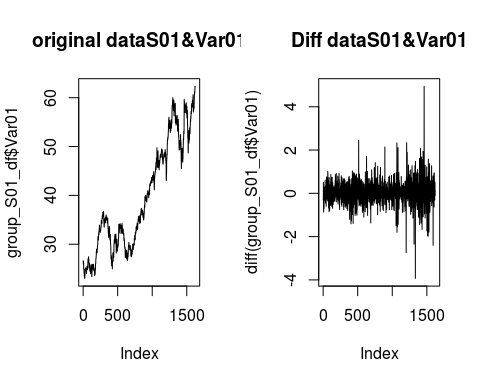
The plots looks it has removed the extreme outliers.

The plots for Var01,Var03,Var05,Var07 shows increase in general during the period of time. But there is no obvious pattern in the fluctuation. In other words, there might be seasonality, but an obvious upward trend. Also, the variance is not stable seeing from the plots and it seems to increase. Thus, we may use difference and logarithm or square root transformation on original data to stabilize the variance.

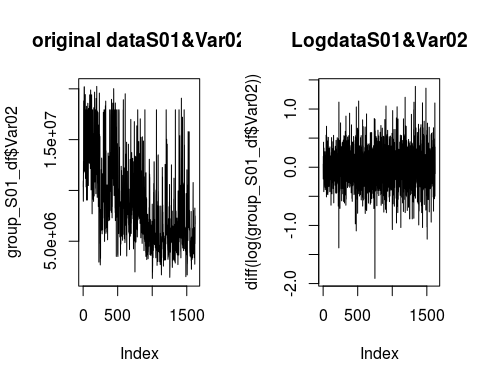
We are using here difference of each value over previous value and difference logarithm transformation for Var02

Lets create plots to have compare how the orignal data and transformed data

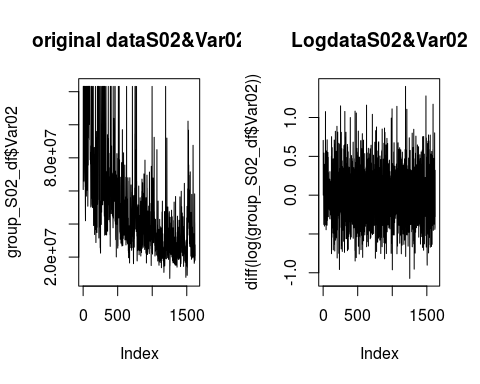
par(mfrow=c(1,2))  
p1 <- plot(group\_S01\_df$Var01, type = "l", main = "original dataS01&Var01")  
p2 <- plot(diff(group\_S01\_df$Var01), type = "l", main = "Diff dataS01&Var01")



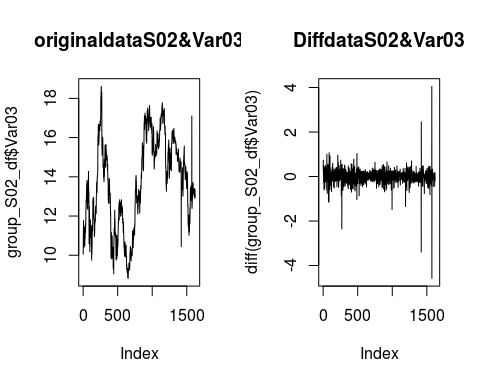
p3 <- plot(group\_S01\_df$Var02, type = "l", main = "original dataS01&Var02")  
p4 <- plot(diff(log(group\_S01\_df$Var02)), type = "l", main = "LogdataS01&Var02")



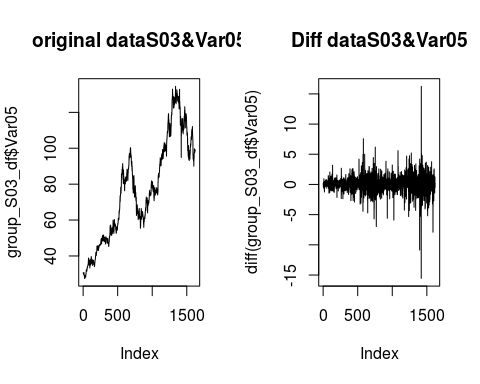
p5 <- plot(group\_S02\_df$Var02, type = "l", main = "original dataS02&Var02")  
p6 <- plot(diff(log(group\_S02\_df$Var02)), type = "l", main = "LogdataS02&Var02")



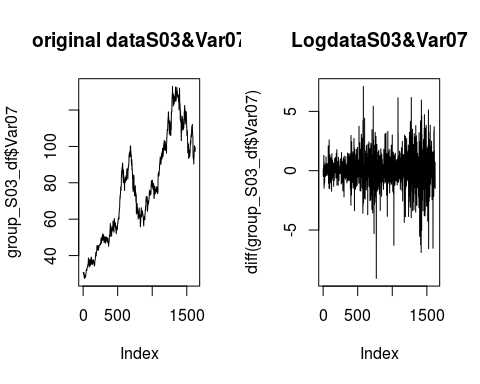
p7 <- plot(group\_S02\_df$Var03, type = "l", main = "originaldataS02&Var03")  
p8 <- plot(diff(group\_S02\_df$Var03), type = "l", main = "DiffdataS02&Var03")



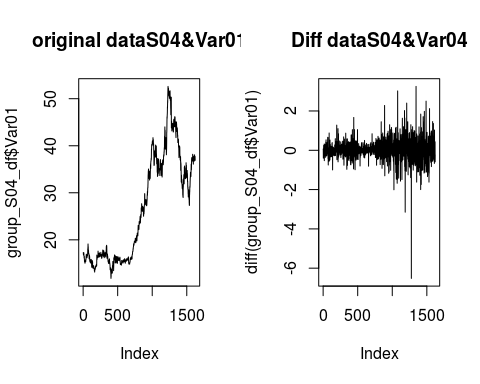
p9 <- plot(group\_S03\_df$Var05, type = "l", main = "original dataS03&Var05")  
p10 <- plot(diff(group\_S03\_df$Var05), type = "l", main = "Diff dataS03&Var05")



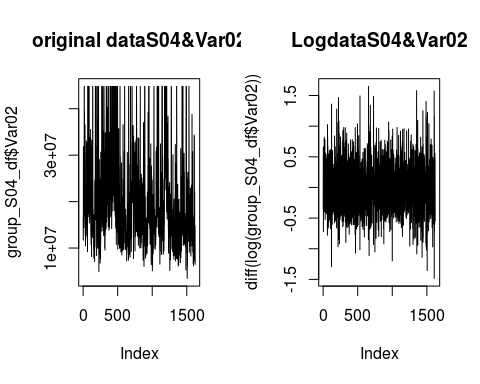
p11 <- plot(group\_S03\_df$Var07, type = "l", main = "original dataS03&Var07")  
p12 <- plot(diff(group\_S03\_df$Var07), type = "l", main = "LogdataS03&Var07")



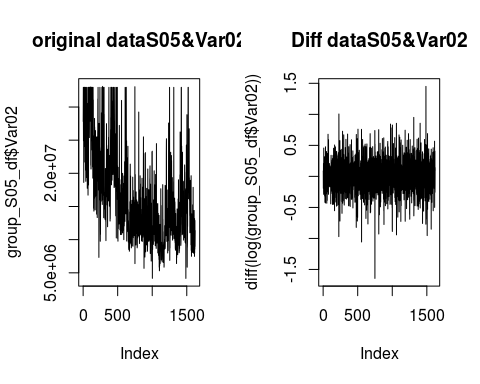
p13 <- plot(group\_S04\_df$Var01, type = "l", main = "original dataS04&Var01")  
p14 <- plot(diff(group\_S04\_df$Var01), type = "l", main = "Diff dataS04&Var04")



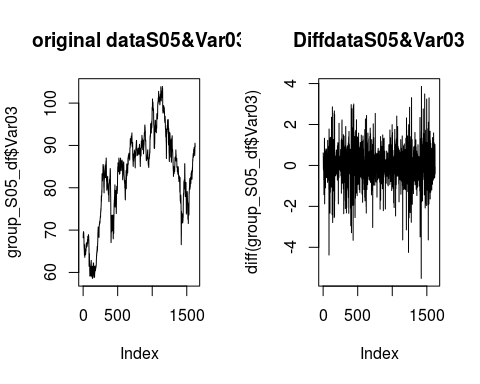
p15 <- plot(group\_S04\_df$Var02, type = "l", main = "original dataS04&Var02")  
p16 <- plot(diff(log(group\_S04\_df$Var02)), type = "l", main = "LogdataS04&Var02")



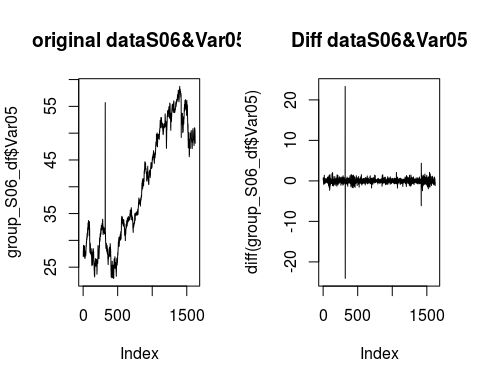
p17 <- plot(group\_S05\_df$Var02, type = "l", main = "original dataS05&Var02")  
p18 <- plot(diff(log(group\_S05\_df$Var02)), type = "l", main = "Diff dataS05&Var02")



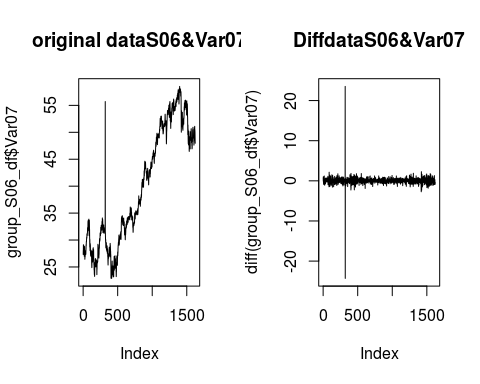
p19 <- plot(group\_S05\_df$Var03, type = "l", main = "original dataS05&Var03")  
p20 <- plot(diff(group\_S05\_df$Var03), type = "l", main = "DiffdataS05&Var03")



p21 <- plot(group\_S06\_df$Var05, type = "l", main = "original dataS06&Var05")  
p22 <- plot(diff(group\_S06\_df$Var05), type = "l", main = "Diff dataS06&Var05")



p23 <- plot(group\_S06\_df$Var07, type = "l", main = "original dataS06&Var07")  
p24 <- plot(diff(group\_S06\_df$Var07), type = "l", main = "DiffdataS06&Var07")



Data looks like it eliminated noise compared to the orginal. Data looks more stationary after applying differencing to Var01,Var03,Var05,Var07 and applying logarithm Var02.

#### Group S01 Forecast

#### Group S02 Forecast

#### Group S03 Forecast

#### Group S04 Forecast

#### Group S05 Forecast

###This Section is using ARIMA for now.

First let’s get the dataframe data into a timeseries for variable 02. The data is stored as a series of dates, 5 days with a break after, so essentially 52 weeks times 5 days will divide it up nicely into 7 years of data. We will then fit the data using auto.arima to find the appropriate values for P,D,Q (p = the number of lag observations, d = degree of differencing, and q = size of the moving average window, for lagged forecast errors ).

We see that a 1,1,3 model appears to be most appropriate.

library(astsa)  
h <- 140  
TsVar02 <- ts(group\_S05\_df$Var02, frequency = 5\*52)  
#TsVar02 <- ts(group\_S01\_df[,1:2])  
TsVar02 <- ts(group\_S05\_df$Var02, start = c(2011, 85), end = c(2017, 166), frequency = 260)  
  
arimaFitVar02 <- auto.arima(TsVar02, seasonal = FALSE)  
arimaFitVar02

## Series: TsVar02   
## ARIMA(1,1,2)   
##   
## Coefficients:  
## ar1 ma1 ma2  
## 0.7600 -1.3275 0.3646  
## s.e. 0.0619 0.0725 0.0587  
##   
## sigma^2 estimated as 1.545e+13: log likelihood=-27244.99  
## AIC=54497.99 AICc=54498.01 BIC=54519.6

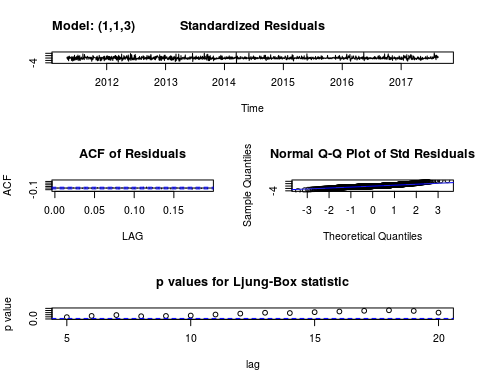
We will use the package Applied Statistical Time Series Analysis (astsa) and it’s wrapper module, sarima around the arima set of tools to do our forecasting.

Looking at the residuals from the package output, we see that they look like white noise, which is what we hope, as we don’t wish to see a pattern. The ACF of the residuals look acceptable too. There are a couple of small spikes just above the threshold of significance, but otherwise fine.

Note the box test results show that there may still be correlation on the residuals as the

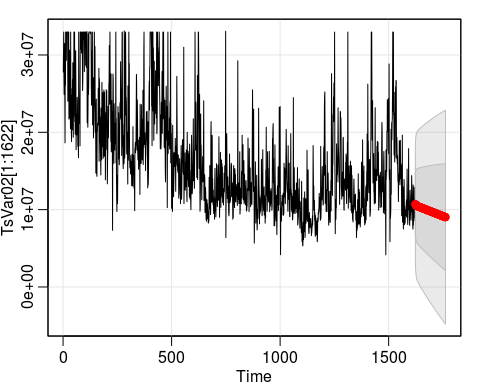
sarima(TsVar02, 1, 1, 3)

## initial value 15.326257   
## iter 2 value 15.241436  
## iter 3 value 15.194854  
## iter 4 value 15.192655  
## iter 5 value 15.192413  
## iter 6 value 15.192297  
## iter 7 value 15.192275  
## iter 8 value 15.190454  
## iter 9 value 15.188221  
## iter 10 value 15.188003  
## iter 11 value 15.187588  
## iter 12 value 15.187445  
## iter 13 value 15.187383  
## iter 14 value 15.187280  
## iter 15 value 15.186786  
## iter 16 value 15.186167  
## iter 17 value 15.186059  
## iter 18 value 15.185659  
## iter 19 value 15.185264  
## iter 20 value 15.185253  
## iter 21 value 15.185251  
## iter 22 value 15.185249  
## iter 23 value 15.185249  
## iter 24 value 15.185248  
## iter 25 value 15.185247  
## iter 26 value 15.185246  
## iter 27 value 15.185246  
## iter 27 value 15.185246  
## iter 27 value 15.185246  
## final value 15.185246   
## converged  
## initial value 15.184123   
## iter 2 value 15.184106  
## iter 3 value 15.184099  
## iter 4 value 15.184074  
## iter 5 value 15.183970  
## iter 6 value 15.183857  
## iter 7 value 15.183732  
## iter 8 value 15.183676  
## iter 9 value 15.183668  
## iter 10 value 15.183664  
## iter 11 value 15.183663  
## iter 11 value 15.183663  
## iter 11 value 15.183663  
## final value 15.183663   
## converged



## $fit  
##   
## Call:  
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,   
## Q), period = S), xreg = constant, optim.control = list(trace = trc, REPORT = 1,   
## reltol = tol))  
##   
## Coefficients:  
## ar1 ma1 ma2 ma3 constant  
## 0.7851 -1.3498 0.3657 0.0159 -1949.525  
## s.e. 0.0830 0.0870 0.0582 0.0334 14597.059  
##   
## sigma^2 estimated as 1.542e+13: log likelihood = -27244.87, aic = 54501.74  
##   
## $degrees\_of\_freedom  
## [1] 1636  
##   
## $ttable  
## Estimate SE t.value p.value  
## ar1 0.7851 0.0830 9.4631 0.0000  
## ma1 -1.3498 0.0870 -15.5194 0.0000  
## ma2 0.3657 0.0582 6.2796 0.0000  
## ma3 0.0159 0.0334 0.4772 0.6333  
## constant -1949.5249 14597.0593 -0.1336 0.8938  
##   
## $AIC  
## [1] 31.37287  
##   
## $AICc  
## [1] 31.37412  
##   
## $BIC  
## [1] 30.38933

fSarima <- sarima.for(TsVar02[1:1622], 140, 1, 1, 3, plot.all = TRUE)



fSarima$pred[1:20]

## [1] 10719434 10657546 10619192 10587296 10560330 10537127 10516797 10498661 10482200 10467016 10452809 10439346 10426453 10413994 10401866 10389991 10378309 10366775 10355354 10344018

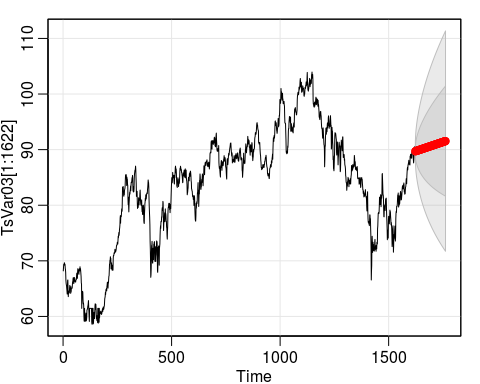
#save prediction into df  
group\_S05\_df[1623:1762,2] <- fSarima$pred

###ARIMA Fit for Var03 Doing the same process for Var03, we see that a 2,1,1 model is best.

library(astsa)  
h <- 140  
TsVar03 <- ts(group\_S05\_df$Var03, frequency = 5\*52)  
  
arimaFitVar03 <- auto.arima(TsVar03, seasonal = FALSE)  
arimaFitVar03

## Series: TsVar03   
## ARIMA(2,1,1)   
##   
## Coefficients:  
## ar1 ar2 ma1  
## 0.7785 -0.1368 -0.6654  
## s.e. 0.1458 0.0251 0.1461  
##   
## sigma^2 estimated as 0.8007: log likelihood=-2118.51  
## AIC=4245.01 AICc=4245.04 BIC=4266.58

fitArimaV03 <- sarima(TsVar03, 2, 1, 1, details = FALSE)  
  
   
##for some reason this blows up if in ts with frequency of 260  
fSarima <- sarima.for(TsVar03[1:1622], n.ahead = 140, 2, 1, 1, plot.all = TRUE)



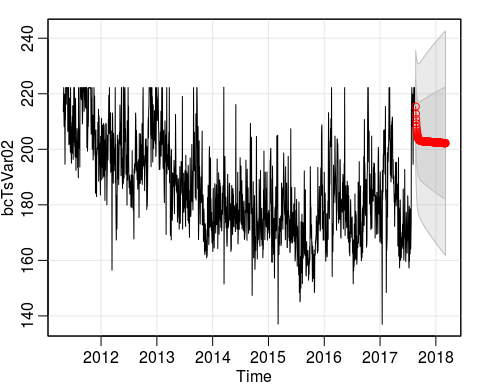
fSarima$pred[1:20]

## [1] 89.69658 89.70958 89.72489 89.73975 89.75394 89.76767 89.78114 89.79445 89.80770 89.82090 89.83408 89.84726 89.86042 89.87359 89.88675 89.89991 89.91308 89.92624 89.93940 89.95256

#save prediction into df  
group\_S05\_df[1623:1762,3] <- fSarima$pred

###Attempted a box cox transform to see how that worked out, really other than scale is pretty similar

bcTsVar02 <- BoxCox(TsVar02, lambda = "auto")  
fSarima <- sarima.for(bcTsVar02, 140, 1, 1, 3, plot.all = TRUE)



####Sanity checks for differencing. Run some difference tests, we see that one difference is absolutely needed, and perhaps two differences could be warranted for var03 as one order of difference gives .07 value for the test statistic (i.e.it is greater than .05), but it is questionable, as extra differencing in itself can lead to errors.

#Reset values  
TsVar02 <- ts(group\_S05\_df$Var02, frequency = 5\*52)  
TsVar03 <- ts(group\_S05\_df$Var03, frequency = 5\*52)  
  
library(urca)  
summary(ur.kpss(TsVar02))

##   
## #######################   
## # KPSS Unit Root Test #   
## #######################   
##   
## Test is of type: mu with 8 lags.   
##   
## Value of test-statistic is: 10.1001   
##   
## Critical value for a significance level of:   
## 10pct 5pct 2.5pct 1pct  
## critical values 0.347 0.463 0.574 0.739

summary(ur.kpss(diff(TsVar02)))

##   
## #######################   
## # KPSS Unit Root Test #   
## #######################   
##   
## Test is of type: mu with 8 lags.   
##   
## Value of test-statistic is: 0.0054   
##   
## Critical value for a significance level of:   
## 10pct 5pct 2.5pct 1pct  
## critical values 0.347 0.463 0.574 0.739

summary(ur.kpss(TsVar03))

##   
## #######################   
## # KPSS Unit Root Test #   
## #######################   
##   
## Test is of type: mu with 8 lags.   
##   
## Value of test-statistic is: 8.3755   
##   
## Critical value for a significance level of:   
## 10pct 5pct 2.5pct 1pct  
## critical values 0.347 0.463 0.574 0.739

summary(ur.kpss(diff(TsVar03)))

##   
## #######################   
## # KPSS Unit Root Test #   
## #######################   
##   
## Test is of type: mu with 8 lags.   
##   
## Value of test-statistic is: 0.0658   
##   
## Critical value for a significance level of:   
## 10pct 5pct 2.5pct 1pct  
## critical values 0.347 0.463 0.574 0.739

print("Do we actually need a second order of differencing?")

## [1] "Do we actually need a second order of differencing?"

summary(ur.kpss(diff(diff(TsVar03))))

##   
## #######################   
## # KPSS Unit Root Test #   
## #######################   
##   
## Test is of type: mu with 8 lags.   
##   
## Value of test-statistic is: 0.003   
##   
## Critical value for a significance level of:   
## 10pct 5pct 2.5pct 1pct  
## critical values 0.347 0.463 0.574 0.739

print("Checking for seasonality:")

## [1] "Checking for seasonality:"

nsdiffs(TsVar02, test = "seas")

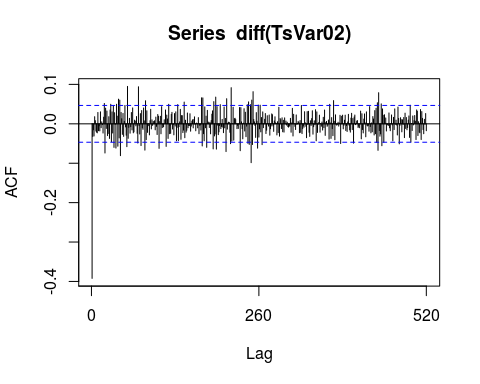
## [1] 0

nsdiffs(TsVar03, test = "seas")

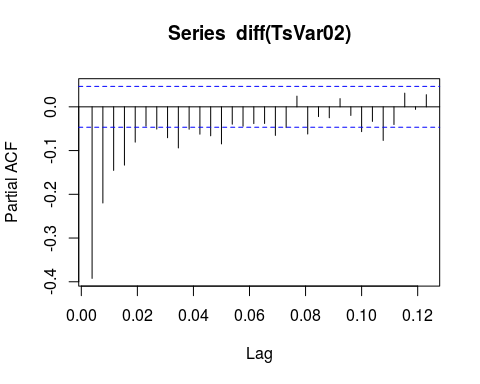
## [1] 0

Checking ACF grapsh for the p and q values.

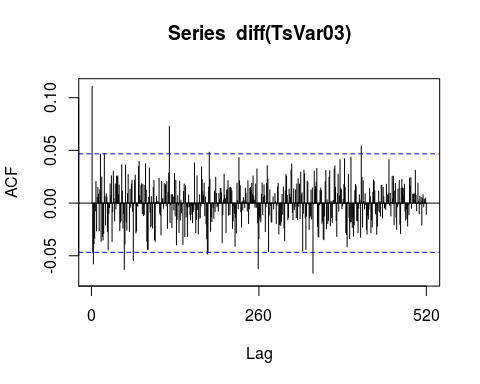
#113  
Acf(diff(TsVar02))



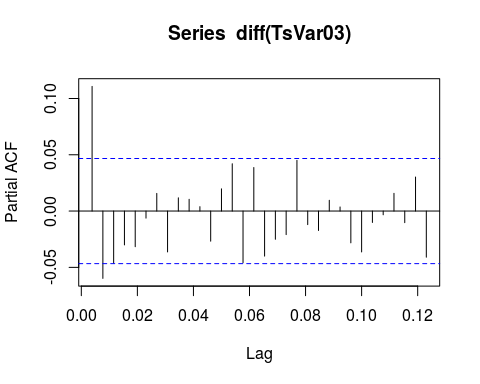
pacf(diff(TsVar02))



Acf(diff(TsVar03))



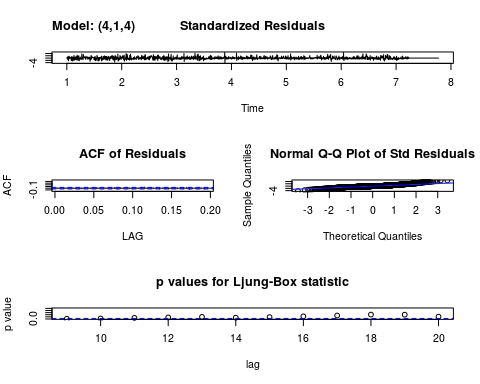
pacf(diff(TsVar03))

 Looking at the plots is seems like a (1,1,4) model might be better for Var02 considering the number of spikes in the acf graph. Rerunning using that model, the box test results look better, althought the AIC numbers are virtually unchanged.

The 2,1,1 based on the acf graphs looks reasonable for Var03.

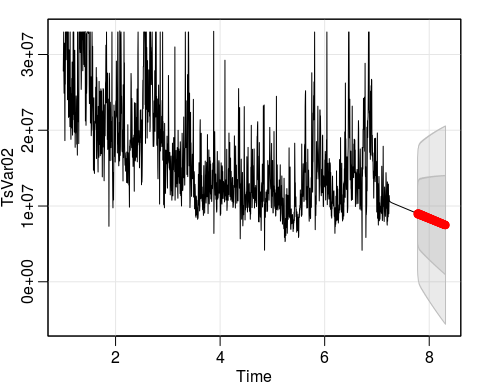
sarima(TsVar02, 4, 1, 4)

## initial value 15.281495   
## iter 2 value 15.183256  
## iter 3 value 15.144326  
## iter 4 value 15.139547  
## iter 5 value 15.132783  
## iter 6 value 15.130061  
## iter 7 value 15.129834  
## iter 8 value 15.129715  
## iter 9 value 15.129699  
## iter 10 value 15.129665  
## iter 11 value 15.129604  
## iter 12 value 15.129349  
## iter 13 value 15.128961  
## iter 14 value 15.128660  
## iter 15 value 15.128515  
## iter 16 value 15.128494  
## iter 17 value 15.128490  
## iter 18 value 15.128488  
## iter 19 value 15.128478  
## iter 20 value 15.128456  
## iter 21 value 15.128447  
## iter 22 value 15.128442  
## iter 23 value 15.128437  
## iter 24 value 15.128436  
## iter 25 value 15.128433  
## iter 26 value 15.128429  
## iter 27 value 15.128419  
## iter 28 value 15.128375  
## iter 29 value 15.128243  
## iter 30 value 15.128077  
## iter 31 value 15.127929  
## iter 32 value 15.127890  
## iter 33 value 15.127877  
## iter 34 value 15.127867  
## iter 35 value 15.127864  
## iter 36 value 15.127850  
## iter 37 value 15.127814  
## iter 38 value 15.127700  
## iter 39 value 15.127392  
## iter 40 value 15.127077  
## iter 41 value 15.126831  
## iter 42 value 15.126676  
## iter 43 value 15.126238  
## iter 44 value 15.125808  
## iter 45 value 15.125571  
## iter 46 value 15.125502  
## iter 47 value 15.125478  
## iter 48 value 15.125467  
## iter 49 value 15.125459  
## iter 50 value 15.125457  
## iter 51 value 15.125455  
## iter 52 value 15.125442  
## iter 53 value 15.125419  
## iter 54 value 15.125413  
## iter 55 value 15.125407  
## iter 56 value 15.125406  
## iter 56 value 15.125405  
## iter 56 value 15.125405  
## final value 15.125405   
## converged  
## initial value 15.125859   
## iter 2 value 15.125851  
## iter 3 value 15.125837  
## iter 4 value 15.125820  
## iter 5 value 15.125803  
## iter 6 value 15.125743  
## iter 7 value 15.125610  
## iter 8 value 15.125538  
## iter 9 value 15.125463  
## iter 10 value 15.125420  
## iter 11 value 15.125386  
## iter 12 value 15.125337  
## iter 13 value 15.125211  
## iter 14 value 15.125125  
## iter 15 value 15.124932  
## iter 16 value 15.124773  
## iter 17 value 15.124578  
## iter 18 value 15.124486  
## iter 19 value 15.124374  
## iter 20 value 15.124264  
## iter 21 value 15.124198  
## iter 22 value 15.124193  
## iter 23 value 15.124176  
## iter 24 value 15.124174  
## iter 25 value 15.124106  
## iter 26 value 15.124090  
## iter 27 value 15.124002  
## iter 28 value 15.123926  
## iter 29 value 15.123845  
## iter 30 value 15.123775  
## iter 31 value 15.123549  
## iter 32 value 15.123266  
## iter 33 value 15.123105  
## iter 34 value 15.122967  
## iter 35 value 15.122865  
## iter 36 value 15.122837  
## iter 37 value 15.122810  
## iter 38 value 15.122801  
## iter 39 value 15.122783  
## iter 40 value 15.122782  
## iter 41 value 15.122781  
## iter 42 value 15.122781  
## iter 43 value 15.122780  
## iter 44 value 15.122778  
## iter 45 value 15.122758  
## iter 46 value 15.122730  
## iter 47 value 15.122683  
## iter 48 value 15.122638  
## iter 49 value 15.122587  
## iter 50 value 15.122566  
## iter 51 value 15.122537  
## iter 52 value 15.122508  
## iter 53 value 15.122474  
## iter 54 value 15.122439  
## iter 55 value 15.122430  
## iter 56 value 15.122404  
## iter 57 value 15.122391  
## iter 58 value 15.122368  
## iter 59 value 15.122367  
## iter 60 value 15.122367  
## iter 61 value 15.122366  
## iter 61 value 15.122366  
## iter 61 value 15.122366  
## final value 15.122366   
## converged



## $fit  
##   
## Call:  
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,   
## Q), period = S), xreg = constant, optim.control = list(trace = trc, REPORT = 1,   
## reltol = tol))  
##   
## Coefficients:  
## ar1 ar2 ar3 ar4 ma1 ma2 ma3 ma4 constant  
## 0.2999 -0.5659 0.8107 -0.0548 -0.8881 0.6700 -1.1851 0.4586 -10983.979  
## s.e. 0.1043 0.0233 0.0756 0.0533 0.1006 0.0389 0.0506 0.0927 9657.103  
##   
## sigma^2 estimated as 1.36e+13: log likelihood = -29129.24, aic = 58278.48  
##   
## $degrees\_of\_freedom  
## [1] 1752  
##   
## $ttable  
## Estimate SE t.value p.value  
## ar1 0.2999 0.1043 2.8748 0.0041  
## ar2 -0.5659 0.0233 -24.3362 0.0000  
## ar3 0.8107 0.0756 10.7283 0.0000  
## ar4 -0.0548 0.0533 -1.0296 0.3034  
## ma1 -0.8881 0.1006 -8.8274 0.0000  
## ma2 0.6700 0.0389 17.2105 0.0000  
## ma3 -1.1851 0.0506 -23.4101 0.0000  
## ma4 0.4586 0.0927 4.9485 0.0000  
## constant -10983.9787 9657.1028 -1.1374 0.2555  
##   
## $AIC  
## [1] 31.25161  
##   
## $AICc  
## [1] 31.25281  
##   
## $BIC  
## [1] 30.27957

sarima.for(TsVar02, 140, 4, 1,4,plot.all = TRUE)



## $pred  
## Time Series:  
## Start = c(7, 203)   
## End = c(8, 82)   
## Frequency = 260   
## [1] 8982938 8894750 9011848 9017161 8877205 8926389 9012631 8891301 8848049 8965362 8916327 8801211 8886314 8925198 8792454 8800340 8899085 8808883 8734015 8836629 8825629 8702901 8754012 8818651 8704613 8676391 8776465 8720849 8625303 8705198 8727055 8608378 8624824 8704657 8616277 8558826 8649831 8628007 8522623 8574695 8621667 8516441 8500689 8585140 8525894 8447728 8521540 8529651 8424582 8447005 8510118 8425127 8382255 8461831 8431992 8342307 8393557 8425528 8329434 8323385 8393460 8332638 8269500 8336525 8333605 8241340 8267518 8315941 8235403 8204524 8273061 8237540 8161876 8210941 8230319 8143365 8144642 8201653 8140867 8090571 8150435 8138844 8058442 8086575 8122233 8046854 8025679 8083752 8044483 7981202 8027087 8036037 7958000 7964596 8009876 7950363 7910911 7963505 7945271 7875712 7904369 7929060 7859251 7845783 7894095 7852654 7800189 7842210 7842656 7773142 7783382 7818248 7760925 7730511 7775933 7752781 7693013 7721080 7736458 7672397 7664909 7704244 7661897 7618773 7656503 7650132 7588626 7601134 7626858 7572375 7549386 7587892 7561266 7510235 7536873 7544442 7486122 7483138 7514321 7472062  
##   
## $se  
## Time Series:  
## Start = c(7, 203)   
## End = c(8, 82)   
## Frequency = 260   
## [1] 3690207 3990556 4181537 4304265 4384019 4457494 4517493 4555493 4591380 4629718 4656021 4677557 4705247 4728814 4745170 4765914 4788942 4804546 4820626 4842455 4859560 4873195 4892466 4911618 4924924 4941085 4961249 4975926 4989652 5008969 5025879 5038615 5055644 5074400 5087733 5102243 5121349 5136466 5149335 5167077 5184259 5196886 5212343 5230727 5244449 5257867 5275878 5291464 5303906 5320201 5337450 5350227 5364416 5382179 5396346 5409022 5425850 5441757 5454050 5469041 5486094 5499159 5512364 5529305 5543875 5556101 5571743 5587769 5600115 5613993 5630610 5644019 5656513 5672515 5687377 5699392 5713927 5729853 5742386 5755369 5771363 5785101 5797130 5812157 5827156 5839147 5852707 5868329 5881117 5893426 5908673 5922666 5934441 5948529 5963491 5975586 5988337 6003486 6016535 6028378 6042825 6056956 6068646 6081884 6096637 6108909 6121029 6135587 6148850 6160410 6174063 6188195 6199921 6212433 6226829 6239299 6250962 6264863 6278256 6289684 6302599 6316590 6328428 6340354 6354282 6366927 6378289 6391521 6404936 6416346 6428612 6442334 6454316 6465800 6479190 6491950 6503146 6515738 6529060 6540528

###Model the other variables for the other stocks ####S01 Var01, var02

TsVar01 <- ts(group\_S01\_df$Var01, frequency = 5\*52)  
arimaFitVar01 <- auto.arima(TsVar01, seasonal = FALSE)  
arimaFitVar01

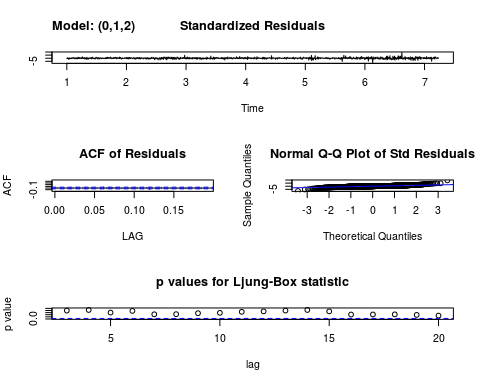
## Series: TsVar01   
## ARIMA(0,1,2) with drift   
##   
## Coefficients:  
## ma1 ma2 drift  
## 0.0875 -0.0727 0.0220  
## s.e. 0.0248 0.0250 0.0129  
##   
## sigma^2 estimated as 0.2612: log likelihood=-1210.51  
## AIC=2429.02 AICc=2429.04 BIC=2450.58

TsVar02 <- ts(group\_S01\_df$Var02, frequency = 5\*52)  
arimaFitVar02 <- auto.arima(TsVar02, seasonal = FALSE)  
arimaFitVar02

## Series: TsVar02   
## ARIMA(1,1,3) with drift   
##   
## Coefficients:  
## ar1 ma1 ma2 ma3 drift  
## 0.872 -1.4349 0.3060 0.1340 -5738.623  
## s.e. 0.042 0.0511 0.0447 0.0346 2703.689  
##   
## sigma^2 estimated as 6.799e+12: log likelihood=-26247.04  
## AIC=52506.07 AICc=52506.12 BIC=52538.42

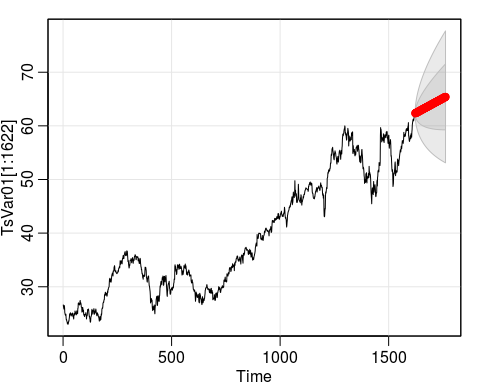
sarima(TsVar01, 0, 1, 2)

## initial value -0.665863   
## iter 2 value -0.672136  
## iter 3 value -0.672176  
## iter 4 value -0.672178  
## iter 4 value -0.672178  
## iter 4 value -0.672178  
## final value -0.672178   
## converged  
## initial value -0.672173   
## iter 1 value -0.672173  
## final value -0.672173   
## converged



## $fit  
##   
## Call:  
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,   
## Q), period = S), xreg = constant, optim.control = list(trace = trc, REPORT = 1,   
## reltol = tol))  
##   
## Coefficients:  
## ma1 ma2 constant  
## 0.0875 -0.0727 0.0220  
## s.e. 0.0248 0.0250 0.0129  
##   
## sigma^2 estimated as 0.2607: log likelihood = -1210.51, aic = 2429.02  
##   
## $degrees\_of\_freedom  
## [1] 1618  
##   
## $ttable  
## Estimate SE t.value p.value  
## ma1 0.0875 0.0248 3.5312 0.0004  
## ma2 -0.0727 0.0250 -2.9083 0.0037  
## constant 0.0220 0.0129 1.7114 0.0872  
##   
## $AIC  
## [1] -0.3406582  
##   
## $AICc  
## [1] -0.3394099  
##   
## $BIC  
## [1] -1.330686

fSarima <- sarima.for(TsVar01[1:1622], 140, 1, 1, 3, plot.all = TRUE)



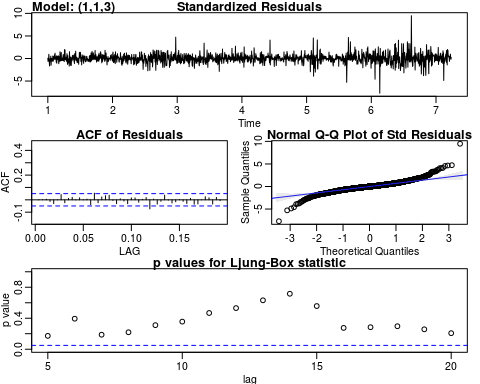
fSarima$pred[1:20]

## [1] 62.34869 62.36247 62.38465 62.40670 62.42873 62.45076 62.47278 62.49480 62.51682 62.53884 62.56087 62.58289 62.60491 62.62693 62.64895 62.67097 62.69300 62.71502 62.73704 62.75906

#save prediction into df  
group\_S01\_df[1623:1762,2] <- fSarima$pred  
  
sarima(TsVar01, 1, 1, 3)

## initial value -0.665684   
## iter 2 value -0.669462  
## iter 3 value -0.672009  
## iter 4 value -0.672020  
## iter 5 value -0.672020  
## iter 6 value -0.672020  
## iter 7 value -0.672020  
## iter 8 value -0.672020  
## iter 9 value -0.672020  
## iter 10 value -0.672020  
## iter 11 value -0.672020  
## iter 12 value -0.672021  
## iter 13 value -0.672021  
## iter 14 value -0.672021  
## iter 15 value -0.672021  
## iter 16 value -0.672021  
## iter 17 value -0.672021  
## iter 18 value -0.672021  
## iter 19 value -0.672021  
## iter 20 value -0.672022  
## iter 21 value -0.672023  
## iter 22 value -0.672025  
## iter 23 value -0.672032  
## iter 24 value -0.672038  
## iter 25 value -0.672038  
## iter 26 value -0.672039  
## iter 27 value -0.672040  
## iter 28 value -0.672041  
## iter 29 value -0.672041  
## iter 29 value -0.672041  
## iter 29 value -0.672041  
## final value -0.672041   
## converged  
## initial value -0.672174   
## iter 2 value -0.672174  
## iter 3 value -0.672174  
## iter 4 value -0.672174  
## iter 5 value -0.672174  
## iter 6 value -0.672174  
## iter 6 value -0.672174  
## iter 6 value -0.672174  
## final value -0.672174   
## converged

## Warning in sqrt(diag(fitit$var.coef)): NaNs produced  
  
## Warning in sqrt(diag(fitit$var.coef)): NaNs produced

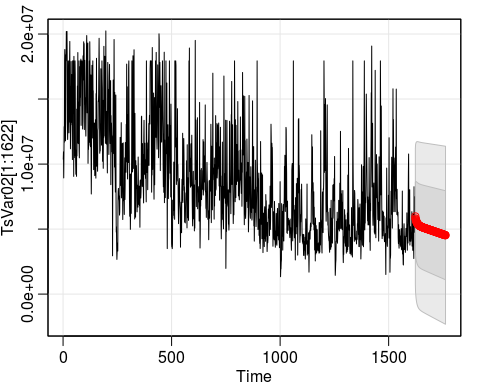


## $fit  
##   
## Call:  
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,   
## Q), period = S), xreg = constant, optim.control = list(trace = trc, REPORT = 1,   
## reltol = tol))  
##   
## Coefficients:

## Warning in sqrt(diag(x$var.coef)): NaNs produced

## ar1 ma1 ma2 ma3 constant  
## 0.2257 -0.1383 -0.0922 0.018 0.0220  
## s.e. NaN NaN NaN NaN 0.0129  
##   
## sigma^2 estimated as 0.2607: log likelihood = -1210.51, aic = 2433.01  
##   
## $degrees\_of\_freedom  
## [1] 1616  
##   
## $ttable  
## Estimate SE t.value p.value  
## ar1 0.2257 NaN NaN NaN  
## ma1 -0.1383 NaN NaN NaN  
## ma2 -0.0922 NaN NaN NaN  
## ma3 0.0180 NaN NaN NaN  
## constant 0.0220 0.0129 1.7074 0.0879  
##   
## $AIC  
## [1] -0.3381947  
##   
## $AICc  
## [1] -0.3369296  
##   
## $BIC  
## [1] -1.321575

fSarima <- sarima.for(TsVar02[1:1622], 140, 1, 1, 3, plot.all = TRUE)



fSarima$pred[1:20]

## [1] 5976242 5815753 5746395 5685180 5631065 5583142 5540618 5502802 5469091 5438961 5411953 5387667 5365755 5345913 5327876 5311413 5296323 5282429 5269580 5257640

group\_S01\_df[1623:1762,3] <- fSarima$pred

####S02 Var02, var03

TsVar02 <- ts(group\_S02\_df$Var02, frequency = 5\*52)  
arimaFitVar02 <- auto.arima(TsVar02, seasonal = FALSE)  
arimaFitVar02

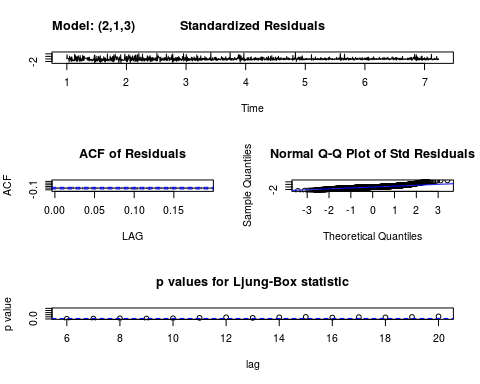
## Series: TsVar02   
## ARIMA(3,1,4) with drift   
##   
## Coefficients:  
## ar1 ar2 ar3 ma1 ma2 ma3 ma4 drift  
## 0.0818 0.8584 -0.1525 -0.5703 -1.0397 0.5069 0.1112 -41486.54  
## s.e. 0.2240 0.0611 0.1669 0.2245 0.0889 0.2168 0.0786 16530.25  
##   
## sigma^2 estimated as 2.745e+14: log likelihood=-29243.06  
## AIC=58504.12 AICc=58504.23 BIC=58552.63

TsVar03 <- ts(group\_S02\_df$Var03, frequency = 5\*52)  
arimaFitVar03 <- auto.arima(TsVar03, seasonal = FALSE)  
arimaFitVar03

## Series: TsVar03   
## ARIMA(2,1,1)   
##   
## Coefficients:  
## ar1 ar2 ma1  
## 0.7197 0.0683 -0.8283  
## s.e. 0.2518 0.0450 0.2499  
##   
## sigma^2 estimated as 0.09362: log likelihood=-378.95  
## AIC=765.9 AICc=765.92 BIC=787.46

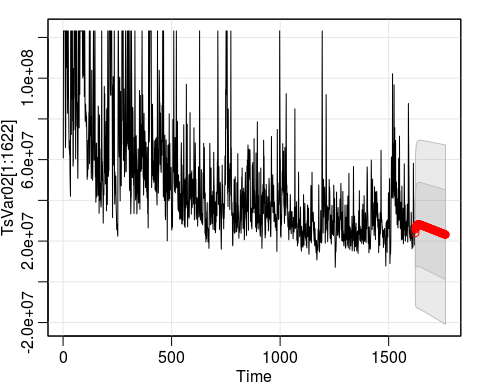
sarima(TsVar02, 2, 1, 3)

## initial value 16.749075   
## iter 2 value 16.663983  
## iter 3 value 16.638407  
## iter 4 value 16.637359  
## iter 5 value 16.635204  
## iter 6 value 16.634410  
## iter 7 value 16.631265  
## iter 8 value 16.630473  
## iter 9 value 16.628598  
## iter 10 value 16.628500  
## iter 11 value 16.628428  
## iter 12 value 16.628408  
## iter 13 value 16.628388  
## iter 14 value 16.628383  
## iter 15 value 16.628379  
## iter 16 value 16.628335  
## iter 17 value 16.628162  
## iter 18 value 16.627746  
## iter 19 value 16.626725  
## iter 20 value 16.626340  
## iter 21 value 16.625694  
## iter 22 value 16.623888  
## iter 23 value 16.622713  
## iter 24 value 16.622430  
## iter 25 value 16.622120  
## iter 26 value 16.622020  
## iter 27 value 16.621729  
## iter 28 value 16.621454  
## iter 29 value 16.621206  
## iter 30 value 16.619859  
## iter 31 value 16.619184  
## iter 32 value 16.619019  
## iter 33 value 16.618840  
## iter 34 value 16.618818  
## iter 35 value 16.618799  
## iter 35 value 16.618799  
## final value 16.618799   
## converged  
## initial value 16.623201   
## iter 2 value 16.623171  
## iter 3 value 16.623167  
## iter 4 value 16.623148  
## iter 5 value 16.623139  
## iter 6 value 16.623123  
## iter 7 value 16.623082  
## iter 8 value 16.623078  
## iter 9 value 16.623077  
## iter 10 value 16.623074  
## iter 11 value 16.623064  
## iter 12 value 16.623040  
## iter 13 value 16.622967  
## iter 14 value 16.622896  
## iter 15 value 16.622891  
## iter 16 value 16.622848  
## iter 17 value 16.622838  
## iter 18 value 16.622739  
## iter 19 value 16.622737  
## iter 20 value 16.622736  
## iter 21 value 16.622732  
## iter 22 value 16.622728  
## iter 23 value 16.622726  
## iter 23 value 16.622726  
## iter 23 value 16.622726  
## final value 16.622726   
## converged



## $fit  
##   
## Call:  
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,   
## Q), period = S), xreg = constant, optim.control = list(trace = trc, REPORT = 1,   
## reltol = tol))  
##   
## Coefficients:  
## ar1 ar2 ma1 ma2 ma3 constant  
## 1.1736 -0.2579 -1.6656 0.6209 0.0480 -41388.49  
## s.e. 0.1288 0.1060 0.1293 0.1867 0.0639 16969.65  
##   
## sigma^2 estimated as 2.74e+14: log likelihood = -29245.54, aic = 58505.08  
##   
## $degrees\_of\_freedom  
## [1] 1615  
##   
## $ttable  
## Estimate SE t.value p.value  
## ar1 1.1736 0.1288 9.1104 0.0000  
## ar2 -0.2579 0.1060 -2.4324 0.0151  
## ma1 -1.6656 0.1293 -12.8797 0.0000  
## ma2 0.6209 0.1867 3.3249 0.0009  
## ma3 0.0480 0.0639 0.7520 0.4522  
## constant -41388.4936 16969.6541 -2.4390 0.0148  
##   
## $AIC  
## [1] 34.25167  
##   
## $AICc  
## [1] 34.25294  
##   
## $BIC  
## [1] 33.27161

fSarima <- sarima.for(TsVar02[1:1622], 140, 1, 1, 3, plot.all = TRUE)

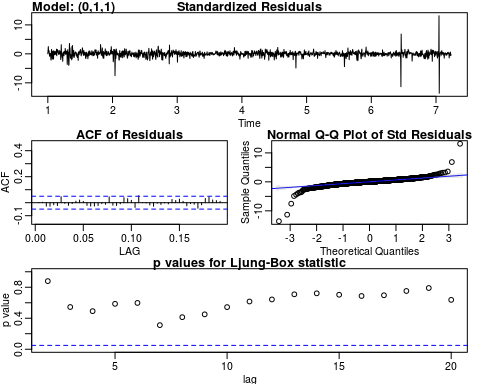


fSarima$pred[1:20]

## [1] 23971947 25768568 26307225 26740411 27087287 27363533 27581982 27753133 27885580 27986355 28061212 28114859 28151150 28173238 28183703 28184657 28177829 28164631 28146221 28123546

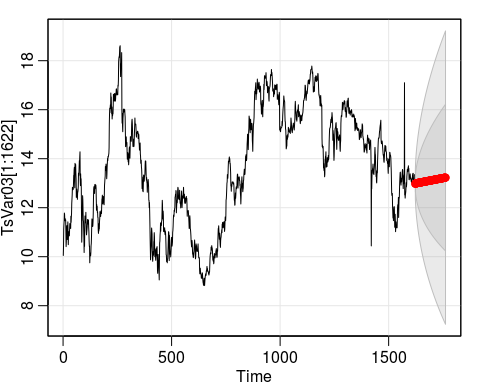
#save prediction into df  
group\_S02\_df[1623:1762,2] <- fSarima$pred  
  
sarima(TsVar03, 0, 1, 1)

## initial value -1.179017   
## iter 2 value -1.184636  
## iter 3 value -1.184636  
## iter 3 value -1.184636  
## iter 3 value -1.184636  
## final value -1.184636   
## converged  
## initial value -1.184639   
## iter 1 value -1.184639  
## final value -1.184639   
## converged



## $fit  
##   
## Call:  
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,   
## Q), period = S), xreg = constant, optim.control = list(trace = trc, REPORT = 1,   
## reltol = tol))  
##   
## Coefficients:  
## ma1 constant  
## -0.1064 0.0018  
## s.e. 0.0248 0.0068  
##   
## sigma^2 estimated as 0.09355: log likelihood = -379.8, aic = 765.6  
##   
## $degrees\_of\_freedom  
## [1] 1619  
##   
## $ttable  
## Estimate SE t.value p.value  
## ma1 -0.1064 0.0248 -4.2896 0.0000  
## constant 0.0018 0.0068 0.2636 0.7921  
##   
## $AIC  
## [1] -1.36682  
##   
## $AICc  
## [1] -1.365578  
##   
## $BIC  
## [1] -2.360172

fSarima <- sarima.for(TsVar03[1:1622], 140, 1, 1, 3, plot.all = TRUE)



fSarima$pred[1:20]

## [1] 12.97327 12.97900 12.98609 12.99083 12.99424 12.99691 12.99916 13.00117 13.00305 13.00486 13.00663 13.00837 13.01010 13.01182 13.01354 13.01525 13.01697 13.01868 13.02039 13.02211

group\_S02\_df[1623:1762,3] <- fSarima$pred

####S03 Var05, var07

TsVar05 <- ts(group\_S03\_df$Var05, frequency = 5\*52)  
arimaFitVar05 <- auto.arima(TsVar05, seasonal = FALSE)  
arimaFitVar05

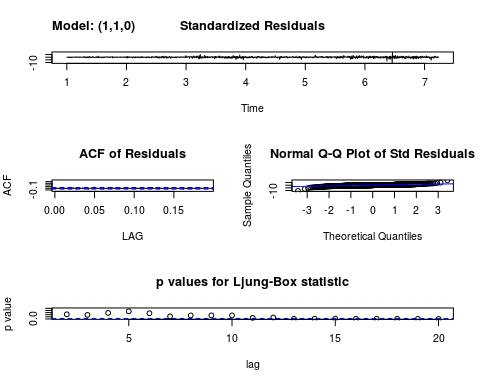
## Series: TsVar05   
## ARIMA(1,1,0)   
##   
## Coefficients:  
## ar1  
## -0.1634  
## s.e. 0.0245  
##   
## sigma^2 estimated as 2.248: log likelihood=-2956.16  
## AIC=5916.32 AICc=5916.33 BIC=5927.1

TsVar07 <- ts(group\_S03\_df$Var07, frequency = 5\*52)  
arimaFitVar07 <- auto.arima(TsVar07, seasonal = FALSE)  
arimaFitVar07

## Series: TsVar07   
## ARIMA(0,1,0)   
##   
## sigma^2 estimated as 1.811: log likelihood=-2781.33  
## AIC=5564.65 AICc=5564.66 BIC=5570.04

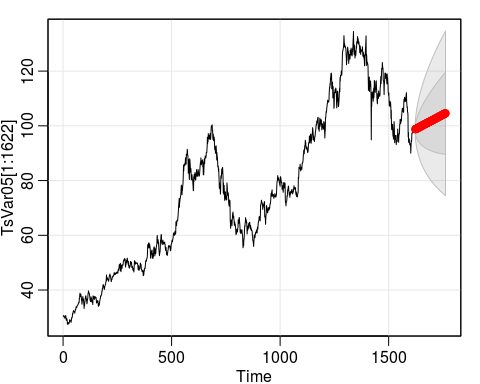
sarima(TsVar05, 1, 1, 0)

## initial value 0.418193   
## iter 2 value 0.404491  
## iter 2 value 0.404491  
## iter 2 value 0.404491  
## final value 0.404491   
## converged  
## initial value 0.404193   
## iter 1 value 0.404193  
## final value 0.404193   
## converged



## $fit  
##   
## Call:  
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,   
## Q), period = S), xreg = constant, optim.control = list(trace = trc, REPORT = 1,   
## reltol = tol))  
##   
## Coefficients:  
## ar1 constant  
## -0.1643 0.0421  
## s.e. 0.0245 0.0320  
##   
## sigma^2 estimated as 2.244: log likelihood = -2955.3, aic = 5916.59  
##   
## $degrees\_of\_freedom  
## [1] 1619  
##   
## $ttable  
## Estimate SE t.value p.value  
## ar1 -0.1643 0.0245 -6.7081 0.0000  
## constant 0.0421 0.0320 1.3159 0.1884  
##   
## $AIC  
## [1] 1.810835  
##   
## $AICc  
## [1] 1.812077  
##   
## $BIC  
## [1] 0.8174829

fSarima <- sarima.for(TsVar05[1:1622], 140, 1, 1, 3, plot.all = TRUE)

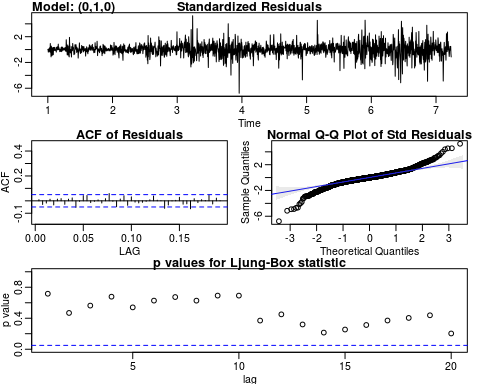


fSarima$pred[1:20]

## [1] 98.72952 98.78150 98.81899 98.86075 98.90278 98.94483 98.98688 99.02893 99.07098 99.11303 99.15508 99.19713 99.23918 99.28123 99.32328 99.36533 99.40738 99.44943 99.49148 99.53353

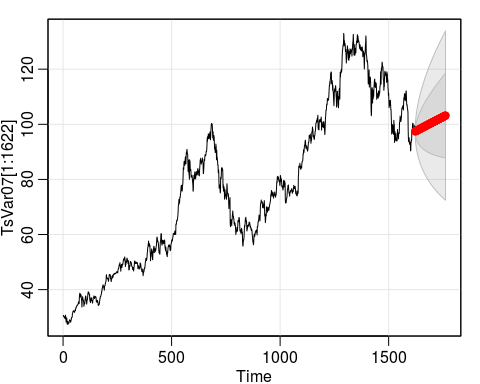
#save prediction into df  
group\_S03\_df[1623:1762,2] <- fSarima$pred  
  
sarima(TsVar07, 0, 1, 0)

## initial value 0.296402   
## iter 1 value 0.296402  
## final value 0.296402   
## converged  
## initial value 0.296402   
## iter 1 value 0.296402  
## final value 0.296402   
## converged



## $fit  
##   
## Call:  
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,   
## Q), period = S), xreg = constant, optim.control = list(trace = trc, REPORT = 1,   
## reltol = tol))  
##   
## Coefficients:  
## constant  
## 0.0412  
## s.e. 0.0334  
##   
## sigma^2 estimated as 1.809: log likelihood = -2780.57, aic = 5565.13  
##   
## $degrees\_of\_freedom  
## [1] 1620  
##   
## $ttable  
## Estimate SE t.value p.value  
## constant 0.0412 0.0334 1.2329 0.2178  
##   
## $AIC  
## [1] 1.594037  
##   
## $AICc  
## [1] 1.595274  
##   
## $BIC  
## [1] 0.5973606

fSarima <- sarima.for(TsVar07[1:1622], 140, 1, 1, 3, plot.all = TRUE)



fSarima$pred[1:20]

## [1] 97.37321 97.49242 97.54952 97.58059 97.62806 97.66520 97.70885 97.74840 97.79053 97.83103 97.87256 97.91344 97.95473 97.99576 98.03696 98.07805 98.11921 98.16033 98.20147 98.24259

group\_S03\_df[1623:1762,3] <- fSarima$pred

####S04 Var01, var02

TsVar01 <- ts(group\_S04\_df$Var01, frequency = 5\*52)  
arimaFitVar01 <- auto.arima(TsVar01, seasonal = FALSE)  
arimaFitVar01

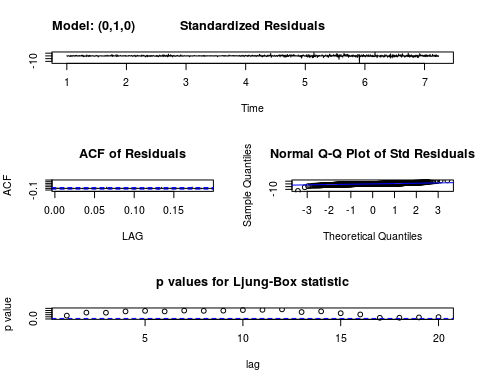
## Series: TsVar01   
## ARIMA(0,1,0)   
##   
## sigma^2 estimated as 0.2557: log likelihood=-1194.84  
## AIC=2391.68 AICc=2391.68 BIC=2397.07

TsVar02 <- ts(group\_S04\_df$Var02, frequency = 5\*52)  
arimaFitVar02 <- auto.arima(TsVar02, seasonal = FALSE)  
arimaFitVar02

## Series: TsVar02   
## ARIMA(2,1,1)   
##   
## Coefficients:  
## ar1 ar2 ma1  
## 0.4778 0.0138 -0.9523  
## s.e. 0.0282 0.0274 0.0134  
##   
## sigma^2 estimated as 5.723e+13: log likelihood=-27974.37  
## AIC=55956.73 AICc=55956.76 BIC=55978.3

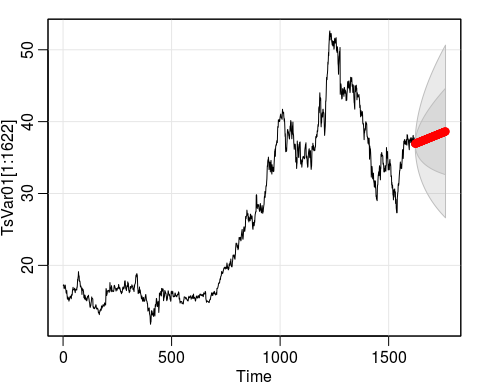
sarima(TsVar01, 0, 1, 0)

## initial value -0.682128   
## iter 1 value -0.682128  
## final value -0.682128   
## converged  
## initial value -0.682128   
## iter 1 value -0.682128  
## final value -0.682128   
## converged



## $fit  
##   
## Call:  
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,   
## Q), period = S), xreg = constant, optim.control = list(trace = trc, REPORT = 1,   
## reltol = tol))  
##   
## Coefficients:  
## constant  
## 0.0122  
## s.e. 0.0126  
##   
## sigma^2 estimated as 0.2556: log likelihood = -1194.37, aic = 2392.74  
##   
## $degrees\_of\_freedom  
## [1] 1620  
##   
## $ttable  
## Estimate SE t.value p.value  
## constant 0.0122 0.0126 0.9684 0.333  
##   
## $AIC  
## [1] -0.3630232  
##   
## $AICc  
## [1] -0.3617856  
##   
## $BIC  
## [1] -1.359699

fSarima <- sarima.for(TsVar01[1:1622], 140, 1, 1, 3, plot.all = TRUE)

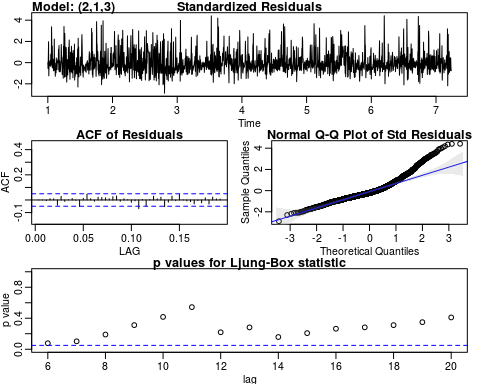


fSarima$pred[1:20]

## [1] 36.91074 36.94176 36.95312 36.96565 36.97778 36.99004 37.00226 37.01449 37.02672 37.03895 37.05118 37.06341 37.07564 37.08787 37.10010 37.11233 37.12456 37.13679 37.14902 37.16125

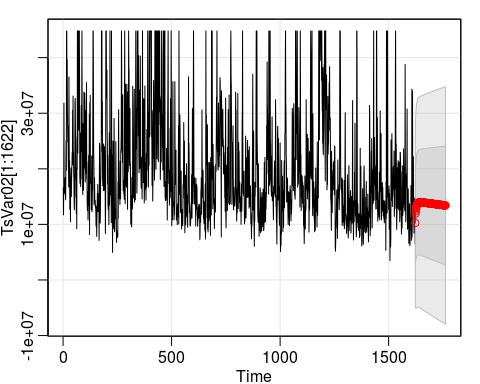
#save prediction into df  
group\_S04\_df[1623:1762,2] <- fSarima$pred  
  
sarima(TsVar02, 2, 1, 3)

## initial value 15.964494   
## iter 2 value 15.874582  
## iter 3 value 15.851302  
## iter 4 value 15.849547  
## iter 5 value 15.848107  
## iter 6 value 15.847620  
## iter 7 value 15.846259  
## iter 8 value 15.841882  
## iter 9 value 15.840427  
## iter 10 value 15.839248  
## iter 11 value 15.838042  
## iter 12 value 15.837468  
## iter 13 value 15.837033  
## iter 14 value 15.837021  
## iter 15 value 15.837018  
## iter 16 value 15.836989  
## iter 17 value 15.836972  
## iter 18 value 15.836669  
## iter 19 value 15.835242  
## iter 20 value 15.835099  
## iter 21 value 15.835049  
## iter 22 value 15.834986  
## iter 23 value 15.834818  
## iter 24 value 15.834743  
## iter 25 value 15.834679  
## iter 26 value 15.834632  
## iter 27 value 15.834597  
## iter 28 value 15.834595  
## iter 29 value 15.834593  
## iter 30 value 15.834584  
## iter 31 value 15.834582  
## iter 32 value 15.834580  
## iter 32 value 15.834580  
## iter 32 value 15.834580  
## final value 15.834580   
## converged  
## initial value 15.834747   
## iter 2 value 15.834742  
## iter 3 value 15.834739  
## iter 4 value 15.834735  
## iter 5 value 15.834733  
## iter 6 value 15.834724  
## iter 7 value 15.834711  
## iter 8 value 15.834699  
## iter 9 value 15.834693  
## iter 10 value 15.834688  
## iter 11 value 15.834675  
## iter 12 value 15.834646  
## iter 13 value 15.834586  
## iter 14 value 15.834502  
## iter 15 value 15.834485  
## iter 16 value 15.834477  
## iter 17 value 15.834475  
## iter 18 value 15.834474  
## iter 18 value 15.834474  
## iter 18 value 15.834474  
## final value 15.834474   
## converged



## $fit  
##   
## Call:  
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,   
## Q), period = S), xreg = constant, optim.control = list(trace = trc, REPORT = 1,   
## reltol = tol))  
##   
## Coefficients:  
## ar1 ar2 ma1 ma2 ma3 constant  
## 0.7852 0.0110 -1.2652 0.1172 0.1602 -5087.528  
## s.e. 0.2534 0.1594 0.2521 0.2810 0.0573 11502.635  
##   
## sigma^2 estimated as 5.665e+13: log likelihood = -27967.78, aic = 55949.56  
##   
## $degrees\_of\_freedom  
## [1] 1615  
##   
## $ttable  
## Estimate SE t.value p.value  
## ar1 0.7852 0.2534 3.0991 0.0020  
## ar2 0.0110 0.1594 0.0690 0.9450  
## ma1 -1.2652 0.2521 -5.0176 0.0000  
## ma2 0.1172 0.2810 0.4172 0.6766  
## ma3 0.1602 0.0573 2.7960 0.0052  
## constant -5087.5277 11502.6351 -0.4423 0.6583  
##   
## $AIC  
## [1] 32.67534  
##   
## $AICc  
## [1] 32.67662  
##   
## $BIC  
## [1] 31.69528

fSarima <- sarima.for(TsVar02[1:1622], 140, 1, 1, 3, plot.all = TRUE)



fSarima$pred[1:20]

## [1] 10252784 11922045 12350210 12692441 12965784 13183901 13357748 13496104 13606009 13693106 13761920 13816076 13858482 13891468 13916903 13936285 13950813 13961452 13968972 13973992

group\_S04\_df[1623:1762,3] <- fSarima$pred

####S06 Var05, var07

TsVar05 <- ts(group\_S06\_df$Var05, frequency = 5\*52)  
arimaFitVar05 <- auto.arima(TsVar05, seasonal = FALSE)  
arimaFitVar05

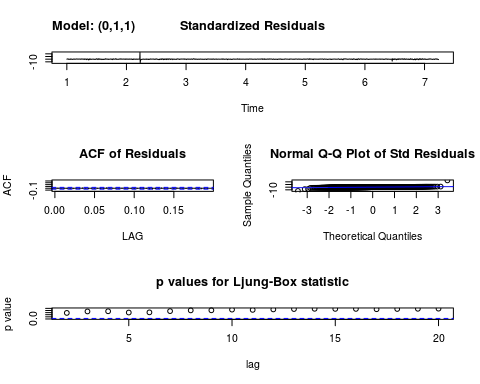
## Series: TsVar05   
## ARIMA(0,1,1)   
##   
## Coefficients:  
## ma1  
## -0.4661  
## s.e. 0.0230  
##   
## sigma^2 estimated as 0.844: log likelihood=-2162.24  
## AIC=4328.48 AICc=4328.48 BIC=4339.26

TsVar07 <- ts(group\_S06\_df$Var07, frequency = 5\*52)  
arimaFitVar07 <- auto.arima(TsVar07, seasonal = FALSE)  
arimaFitVar07

## Series: TsVar07   
## ARIMA(0,1,1)   
##   
## Coefficients:  
## ma1  
## -0.4553  
## s.e. 0.0232  
##   
## sigma^2 estimated as 0.8512: log likelihood=-2169.12  
## AIC=4342.25 AICc=4342.25 BIC=4353.03

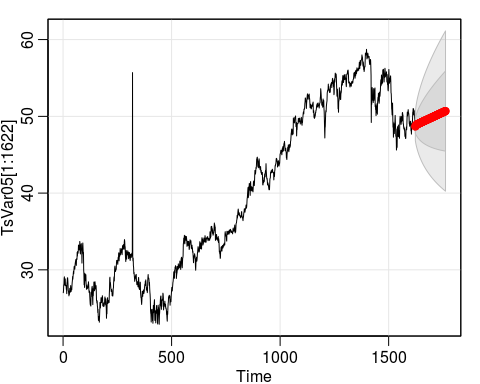
sarima(TsVar05, 0, 1, 1)

## initial value 0.008912   
## iter 2 value -0.080612  
## iter 3 value -0.084606  
## iter 4 value -0.085420  
## iter 5 value -0.085437  
## iter 6 value -0.085437  
## iter 7 value -0.085437  
## iter 8 value -0.085437  
## iter 8 value -0.085437  
## iter 8 value -0.085437  
## final value -0.085437   
## converged  
## initial value -0.085407   
## iter 2 value -0.085407  
## iter 3 value -0.085407  
## iter 3 value -0.085407  
## iter 3 value -0.085407  
## final value -0.085407   
## converged



## $fit  
##   
## Call:  
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,   
## Q), period = S), xreg = constant, optim.control = list(trace = trc, REPORT = 1,   
## reltol = tol))  
##   
## Coefficients:  
## ma1 constant  
## -0.4673 0.0131  
## s.e. 0.0230 0.0122  
##   
## sigma^2 estimated as 0.8429: log likelihood = -2161.65, aic = 4329.31  
##   
## $degrees\_of\_freedom  
## [1] 1619  
##   
## $ttable  
## Estimate SE t.value p.value  
## ma1 -0.4673 0.0230 -20.3260 0.0000  
## constant 0.0131 0.0122 1.0813 0.2797  
##   
## $AIC  
## [1] 0.8315002  
##   
## $AICc  
## [1] 0.8327424  
##   
## $BIC  
## [1] -0.1618519

fSarima <- sarima.for(TsVar05[1:1622], 140, 1, 1, 3, plot.all = TRUE)

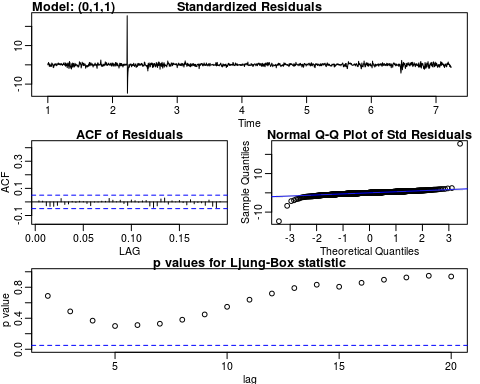


fSarima$pred[1:20]

## [1] 48.66678 48.72965 48.78670 48.82977 48.86331 48.89037 48.91300 48.93263 48.95020 48.96638 48.98161 48.99619 49.01033 49.02417 49.03780 49.05129 49.06469 49.07802 49.09131 49.10457

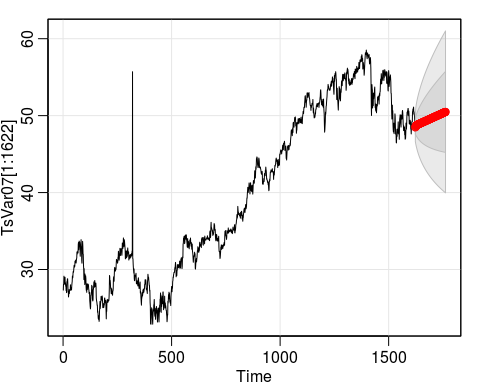
#save prediction into df  
group\_S06\_df[1623:1762,2] <- fSarima$pred  
  
sarima(TsVar07, 0, 1, 1)

## initial value 0.010266   
## iter 2 value -0.077502  
## iter 3 value -0.080668  
## iter 4 value -0.081108  
## iter 5 value -0.081114  
## iter 6 value -0.081114  
## iter 6 value -0.081114  
## final value -0.081114   
## converged  
## initial value -0.081122   
## iter 2 value -0.081122  
## iter 3 value -0.081122  
## iter 4 value -0.081122  
## iter 4 value -0.081122  
## iter 4 value -0.081122  
## final value -0.081122   
## converged



## $fit  
##   
## Call:  
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,   
## Q), period = S), xreg = constant, optim.control = list(trace = trc, REPORT = 1,   
## reltol = tol))  
##   
## Coefficients:  
## ma1 constant  
## -0.4563 0.0128  
## s.e. 0.0232 0.0125  
##   
## sigma^2 estimated as 0.8501: log likelihood = -2168.6, aic = 4343.2  
##   
## $degrees\_of\_freedom  
## [1] 1619  
##   
## $ttable  
## Estimate SE t.value p.value  
## ma1 -0.4563 0.0232 -19.6565 0.000  
## constant 0.0128 0.0125 1.0239 0.306  
##   
## $AIC  
## [1] 0.8400777  
##   
## $AICc  
## [1] 0.8413199  
##   
## $BIC  
## [1] -0.1532744

fSarima <- sarima.for(TsVar07[1:1622], 140, 1, 1, 3, plot.all = TRUE)



fSarima$pred[1:20]

## [1] 48.46941 48.54925 48.63374 48.68648 48.72155 48.74678 48.76652 48.78322 48.79822 48.81227 48.82580 48.83903 48.85210 48.86508 48.87801 48.89091 48.90380 48.91667 48.92954 48.94241

group\_S06\_df[1623:1762,3] <- fSarima$pred

#### Group S06 Forecast

#put the sequence numbers back in  
seqNumbers <- distinct(as.data.frame(project\_in\_df[,1]))  
group\_S01\_df[,1] = seqNumbers  
group\_S02\_df[,1] = seqNumbers  
group\_S03\_df[,1] = seqNumbers  
group\_S04\_df[,1] = seqNumbers  
group\_S05\_df[,1] = seqNumbers  
group\_S06\_df[,1] = seqNumbers

#### Export Results

write.xlsx(project\_in\_df, file = "output.xls", sheetName = "Set for Class", row.names=FALSE, append = FALSE)  
write.xlsx(group\_S01\_df, file = "output.xls", sheetName = "S01", row.names=FALSE, append = TRUE)  
write.xlsx(group\_S02\_df, file = "output.xls", sheetName = "S02", row.names=FALSE, append = TRUE)  
write.xlsx(group\_S03\_df, file = "output.xls", sheetName = "S03", row.names=FALSE, append = TRUE)  
write.xlsx(group\_S04\_df, file = "output.xls", sheetName = "S04", row.names=FALSE, append = TRUE)  
write.xlsx(group\_S05\_df, file = "output.xls", sheetName = "S05", row.names=FALSE, append = TRUE)  
write.xlsx(group\_S06\_df, file = "output.xls", sheetName = "S06", row.names=FALSE, append = TRUE)