Data 624 Project 1
CUNY SPS MSDS
Dr. Scott Burk
Summary 2022
Group 4:

Melvin Matanos	Claire Meyer	Chinedu Onyeka
Euclid Zhang	Jie Zou	

6/11/2022

Table of Contents

Executive Summary

In this project, we explored and pre-processed the data from a de-identified Excel spreadsheet, then generated forecasts for future time periods.

Upon exploration, the found that there are a few extreme outliers and missing values in the data. The extreme outliers and missing values were replaced/filled by reasonable values. For variables with strong correlations and sufficient data, linear regression models were used to produce the reasonable values. For the remaining missing values, linear interpolation was used. The linear interpolation method estimates the missing values so that the filled values and the adjacent available values in the time series are connected by a straight line. We also found that variable 02 is highly right-skewed so a log-transformation was applied to it when creating our time series models.

Two types of models were built for each time series, the ETS and ARIMA models. To identify the optimal model to develop these forecasts, we evaluated both ETS and ARIMA models by the MAPE (mean absolute percentage error) scores, and leveraged cross-validation to compare performance. Additionally, we performed goodness of fit checks by examining the residuals of the models. Ultimately, the ARIMA models performed best, and we finalized our forecast values from there.

Data Exploration

Out data is the observations for 6 individual categories (S01, S02, S03, S04, S05, S06) during a period of length 1622. Each observation contains 5 variables (Var01, Var02, Var03, Var05, Var07). So there are 6*5=30 time series (one per variable per category). In this analysis, we will focus on the following 12 time series and perform forecast the values for the next 140 periods:

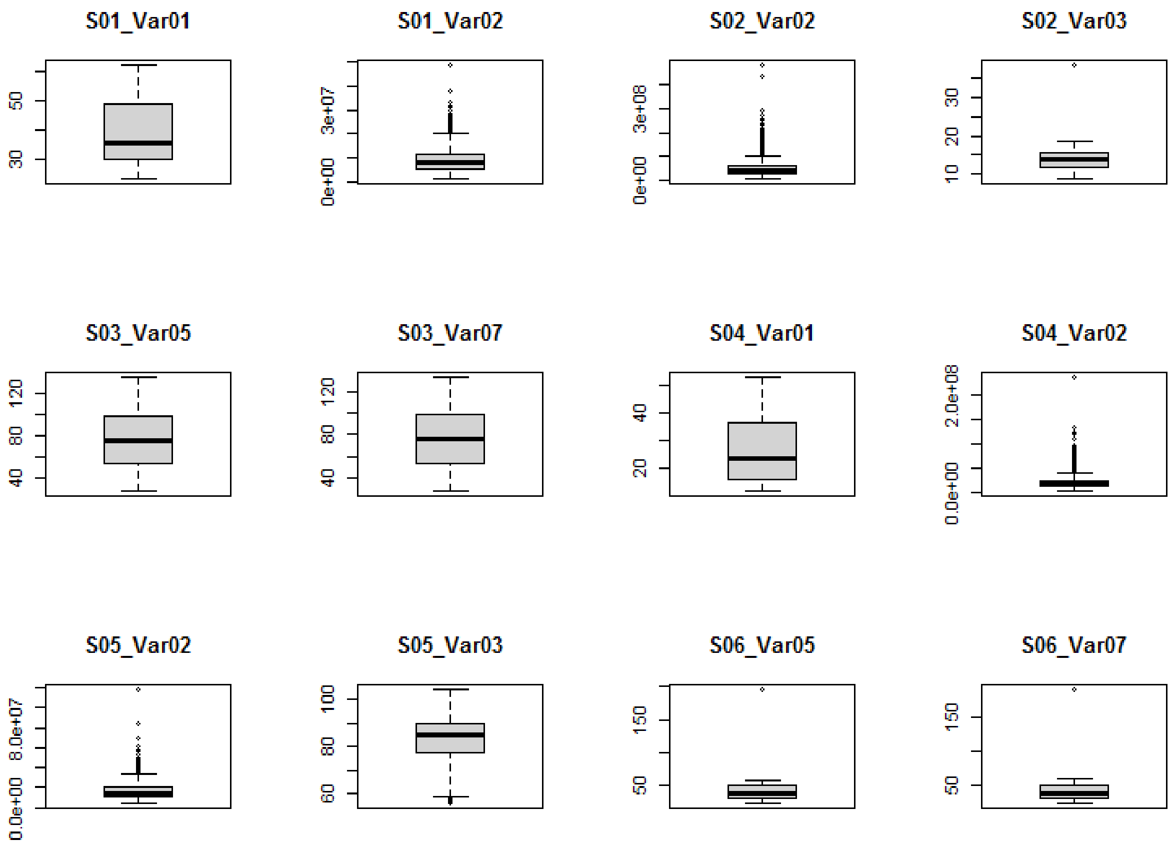
- Group S01 – Var01, Var02
- Group S02 – Var02, Var03
- Group S03 – Var05, Var07
- Group S04 – Var01, Var02
- Group S05 – Var02, Var03
- Group S06 – Var05, Var07

First, let look at the summary of the data:

```
##    SeriesInd     group         Var01            Var02
##  Min.   :40669   S01:1622   Min.   :  9.03   Min.   :  1339900
##  1st Qu.:41253   S02:1622   1st Qu.: 23.10   1st Qu.: 12520675
##  Median :41846   S03:1622   Median : 38.44   Median : 21086550
##  Mean   :41843   S04:1622   Mean   : 46.98   Mean   : 37035741
##  3rd Qu.:42430   S05:1622   3rd Qu.: 66.78   3rd Qu.: 42486700
##  Max.   :43021   S06:1622   Max.   :195.18   Max.   :480879500
##                             NA's   :14       NA's   :2
##      Var03            Var05            Var07
##  Min.   :  8.82   Min.   :  8.99   Min.   :  8.92
##  1st Qu.: 22.59   1st Qu.: 22.91   1st Qu.: 22.88
##  Median : 37.66   Median : 38.05   Median : 38.05
##  Mean   : 46.12   Mean   : 46.55   Mean   : 46.56
##  3rd Qu.: 65.88   3rd Qu.: 66.38   3rd Qu.: 66.31
##  Max.   :189.36   Max.   :195.00   Max.   :189.72
##  NA's   :26       NA's   :26       NA's   :26
```

We have some **missing values in each variables**, these missing values will be imputed / filled with reasonable values before building our models. The variables also have maximum values much larger than the 3rd quartile (the value that is larger than 75% of all values in the same variable), there may be **extreme outliers** in the data.

We can check the outliers and also the skewness of the variables form the boxplots.
The following are the boxplots of the 12 time series.



Var02 of all categories are highly right skewed that may need to be transformed to stabilize the variance.
S02_Var03, S06_Var05, and S06_Var07 have an extreme outlier.
If the extreme outliers are excluded, all Variables except Var02 have stable variance and no transformation is needed.
We will remove the extreme outliers and impute them with reasonable values along with other missing values.

Data Preparation

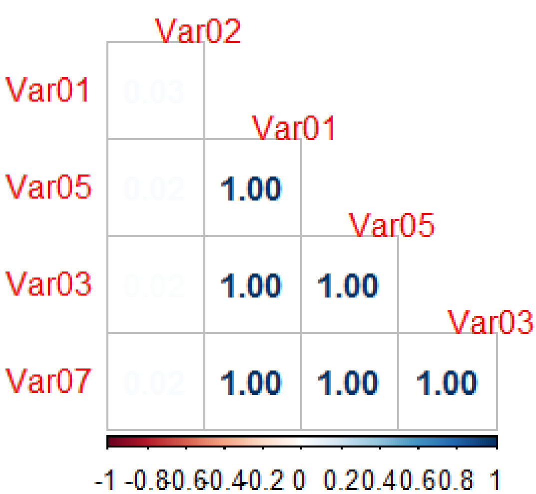
Missing Values Imputation

The following are the records with missing value for our original data.

```
##      SeriesInd group Var01    Var02 Var03 Var05 Var07
## 118      40697   S06    NA       NA    NA    NA    NA
## 4769     41821   S05    NA       NA    NA    NA    NA
## 9217     42897   S03    NA 42343600    NA    NA    NA
## 9218     42897   S02    NA 38160300    NA    NA    NA
## 9219     42897   S01    NA  7329600    NA    NA    NA
```

```
## 9220     42897   S06    NA 19885500    NA    NA    NA
## 9221     42897   S05    NA 16610900    NA    NA    NA
## 9222     42897   S04    NA  9098800    NA    NA    NA
## 9223     42898   S03    NA 50074700    NA    NA    NA
## 9224     42898   S02    NA 45801300    NA    NA    NA
## 9225     42898   S01    NA  6121400    NA    NA    NA
## 9226     42898   S06    NA 32570900    NA    NA    NA
## 9227     42898   S05    NA 19331600    NA    NA    NA
## 9228     42898   S04    NA 11188200    NA    NA    NA
## 9637     42997   S03 95.43 32026000    NA    NA    NA
## 9638     42997   S02 13.26 19465000    NA    NA    NA
## 9639     42997   S01 58.83  6337000    NA    NA    NA
## 9640     42997   S06 49.21 13222800    NA    NA    NA
## 9641     42997   S05 90.40 13191900    NA    NA    NA
## 9642     42997   S04 36.72 34330700    NA    NA    NA
## 9643     43000   S03 97.19 38018600    NA    NA    NA
## 9644     43000   S02 13.20 16234300    NA    NA    NA
## 9645     43000   S01 59.28  3690900    NA    NA    NA
## 9646     43000   S06 48.88 10644000    NA    NA    NA
## 9647     43000   S05 89.90 11766100    NA    NA    NA
## 9648     43000   S04 36.95  7785800    NA    NA    NA
```

We can checking the correlations between variables. A correlation close to 1 or -1 is considered as a strong correlation between two variables. That is, the value of one variable is highly dependent to the other variable. We can use linear models to impute the missing values of one variable using another variable.



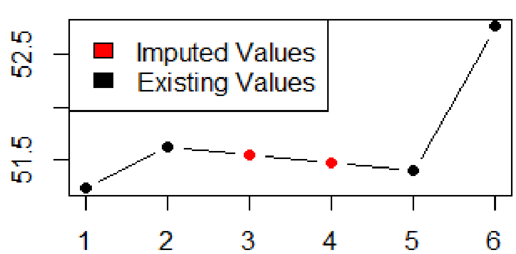
From the correlation plot, Var03, Var05, and Var07 are highly correlated to Var01. Var02 seems to be independent to the other variables.
We can impute the missing values of Var03, Var05, and Var07 where Var01 is available, using linear regression models. The summaries of the models (see Appendix A) show that the linear models are fitting to the data really well as indicated by an R-Squared score of almost 1. For records with all Var01, Var03, Var05, and Var07 missing, We will fill in the missing values using the linear interpolation method.

The linear interpolation method connects the previous and next available values by a straight line and fill in the missing values by the points fall on the line. For example, we have the following 2 missing values for group S01 Variable 01:

```
##       S01_Var01 S01_Var02
## 42895     51.23  15765600
## 42896     51.63   7321700
## 42897        NA   7329600
## 42898        NA   6121400
## 42899     51.40   8060600
## 42903     52.77   6567400
```

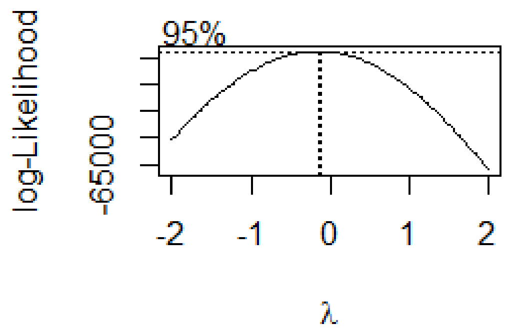
The missing values are filled by 51.55 and 51.47 which are on the line connecting 51.63 and 51.4 with equal distance.

```
##       S01_Var01 S01_Var02
## 42895  51.23000  15765600
## 42896  51.63000   7321700
## 42897  51.55333   7329600
## 42898  51.47667   6121400
## 42899  51.40000   8060600
## 42903  52.77000   6567400
```



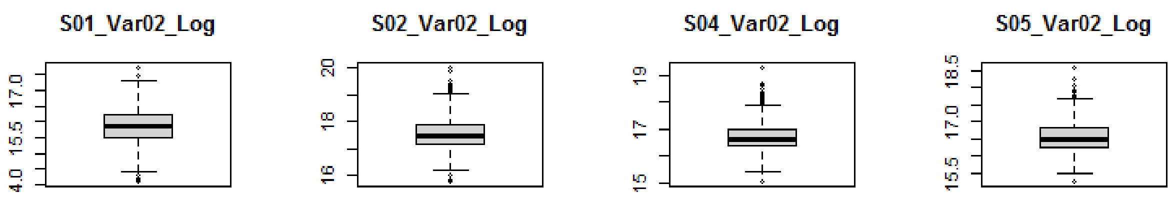
Data Transformation

As we have seen from the boxplots above, Var02 is highly right skewed. We will transform the variable using the Box-Cox Transformation method. The Box-Cox Transformation finds the optimal power that can be applied to the data so that the transformed data is close to the normal distribution.



A parameter close to 0 suggests that a log-transformation is appropriate.

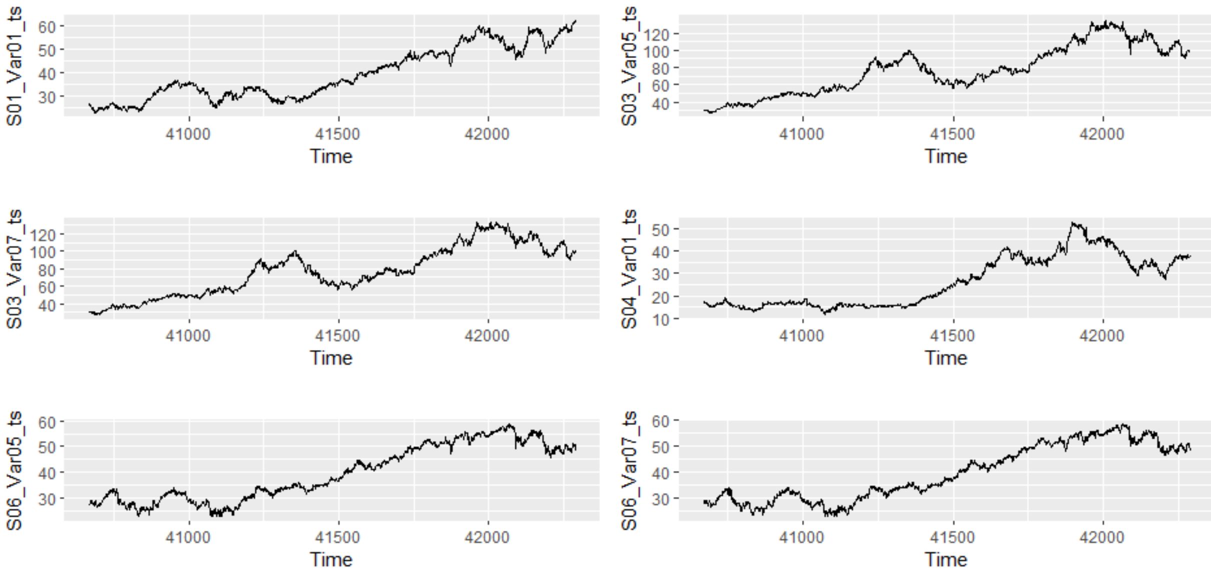
The followings are the boxplots of the log-transformated values for Var02 time series



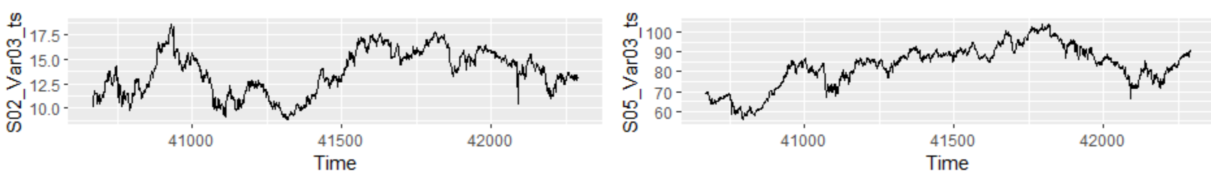
The variance is much stabler than before.

Time Series Exploration

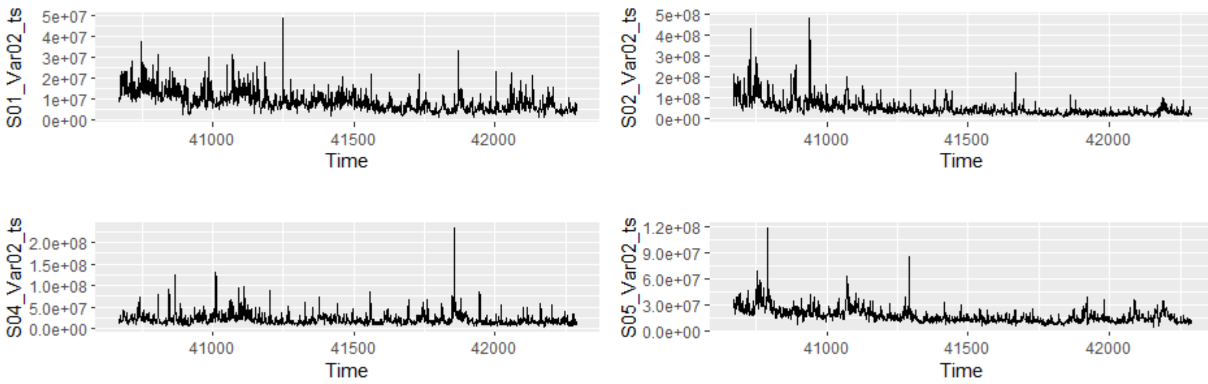
Time Plots



- There is no apparent seasonal behaviours in the time series for Var01, Var05, and Var07.
- There are apparent trends in the time series for Var01, Var05, and Var07.



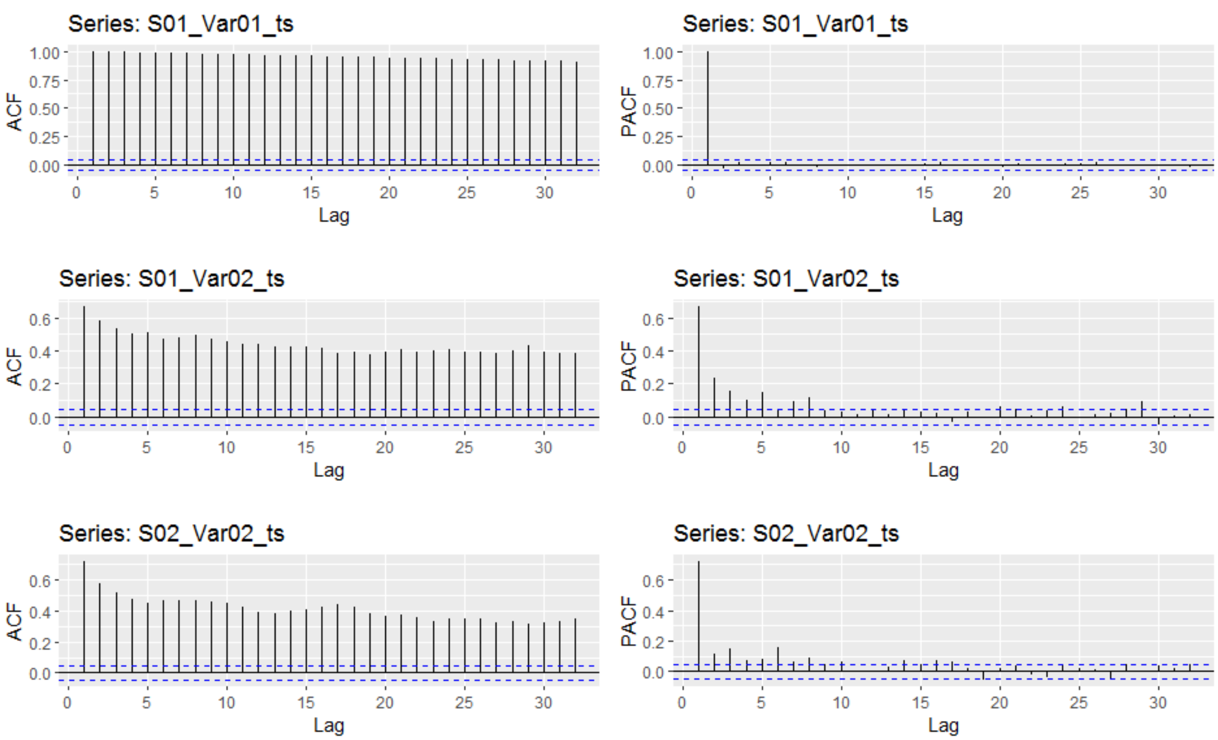
- There is no apparent seasonal behaviours in the time series for Var03.
- Var03 seems to have cyclic behaviours instead of trends.

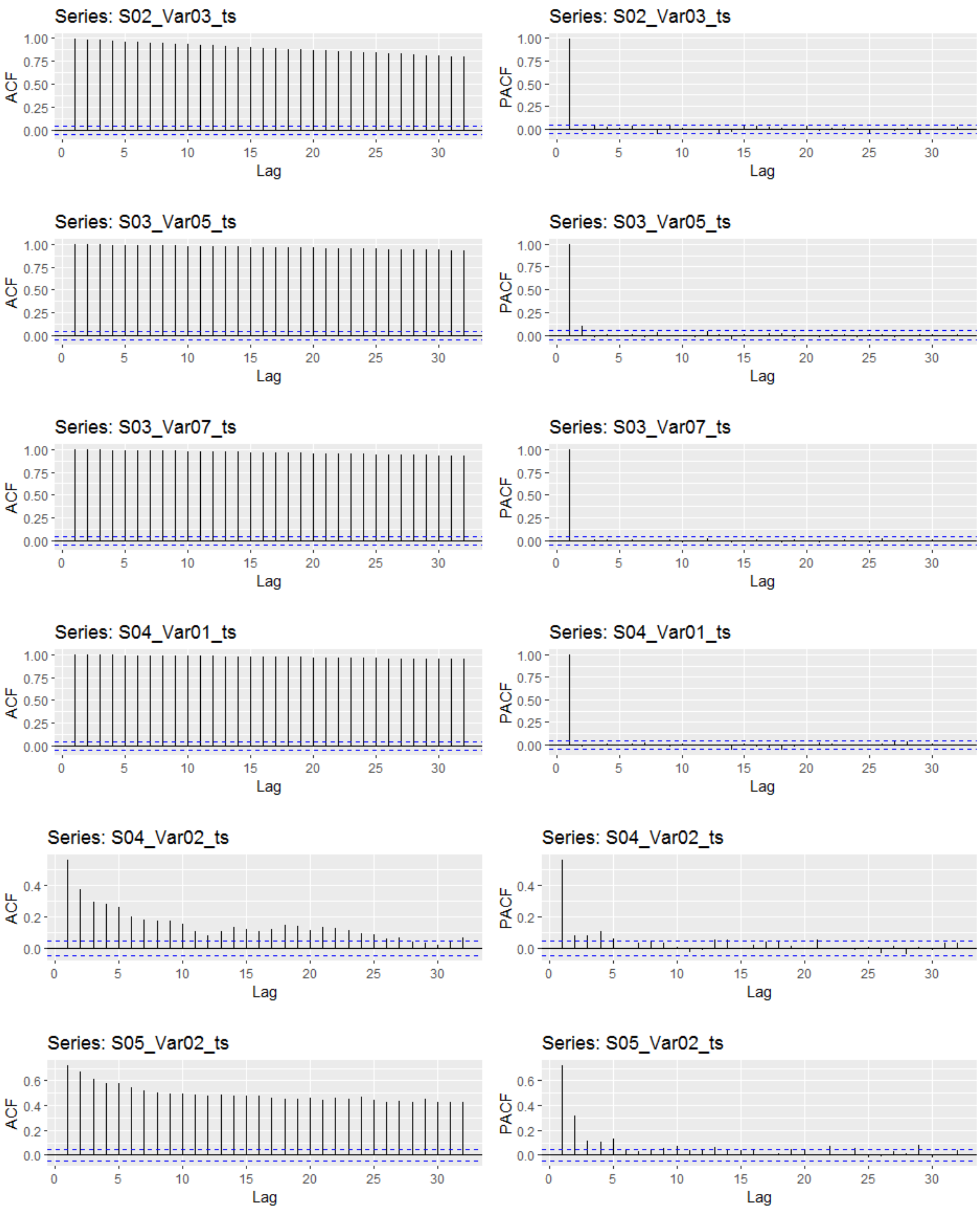


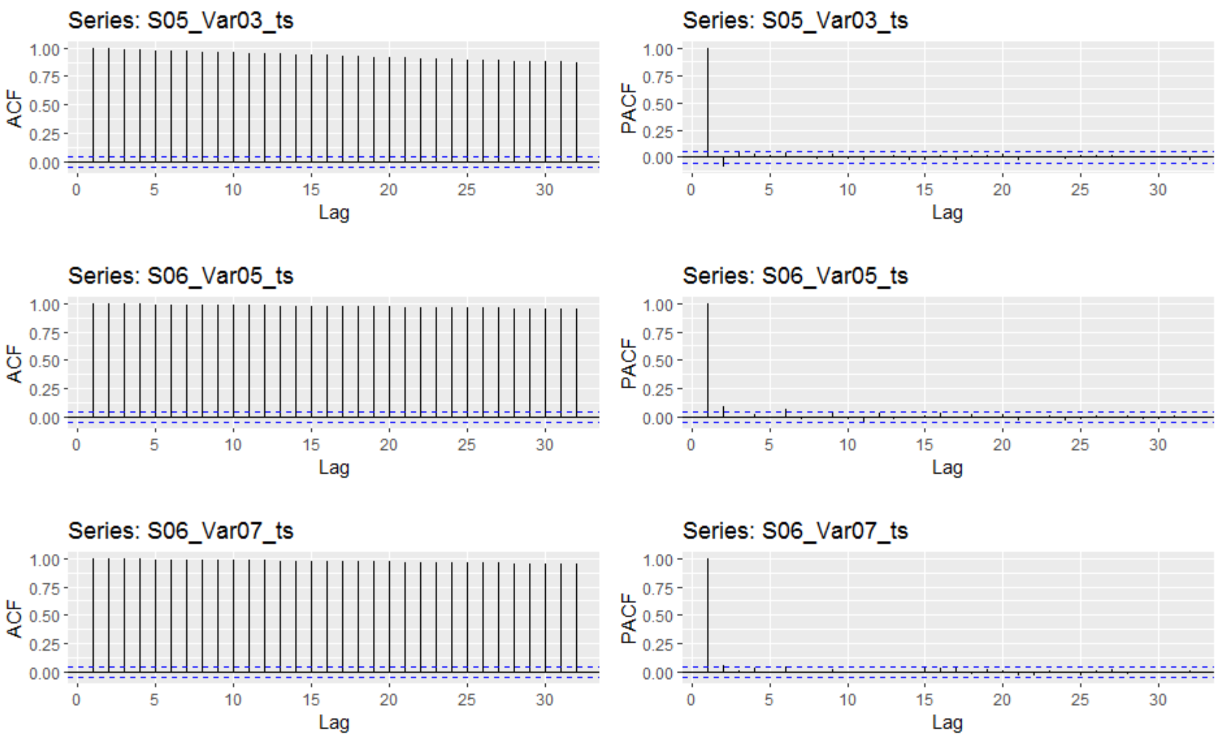
The time series for Var02 do not have apparent patterns. We may need to check the autocorrelations to verify if they are stationary.

ACF and PACF Plots

The ACF (Auto Correlation Function) plot and PACF (Partial Auto Correlation Function) show how strong that a value in a time series depends on its past values. A time series with high autocorrelation is non-stationary and hence there are predictable patterns.




Series: S02_Var03_ts
ACF
PACF
Lag
Series: S03_Var05_ts
ACF
PACF
Lag
Series: S03_Var07_ts
ACF
PACF
Lag
Series: S04_Var01_ts
ACF
PACF
Lag
Series: S04_Var02_ts
ACF
PACF
Lag
Series: S05_Var02_ts
ACF
PACF
Lag



All time series have significant autocorrelations (the ones with correlation higher than the threshold indicated by the blue dashed lines) for multiple lags in the ACF plots. The autocorrelations in the PACF plots are not so strong. We confirm that the time series are non-stationary.

Building models

Modeling Approach

The are generally two popular types of time series models: *Exponential Smoothing* (ETS) model and *ARIMA* (AutoRegressive Integrated Moving Average) model. The fitted values of an *Exponential Smoothing* model are affected by all past values, while the fitted values of an *ARIMA* model are affected by a number of most recent past values.

For each of the 12 time series, we perform the followings:

- Build the optimal *Exponential Smoothing* (ETS) model (log-transformation applied to the Var02 to stabilize the variance).
- Build the optimal *ARIMA* model (log-transformation applied to the Var02 to stabilize the variance).
- Perform cross-validation on the modeling method of *Exponential Smoothing*. Cross-validation is used to verify how well the model performs with unseen data.

- Perform cross-validation on the modeling method of *ARIMA*. Cross-validation is used to verify how well the model performs with unseen data.

- Compare the RMSE (Root Mean Squared Error) of models from the training data and cross-validation.

- Verify if there is any lack of fit by checking the residuals from the models.

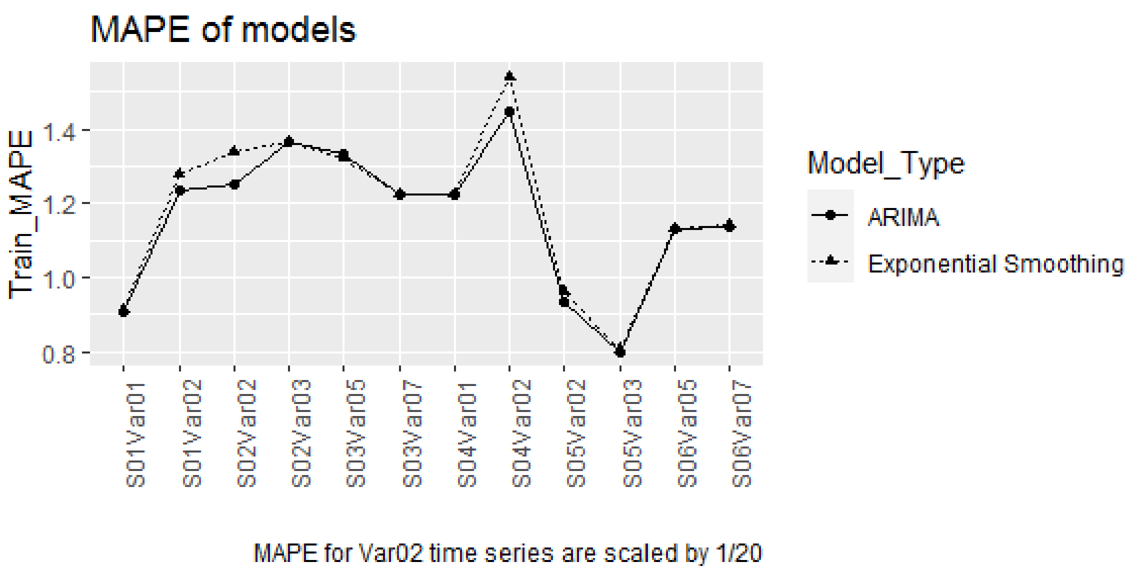
- Select the most appropriate model for forecasting.

The summaries of the *Exponential Smoothing* models are showed in Appendix B and the summaries of the *ARIMA* models are showed in Appendix C.

Model Performance

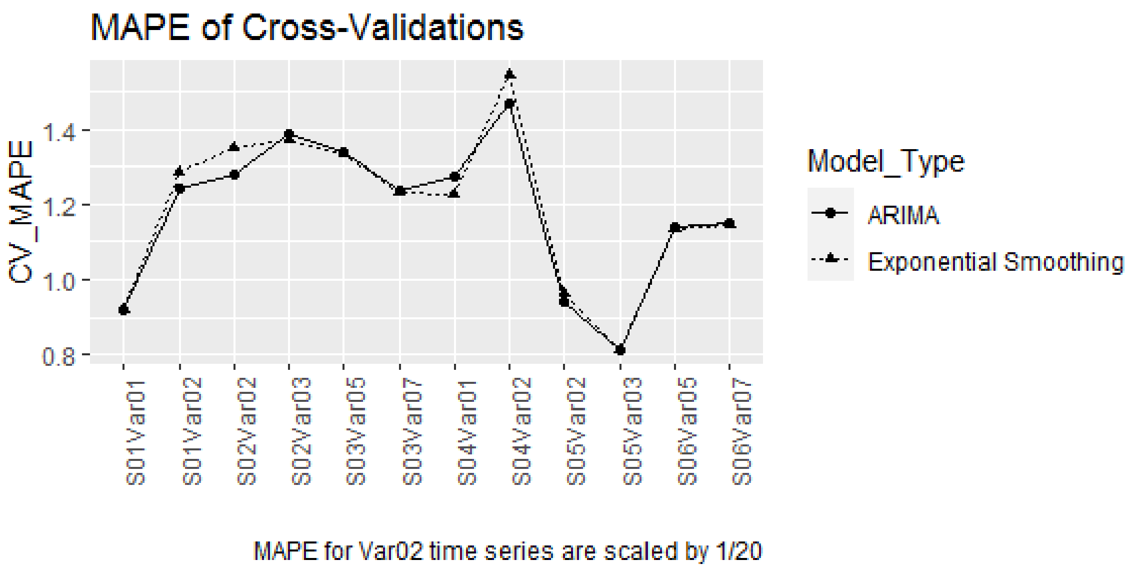
The MAPE (mean absolute percentage error) is one of the measurements that evaluate the distance between a model's fitted values and the actual values. For example, an MAPE of 0.8 (percent) indicates that the model's fitted values is about $\pm 0.80\%$ to the actual values **on average**. A smaller MAPE usually implies better performance.

The following are the MAPE values of the models.



The MAPE from the models are used to check the performance of the models based on the *training data*. From the plot, the *ARIMA* models and *Exponential Smoothing** models have very close performance. The *ARIMA* models are slightly better.

The following are the MAPE values from the outputs of Cross-Validations.

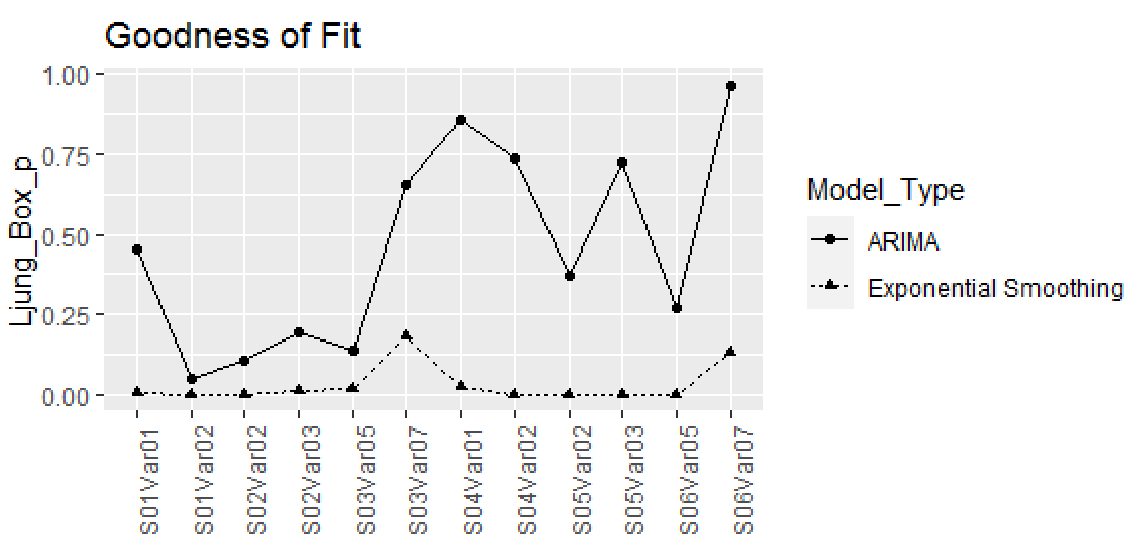


The MAPE of Cross-Validations are used to check the performance of the models with *unseen data*. From the plot, the *ARIMA* models and *Exponential Smoothing** models have very close performance. The *ARIMA* models are slightly better in most cases.

Goodness of Fit

We can also verify how well the models fit in the data. We use the Ljung Box test on the residuals of the models. The null hypothesis of the test is that there is no significant autocorrelations in the model's residuals, that is, the model explains all the variance / patterns of the data. A higher p-value implies that the model is fitting the data better. Details of the residuals verification are showed in Appendix D and E.

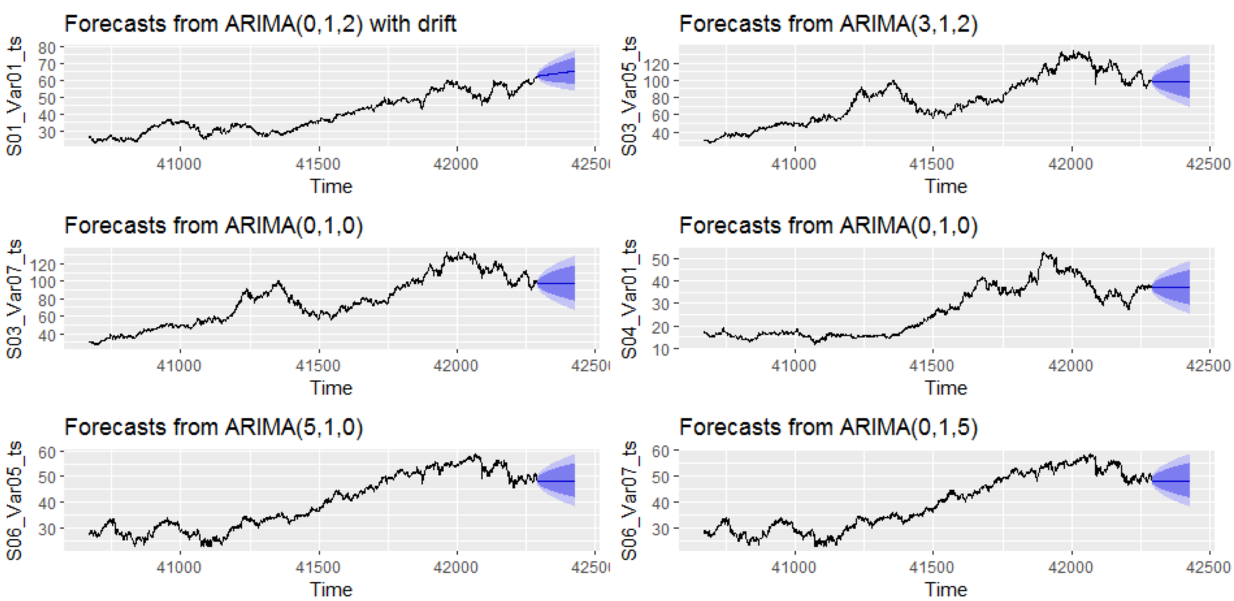
The p-values of the tests for the models are showed in the plot below.

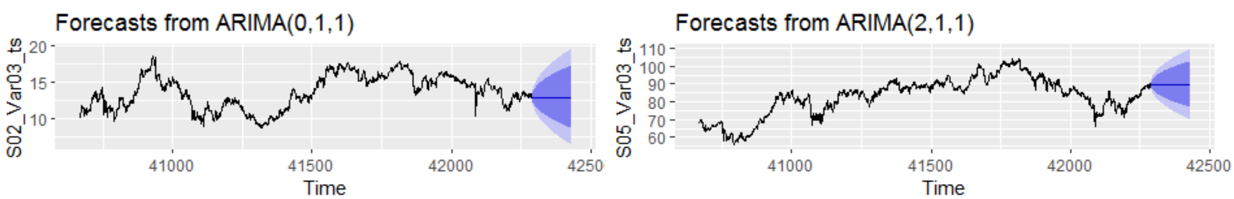


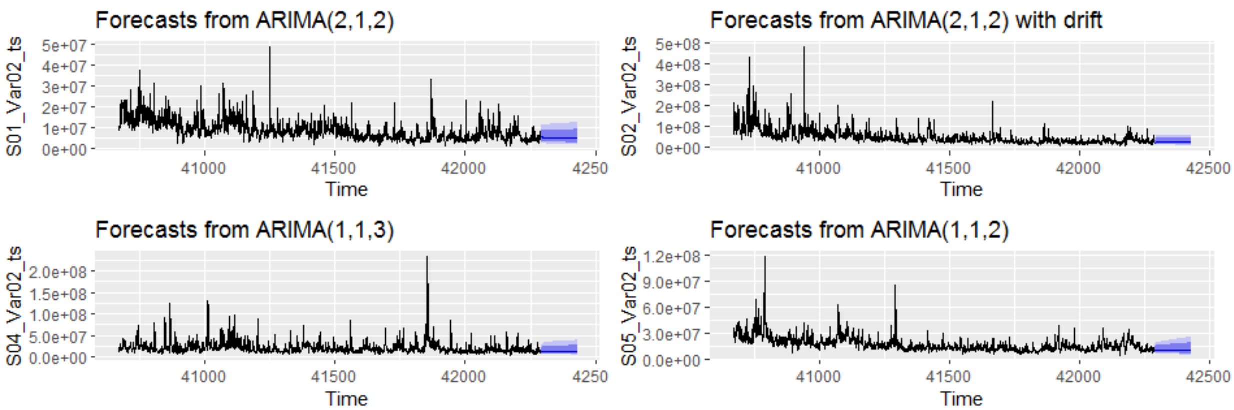
Based on the ljung-box test p-values, the ARIMA models are fitting to the data better so we will choose the ARIMA models for forecasting.

Model Forecast

We forecast the values for the 12 times in the next 140 periods.
The solid line represents the expected values. The dark blue area represents the 80% confidence interval, which indicates that there is 80% chance that the forecasted values will fall within that area. The light blue area represents the 95% confidence interval, there is 95% chance that the forecasted values will be in the area. The width of the confidence interval is increasing, which is reasonable since the uncertainty increases accordingly to the length of time.







Summary

This data was de-identified, and as such, an exercise in using pure analysis to generate a forecast, vs. relying on or leveraging data intuition. Further, with missing values and a highly-skewed variable, this data emphasized how pre-processing was critical to develop performant models and forecasts. While MAPE performance between the two model types (**Exponential Smoothing** and **ARIMA**) was close, we found that the ARIMA models have better ability (goodness-of-fit) in capturing the predictable patterns of the data. The ARIMA models are then determined to be the better models to produce plausible forecasts.

Appendices

A. Missing Value Imputation Linear Models Summaries

Model for imputing Var03

```
##
## Call:
## lm(formula = Var03 ~ Var01, data = raw_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.0364  -0.1437   0.0828   0.2463   1.3621
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.1585557  0.0101030  -15.69   <2e-16 ***
## Var01        0.9852475  0.0001821 5411.04   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5299 on 9704 degrees of freedom
##   (866 observations deleted due to missingness)
## Multiple R-squared:  0.9997, Adjusted R-squared:  0.9997
## F-statistic: 2.928e+07 on 1 and 9704 DF,  p-value: < 2.2e-16
```

Model for imputing Var05

```
##
## Call:
## lm(formula = Var05 ~ Var01, data = raw_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.0767  -0.1259   0.0747   0.2144   1.2858
##
## Coefficients:
##               Estimate Std. Error  t value Pr(>|t|)
## (Intercept) -0.0817381  0.0082988   -9.849   <2e-16 ***
## Var01        0.9929087  0.0001496 6638.681   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.4353 on 9704 degrees of freedom
##   (866 observations deleted due to missingness)
## Multiple R-squared:  0.9998, Adjusted R-squared:  0.9998
## F-statistic: 4.407e+07 on 1 and 9704 DF,  p-value: < 2.2e-16
```

Model for imputing Var07

```
##
## Call:
## lm(formula = Var07 ~ Var01, data = raw_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.4973 -0.1218  0.0771  0.2249  1.0302
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.0733544  0.0084798   -8.65   <2e-16 ***
## Var01        0.9928060  0.0001528 6496.25   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4448 on 9704 degrees of freedom
##   (866 observations deleted due to missingness)
## Multiple R-squared:  0.9998, Adjusted R-squared:  0.9998
## F-statistic: 4.22e+07 on 1 and 9704 DF,  p-value: < 2.2e-16
```

B. *Exponential Smoothing* (ETS) Models Summaries

S01_Var01

```
## ETS(M,N,N)
##
## Call:
##  ets(y = S01_Var01_ts)
##
##   Smoothing parameters:
##     alpha = 0.9999
##
##   Initial states:
##     l = 26.6044
##
##   sigma:  0.013
##
##      AIC     AICc      BIC
## 9704.278 9704.293 9720.452
##
## Training set error measures:
##                      ME      RMSE       MAE        MPE      MAPE      MASE
## Training set 0.02201555 0.5141259 0.3498242 0.04405534 0.9147066 0.9994097
##                    ACF1
## Training set 0.07946829
```

S01_Var02

```
## ETS(A,N,N)
##
## Call:
##  ets(y = S01_Var02_ts, lambda = 0)
##
##   Box-Cox transformation: lambda= 0
##
##   Smoothing parameters:
##     alpha = 0.4118
##
##   Initial states:
##     l = 16.19
##
##   sigma:  0.3289
```

```
##
##      AIC     AICc      BIC
## 8385.275 8385.290 8401.449
##
## Training set error measures:
##                    ME    RMSE     MAE       MPE     MAPE      MASE     ACF1
## Training set 278660.9 3390918 2195400 -5.515789 25.56845 0.8693303 0.074456
```

S02_Var02

```
## ETS(A,N,N)
##
## Call:
##  ets(y = S02_Var02_ts, lambda = 0)
##
##   Box-Cox transformation: lambda= 0
##
##   Smoothing parameters:
##     alpha = 0.4846
##
##   Initial states:
##     l = 18.5004
##
##   sigma:  0.3445
##
##      AIC     AICc      BIC
## 8535.430 8535.445 8551.604
##
## Training set error measures:
##                   ME     RMSE      MAE      MPE     MAPE      MASE      ACF1
## Training set 1776627 26544294 14400232 -5.94146 26.70957 0.9200695 0.1671709
```

S02_Var03

```
## ETS(A,Ad,N)
##
## Call:
##  ets(y = S02_Var03_ts)
##
##   Smoothing parameters:
##     alpha = 0.9999
##     beta  = 0.0022
```

```
##     phi   = 0.8
##
##   Initial states:
##     l = 9.5853
##     b = 0.5614
##
##   sigma:  0.2676
##
##      AIC     AICc      BIC
## 7719.151 7719.203 7751.500
##
## Training set error measures:
##                       ME      RMSE       MAE         MPE     MAPE      MASE
## Training set 0.000688936 0.2671646 0.1781965 -0.01612067 1.363142 0.9965499
##                    ACF1
## Training set 0.03222931
```

S03_Var05

```
## ETS(M,A,N)
##
## Call:
##  ets(y = S03_Var05_ts)
##
##   Smoothing parameters:
##     alpha = 0.8817
##     beta  = 1e-04
##
##   Initial states:
##     l = 31.2732
##     b = 0.0755
##
##   sigma:  0.0184
##
##      AIC     AICc      BIC
## 12882.53 12882.57 12909.49
##
## Training set error measures:
##                       ME     RMSE      MAE         MPE     MAPE      MASE
## Training set -0.03664842 1.500653 1.002857 -0.06643568 1.322686 0.9900135
```

```
##                      ACF1
## Training set -0.04393656
```

S03_Var07

```
## ETS(M,N,N)
##
## Call:
##  ets(y = S03_Var07_ts)
##
##   Smoothing parameters:
##     alpha = 0.9999
##
##   Initial states:
##     l = 30.5666
##
##   sigma:  0.0169
##
##      AIC     AICc      BIC
## 12594.60 12594.62 12610.78
##
## Training set error measures:
##                      ME     RMSE       MAE        MPE     MAPE      MASE
## Training set 0.04117162 1.343867 0.9299661 0.05716793 1.225678 0.9993851
##                    ACF1
## Training set 0.01190903
```

S04_Var01

```
## ETS(M,N,N)
##
## Call:
##  ets(y = S04_Var01_ts)
##
##   Smoothing parameters:
##     alpha = 0.9999
##
##   Initial states:
##     l = 17.1959
##
##   sigma:  0.0177
##
```

```
##      AIC     AICc      BIC
## 9232.875 9232.889 9249.049
##
## Training set error measures:
##                      ME      RMSE      MAE        MPE     MAPE      MASE
## Training set 0.01215544 0.5054759 0.323489 0.03157236 1.223006 0.9994009
##                    ACF1
## Training set 0.02437668
```

S04_Var02

```
## ETS(A,N,N)
##
## Call:
##  ets(y = S04_Var02_ts, lambda = 0)
##
##   Box-Cox transformation: lambda= 0
##
##   Smoothing parameters:
##     alpha = 0.438
##
##   Initial states:
##     l = 16.5866
##
##   sigma:  0.3994
##
##      AIC     AICc      BIC
## 9015.295 9015.310 9031.470
##
## Training set error measures:
##                   ME     RMSE     MAE       MPE     MAPE      MASE      ACF1
## Training set 1087592 11787369 6726514 -7.715586 30.80274 0.8971218 0.1623986
```

S05_Var02

```
## ETS(A,N,N)
##
## Call:
##  ets(y = S05_Var02_ts, lambda = 0)
##
##   Box-Cox transformation: lambda= 0
##
```

```
##   Smoothing parameters:
##     alpha = 0.3905
##
##   Initial states:
##     l = 17.1884
##
##   sigma:  0.2504
##
##      AIC     AICc      BIC
## 7501.250 7501.265 7517.424
##
## Training set error measures:
##                     ME    RMSE     MAE       MPE     MAPE      MASE       ACF1
## Training set 323192.3 5375498 3309989 -3.326119 19.22076 0.8805635 0.06831756
```

S05_Var03

```
## ETS(A,N,N)
##
## Call:
##  ets(y = S05_Var03_ts)
##
##   Smoothing parameters:
##     alpha = 0.9999
##
##   Initial states:
##     l = 68.1894
##
##   sigma:  0.9025
##
##      AIC     AICc      BIC
## 11659.98 11659.99 11676.15
##
## Training set error measures:
##                      ME      RMSE      MAE       MPE      MAPE      MASE
## Training set 0.01326311 0.9019141 0.650587 0.0102412 0.8060058 0.9993993
##                   ACF1
## Training set 0.1124908
```

S06_Var05

```
## ETS(A,N,N)
##
## Call:
##  ets(y = S06_Var05_ts)
##
##   Smoothing parameters:
##     alpha = 0.8687
##
##   Initial states:
##     l = 27.0696
##
##   sigma:  0.5657
##
##      AIC     AICc      BIC
## 10144.57 10144.59 10160.75
##
## Training set error measures:
##                      ME      RMSE      MAE        MPE     MAPE      MASE
## Training set 0.01499616 0.5653124 0.413597 0.02680856 1.130627 0.9979331
##                       ACF1
## Training set -0.0007854269
```

S06_Var07

```
## ETS(A,N,N)
##
## Call:
##  ets(y = S06_Var07_ts)
##
##   Smoothing parameters:
##     alpha = 0.9197
##
##   Initial states:
##     l = 27.3842
##
##   sigma:  0.5597
##
##      AIC     AICc      BIC
## 10110.20 10110.21 10126.37
##
## Training set error measures:
```

```
##                      ME      RMSE      MAE        MPE     MAPE      MASE
## Training set 0.01382323 0.5593534 0.416867 0.02377782 1.139434 0.9949819
##                     ACF1
## Training set 0.001390093
```

C. *ARIMA* Models Summaries

S01_Var01

```
## Series: S01_Var01_ts
## ARIMA(0,1,2) with drift
##
## Coefficients:
##          ma1      ma2   drift
##       0.0875  -0.0731  0.0220
## s.e.  0.0248   0.0250  0.0129
##
## sigma^2 = 0.2612:  log likelihood = -1210.39
## AIC=2428.77   AICc=2428.79   BIC=2450.33
##
## Training set error measures:
##                        ME      RMSE       MAE         MPE      MAPE      MASE
## Training set 1.925779e-05 0.5103994 0.3471124 -0.01588171 0.9089467 0.9916625
##                     ACF1
## Training set -0.00036009
```

S01_Var02

```
## Series: S01_Var02_ts
## ARIMA(2,1,2)
## Box Cox transformation: lambda= 0
##
## Coefficients:
##          ar1      ar2      ma1     ma2
##       1.1644  -0.2564  -1.6845  0.6895
## s.e.  0.0762   0.0514   0.0687  0.0672
##
## sigma^2 = 0.1003:  log likelihood = -435.11
## AIC=880.21   AICc=880.25   BIC=907.17
##
## Training set error measures:
##                  ME    RMSE     MAE      MPE     MAPE      MASE       ACF1
## Training set 308130 3289760 2102744 -6.17527 24.70748 0.8326409 -0.03069319
```

S02_Var02

```
## Series: S02_Var02_ts
## ARIMA(2,1,2) with drift
```

```
## Box Cox transformation: lambda= 0
##
## Coefficients:
##          ar1      ar2      ma1     ma2   drift
##       1.3074  -0.3733  -1.7772  0.7796  -8e-04
## s.e.  0.0537   0.0396   0.0458  0.0456   3e-04
##
## sigma^2 = 0.1061:  log likelihood = -480.3
## AIC=972.6   AICc=972.65   BIC=1004.94
##
## Training set error measures:
##                   ME     RMSE      MAE      MPE     MAPE      MASE      ACF1
## Training set 2694708 25290549 13499560 -5.14813 25.06196 0.8625231 0.1063655
```

S02_Var03

```
## Series: S02_Var03_ts
## ARIMA(0,1,1)
##
## Coefficients:
##          ma1
##       0.0385
## s.e.  0.0253
##
## sigma^2 = 0.07164:  log likelihood = -163
## AIC=330   AICc=330.01   BIC=340.79
##
## Training set error measures:
##                       ME      RMSE       MAE          MPE     MAPE      MASE
## Training set 0.001731614 0.2674853 0.1782592 -0.006086735 1.364126 0.9969002
##                       ACF1
## Training set -0.0007300862
```

S03_Var05

```
## Series: S03_Var05_ts
## ARIMA(3,1,2)
##
## Coefficients:
##           ar1      ar2      ar3     ma1     ma2
##       -0.7344  -1.0179  -0.1250  0.5770  0.9555
## s.e.   0.0323   0.0399   0.0291  0.0202  0.0319
```

```
##
## sigma^2 = 2.232:  log likelihood = -2948.33
## AIC=5908.65   AICc=5908.71   BIC=5941
##
## Training set error measures:
##                      ME     RMSE      MAE        MPE     MAPE      MASE
## Training set 0.04783634 1.491093 1.004371 0.06349747 1.329995 0.9915081
##                      ACF1
## Training set 0.0001048105
```

S03_Var07

```
## Series: S03_Var07_ts
## ARIMA(0,1,0)
##
## sigma^2 = 1.807:  log likelihood = -2779.69
## AIC=5561.37   AICc=5561.37   BIC=5566.76
##
## Training set error measures:
##                      ME     RMSE       MAE        MPE     MAPE      MASE
## Training set 0.04118231 1.343865 0.9299835 0.05721109 1.225727 0.9994037
##                    ACF1
## Training set 0.01181041
```

S04_Var01

```
## Series: S04_Var01_ts
## ARIMA(0,1,0)
##
## sigma^2 = 0.2557:  log likelihood = -1194.66
## AIC=2391.32   AICc=2391.32   BIC=2396.71
##
## Training set error measures:
##                      ME      RMSE      MAE        MPE    MAPE      MASE
## Training set 0.01216227 0.5054748 0.323494 0.03161619 1.22304 0.9994162
##                    ACF1
## Training set 0.02427769
```

S04_Var02

```
## Series: S04_Var02_ts
## ARIMA(1,1,3)
## Box Cox transformation: lambda= 0
```

```
##
## Coefficients:
##          ar1      ma1     ma2     ma3
##       0.8163  -1.3005  0.1657  0.1458
## s.e.  0.0537   0.0614  0.0456  0.0375
##
## sigma^2 = 0.1451:  log likelihood = -734.59
## AIC=1479.19   AICc=1479.22   BIC=1506.14
##
## Training set error measures:
##                   ME     RMSE     MAE       MPE     MAPE     MASE      ACF1
## Training set 1549863 11493142 6269572 -7.257727 28.88989 0.836179 0.1183397
```

S05_Var02

```
## Series: S05_Var02_ts
## ARIMA(1,1,2)
## Box Cox transformation: lambda= 0
##
## Coefficients:
##          ar1      ma1     ma2
##       0.7250  -1.3023  0.3347
## s.e.  0.0514   0.0625  0.0545
##
## sigma^2 = 0.05962:  log likelihood = -13.71
## AIC=35.43   AICc=35.45   BIC=56.99
##
## Training set error measures:
##                    ME    RMSE     MAE      MPE     MAPE      MASE       ACF1
## Training set 385819.4 5292300 3212931 -3.51762 18.65511 0.8547429 0.01825639
```

S05_Var03

```
## Series: S05_Var03_ts
## ARIMA(2,1,1)
##
## Coefficients:
##          ar1      ar2      ma1
##       0.8257  -0.1322  -0.7112
## s.e.  0.1536   0.0253   0.1539
##
## sigma^2 = 0.8006:  log likelihood = -2118.36
```

```
## AIC=4244.72   AICc=4244.74   BIC=4266.28
##
## Training set error measures:
##                     ME      RMSE       MAE        MPE      MAPE     MASE
## Training set 0.01408926 0.8936581 0.6450925 0.01106983 0.7988483 0.990959
##                     ACF1
## Training set -0.001023053
```

S06_Var05

```
## Series: S06_Var05_ts
## ARIMA(5,1,0)
##
## Coefficients:
##           ar1      ar2      ar3      ar4      ar5
##       -0.1294  -0.0138  -0.0481  -0.0034  -0.0925
## s.e.   0.0247   0.0250   0.0250   0.0250   0.0248
##
## sigma^2 = 0.3173:  log likelihood = -1367.17
## AIC=2746.33   AICc=2746.38   BIC=2778.68
##
## Training set error measures:
##                     ME      RMSE       MAE        MPE     MAPE      MASE
## Training set 0.01698597 0.5622222 0.4131744 0.03115046 1.131481 0.9969135
##                     ACF1
## Training set 0.0008496237
```

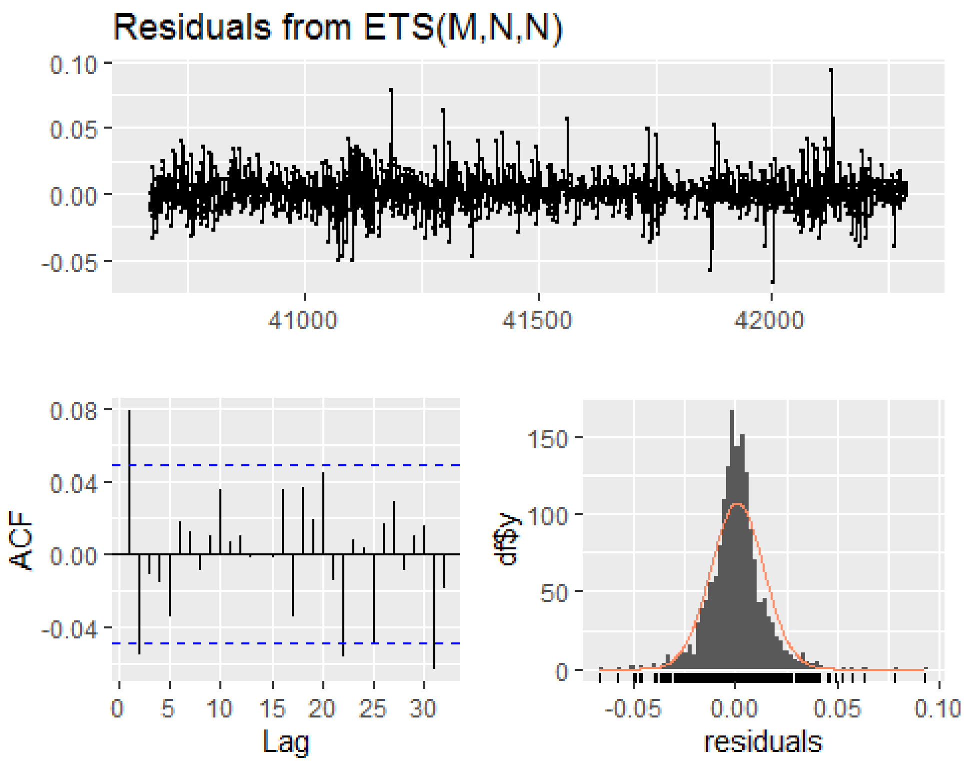
S06_Var07

```
## Series: S06_Var07_ts
## ARIMA(0,1,5)
##
## Coefficients:
##           ma1      ma2      ma3     ma4      ma5
##       -0.0804  -0.0245  -0.0454  0.0000  -0.0646
## s.e.   0.0248   0.0249   0.0250  0.0252   0.0247
##
## sigma^2 = 0.3118:  log likelihood = -1353.09
## AIC=2718.17   AICc=2718.23   BIC=2750.52
##
## Training set error measures:
##                   ME      RMSE       MAE       MPE    MAPE      MASE
```

```
## Training set 0.016486 0.5573656 0.4158866 0.0293722 1.13612 0.9926419
##                       ACF1
## Training set -0.0007213534
```

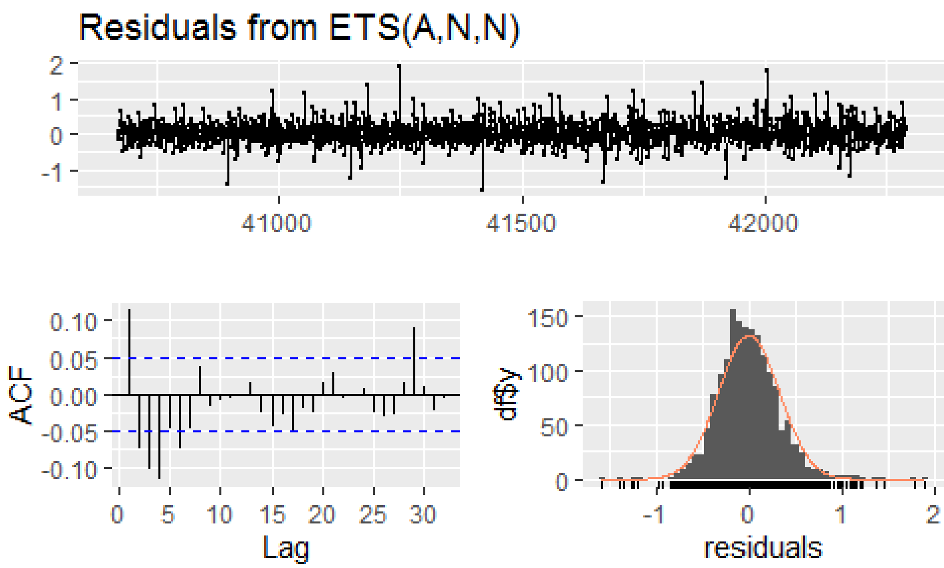
D. *Exponential Smoothing* (ETS) Models Residuals

S01_Var01



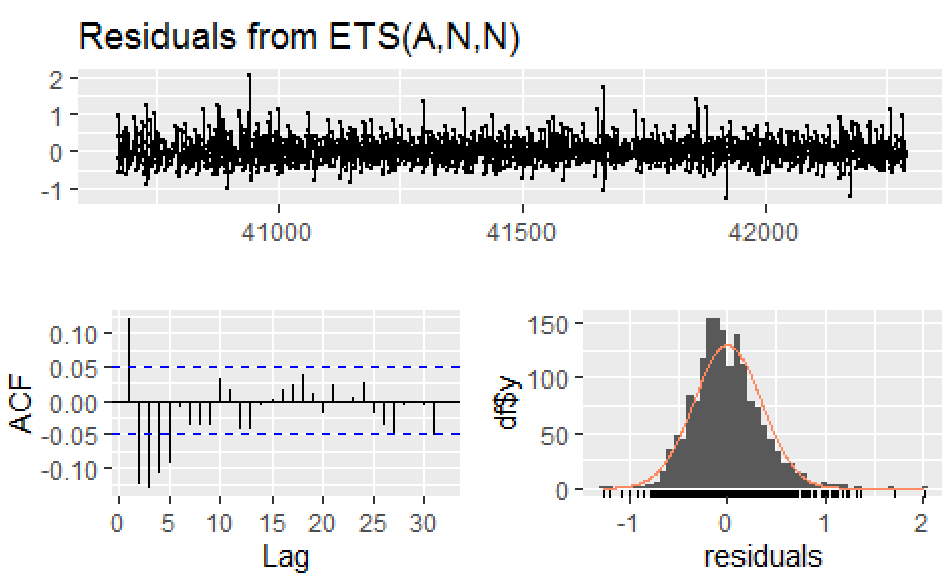

```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(M,N,N)
## Q* = 20.797, df = 8, p-value = 0.007706
##
## Model df: 2.   Total lags used: 10
```

S01_Var02




```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(A,N,N)
## Q* = 87.564, df = 8, p-value = 1.443e-15
##
## Model df: 2.   Total lags used: 10
```

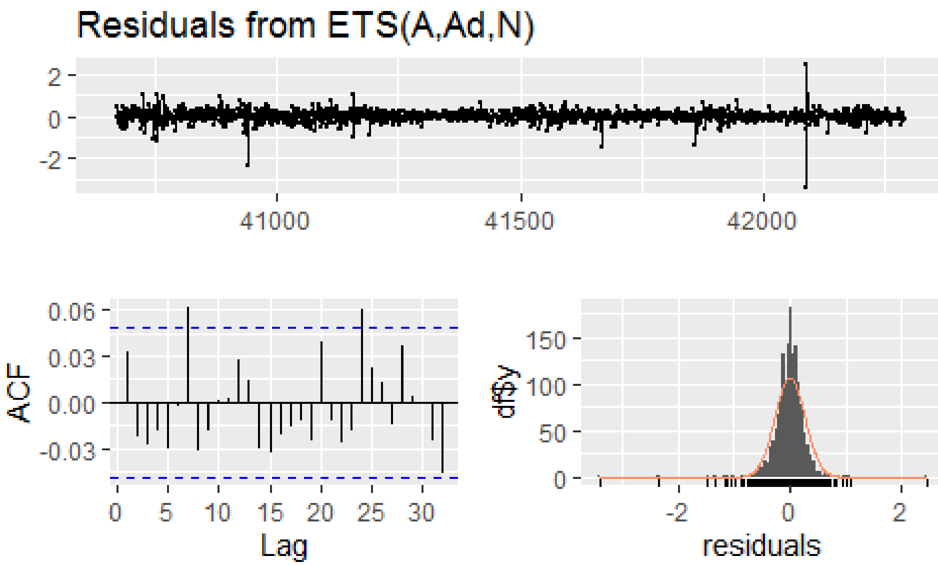
S02_Var02




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

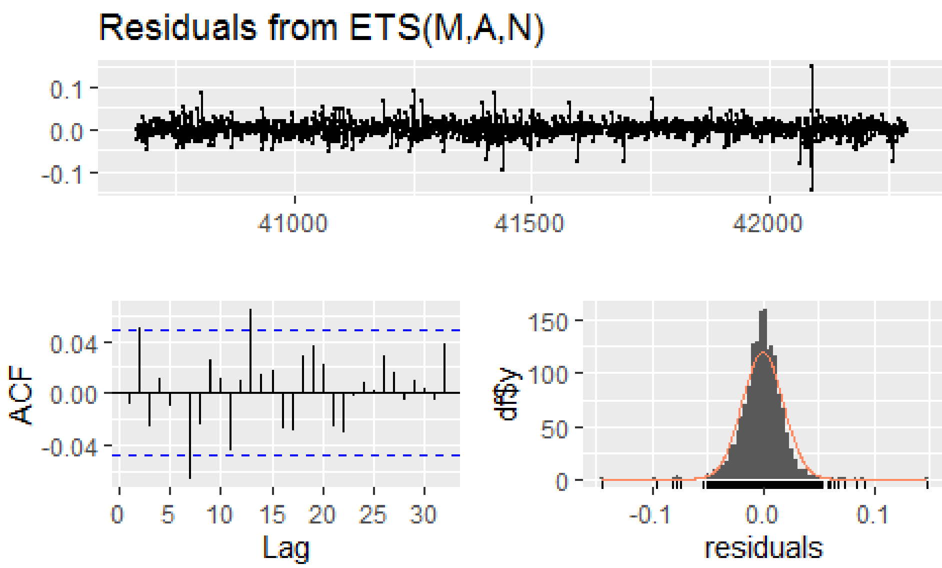
```
##
## data:  Residuals from ETS(A,N,N)
## Q* = 119.91, df = 8, p-value < 2.2e-16
##
## Model df: 2.   Total lags used: 10
```

S02_Var03



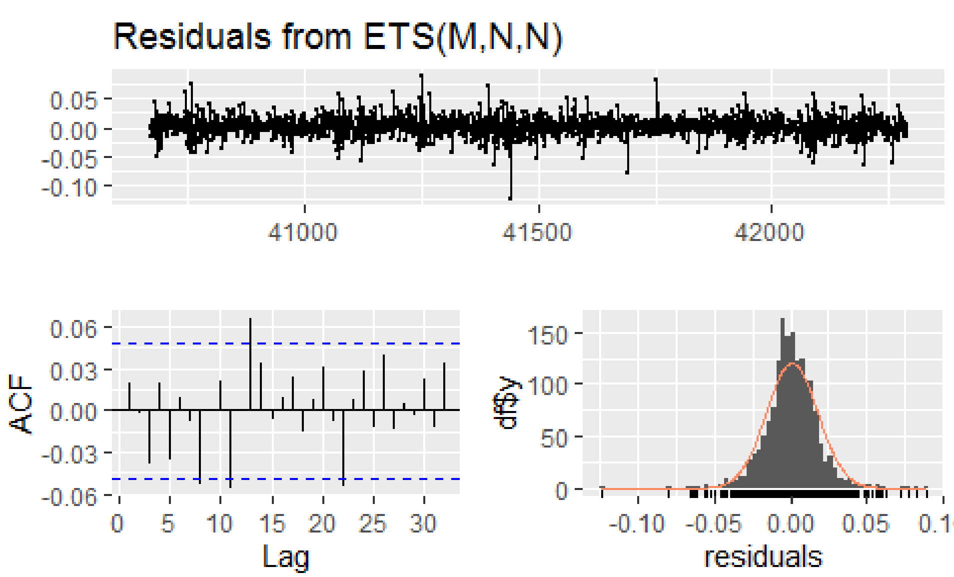

```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(A,Ad,N)
## Q* = 14.264, df = 5, p-value = 0.01402
##
## Model df: 5.   Total lags used: 10
```

S03_Var05




```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(M,A,N)
## Q* = 15.252, df = 6, p-value = 0.01839
##
## Model df: 4.   Total lags used: 10
```

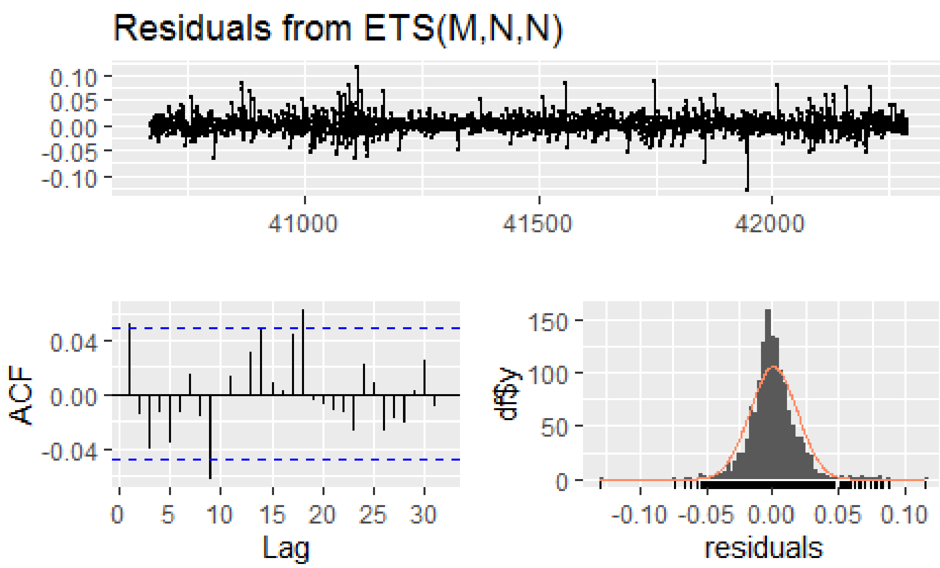
S03_Var07




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

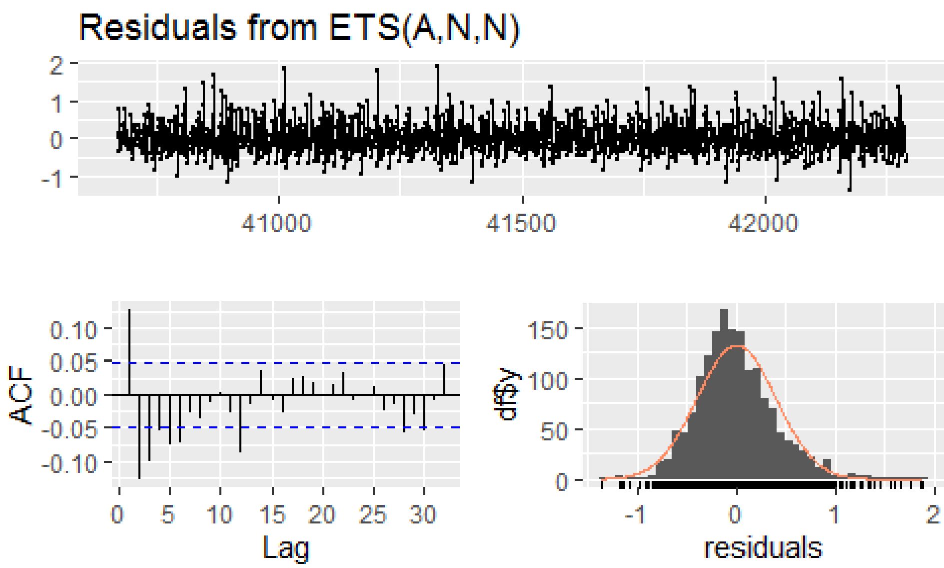
```
##
## data:  Residuals from ETS(M,N,N)
## Q* = 11.343, df = 8, p-value = 0.183
##
## Model df: 2.   Total lags used: 10
```

S04_Var01



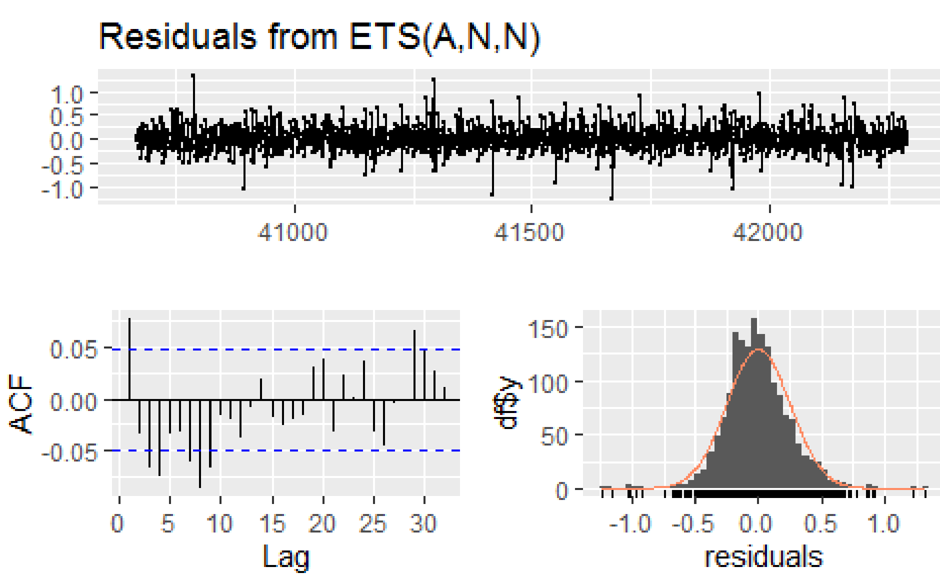

```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(M,N,N)
## Q* = 17.498, df = 8, p-value = 0.02532
##
## Model df: 2.   Total lags used: 10
```

S04_Var02




```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(A,N,N)
## Q* = 95.622, df = 8, p-value < 2.2e-16
##
## Model df: 2.   Total lags used: 10
```

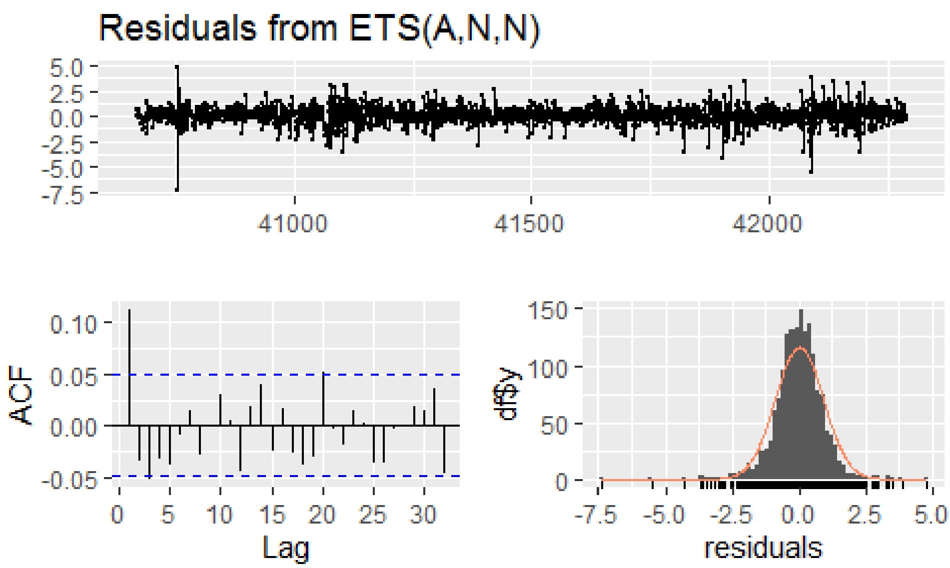
S05_Var02




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

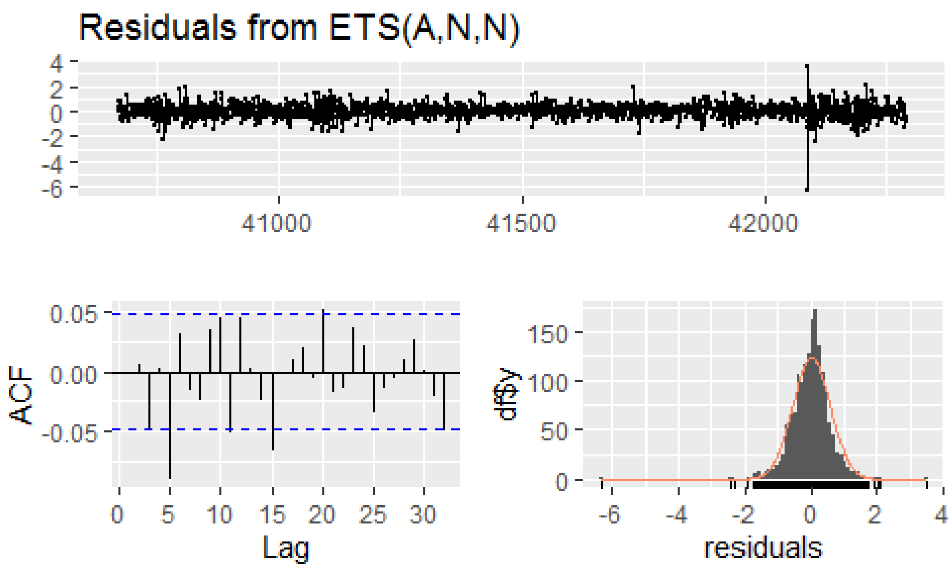
```
##
## data:  Residuals from ETS(A,N,N)
## Q* = 57.541, df = 8, p-value = 1.412e-09
##
## Model df: 2.   Total lags used: 10
```

S05_Var03



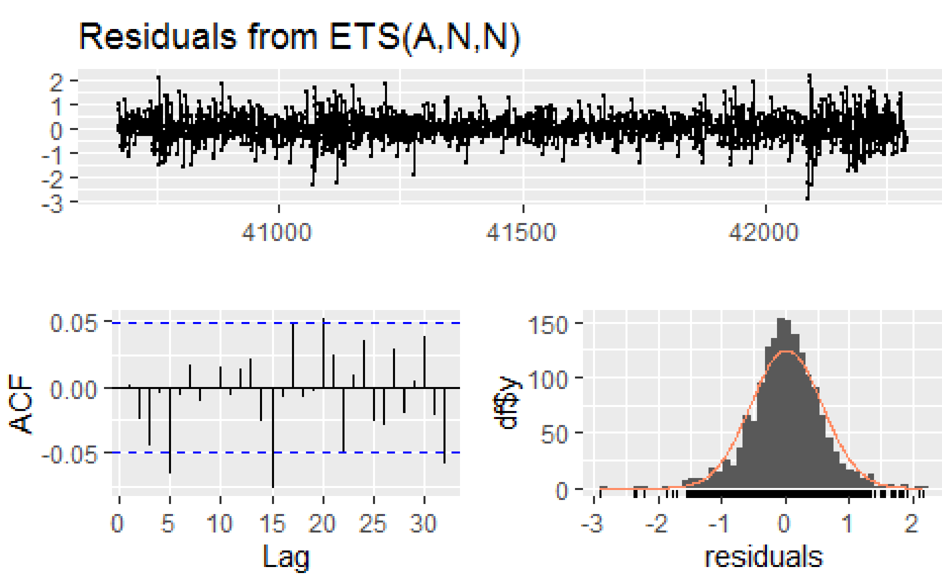

```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(A,N,N)
## Q* = 34.514, df = 8, p-value = 3.278e-05
##
## Model df: 2.   Total lags used: 10
```

S06_Var05




```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(A,N,N)
## Q* = 25.357, df = 8, p-value = 0.001352
##
## Model df: 2.   Total lags used: 10
```

S06_Var07

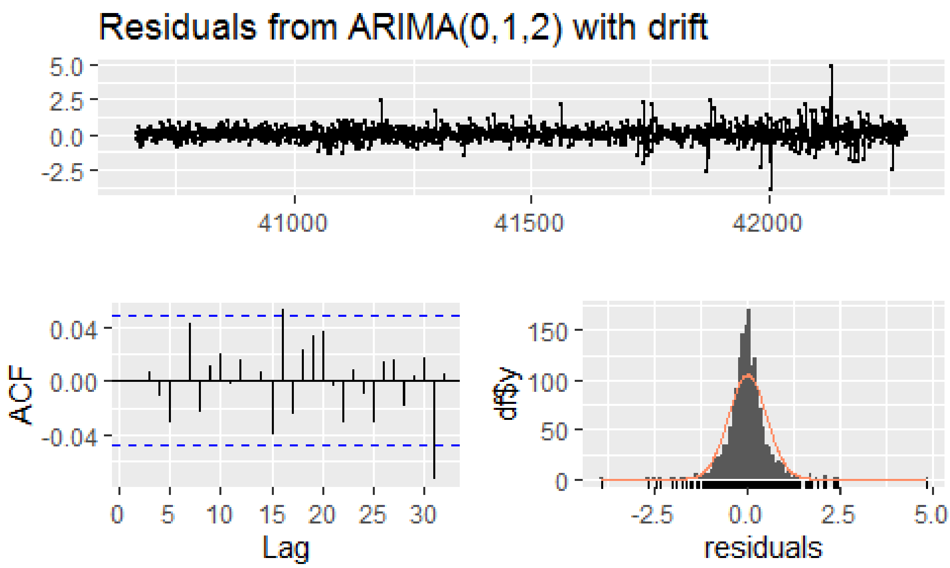



```
##
##  Ljung-Box test
```

```
##
## data:  Residuals from ETS(A,N,N)
## Q* = 12.398, df = 8, p-value = 0.1343
##
## Model df: 2.   Total lags used: 10
```

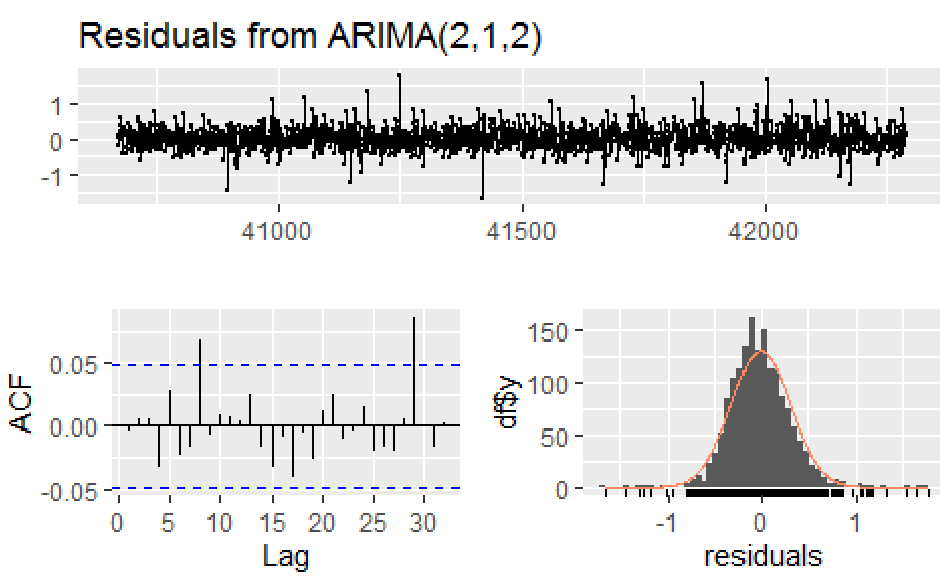
E. *ARIMA* Models Residuals

S01_Var01



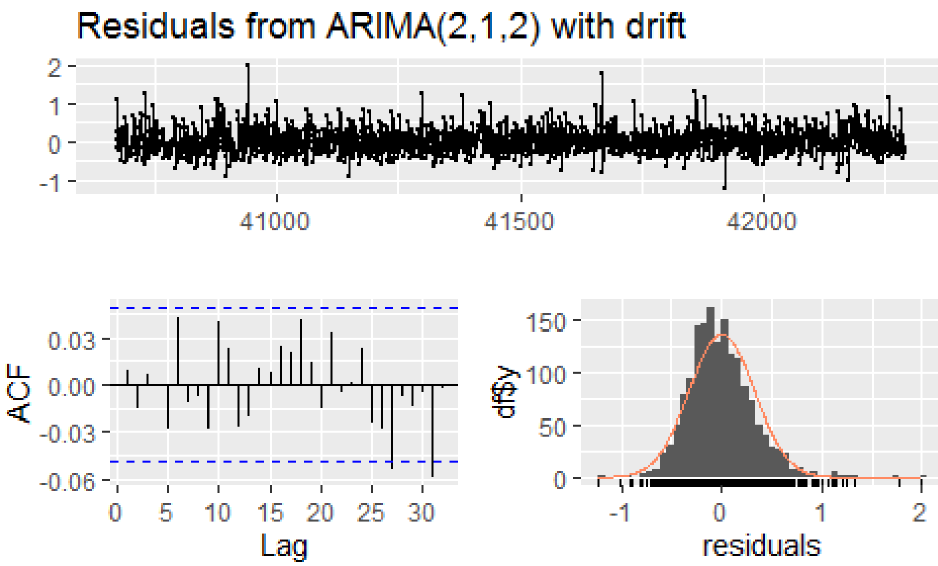

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,2) with drift
## Q* = 6.7638, df = 7, p-value = 0.4539
##
## Model df: 3.   Total lags used: 10
```

S01_Var02



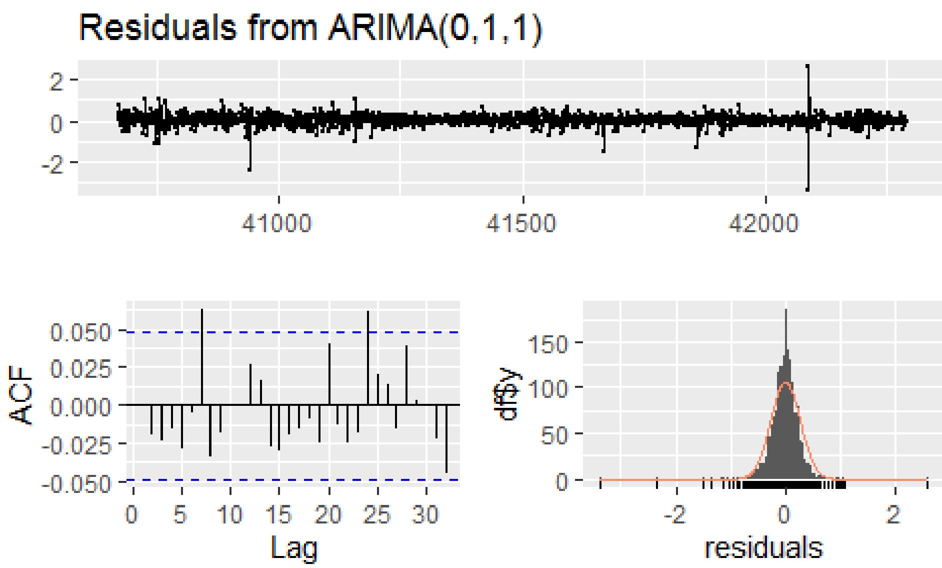
```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,1,2)
## Q* = 12.435, df = 6, p-value = 0.05295
##
## Model df: 4.   Total lags used: 10
```

S02_Var02



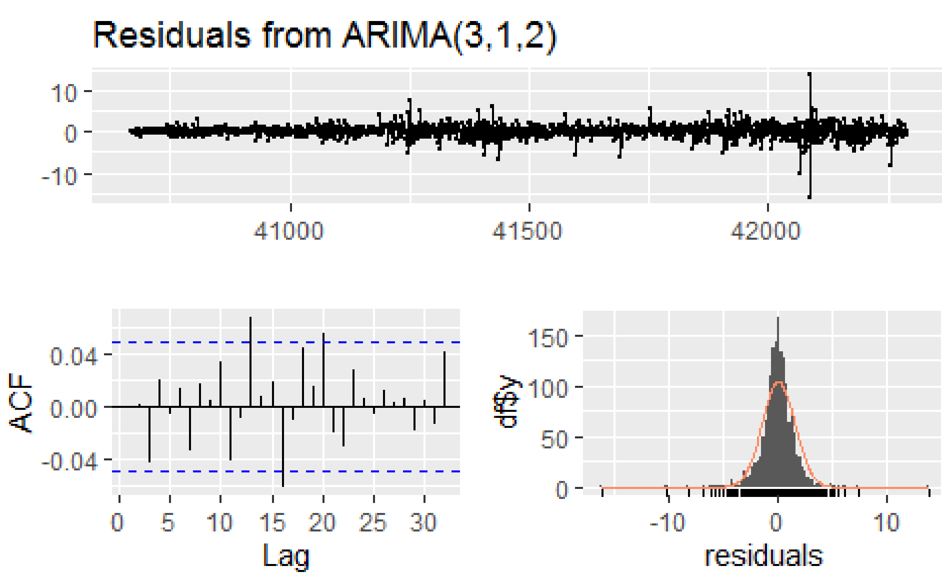

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,1,2) with drift
## Q* = 8.964, df = 5, p-value = 0.1105
##
## Model df: 5.   Total lags used: 10
```

S02_Var03




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,1)
## Q* = 12.282, df = 9, p-value = 0.1979
##
## Model df: 1.   Total lags used: 10
```

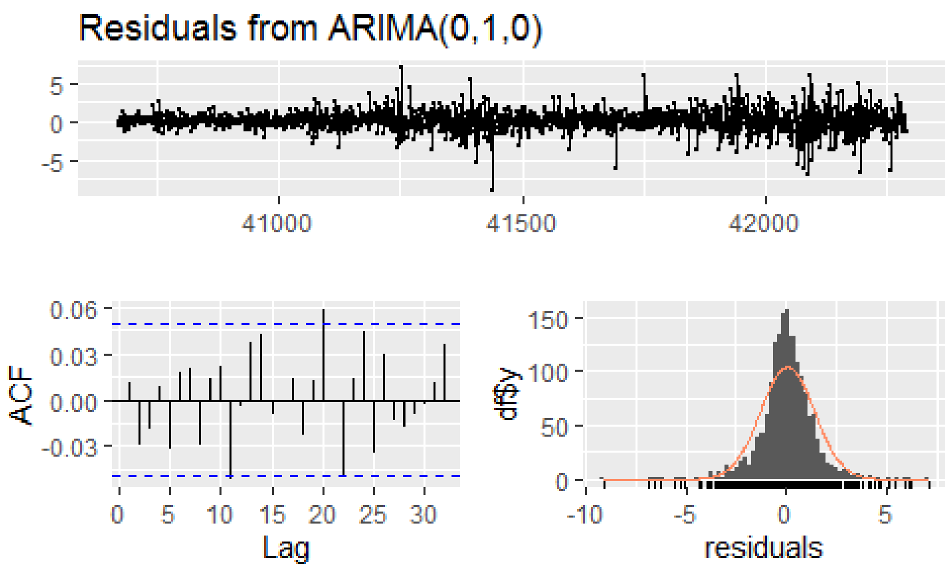
S03_Var05




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

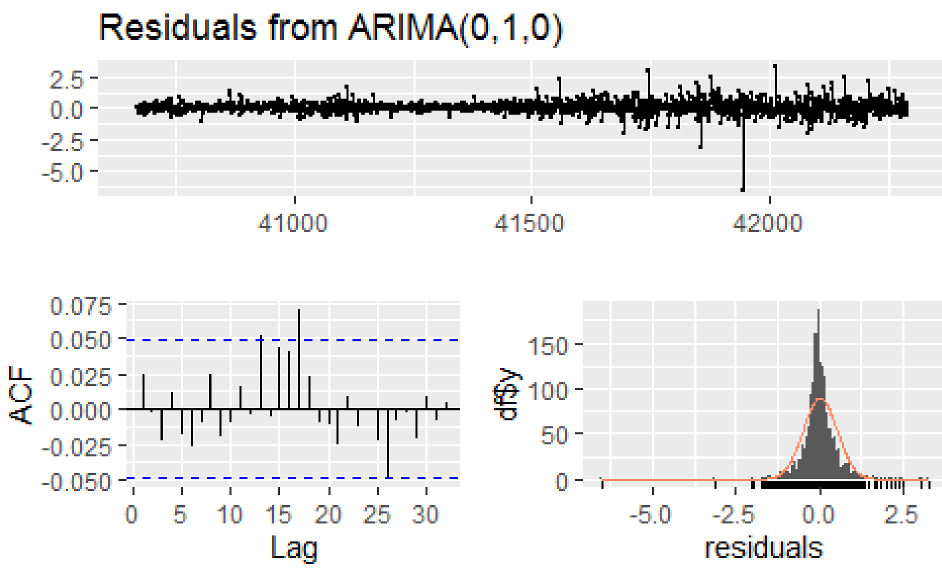
```
##
## data:  Residuals from ARIMA(3,1,2)
## Q* = 8.265, df = 5, p-value = 0.1422
##
## Model df: 5.   Total lags used: 10
```

S03_Var07



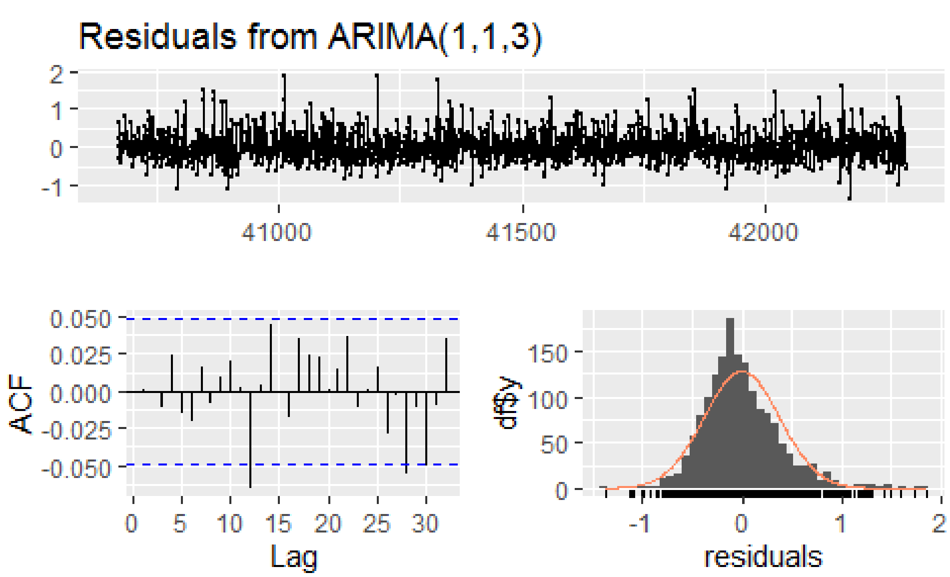

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,0)
## Q* = 7.7425, df = 10, p-value = 0.654
##
## Model df: 0.   Total lags used: 10
```

S04_Var01




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,0)
## Q* = 5.5178, df = 10, p-value = 0.854
##
## Model df: 0.   Total lags used: 10
```

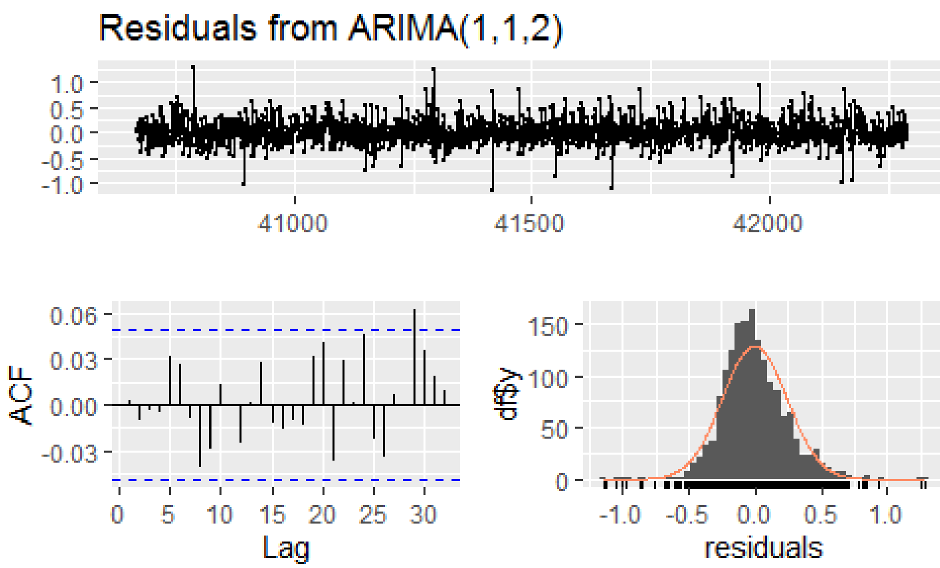
S04_Var02




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

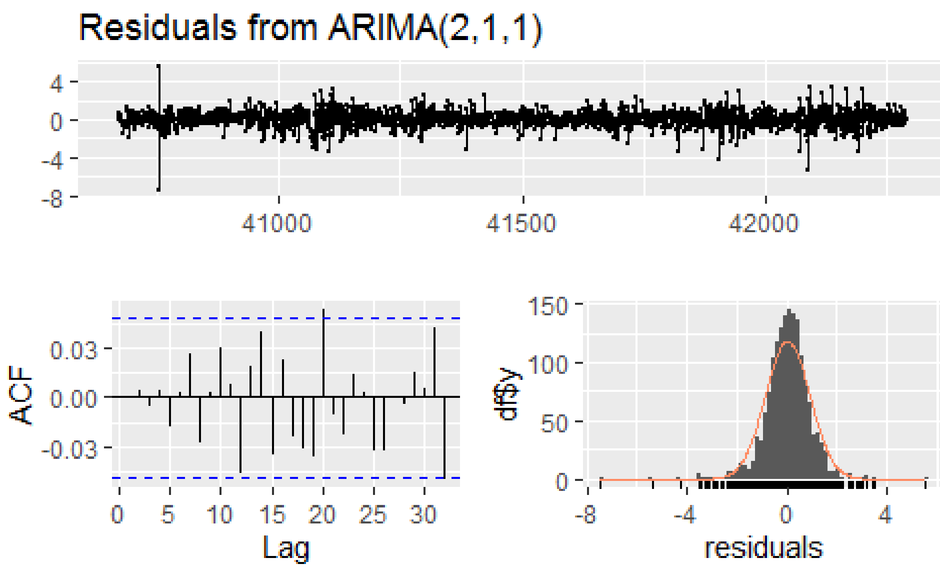
```
##
## data:  Residuals from ARIMA(1,1,3)
## Q* = 3.546, df = 6, p-value = 0.7378
##
## Model df: 4.   Total lags used: 10
```

S05_Var02



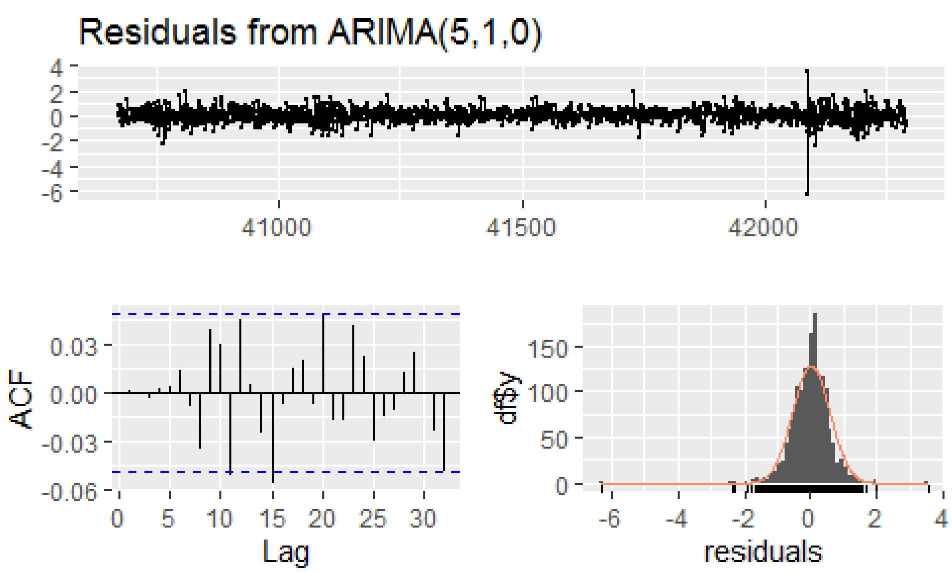

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,2)
## Q* = 7.5911, df = 7, p-value = 0.37
##
## Model df: 3.   Total lags used: 10
```

S05_Var03




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,1,1)
## Q* = 4.4602, df = 7, p-value = 0.7255
##
## Model df: 3.   Total lags used: 10
```

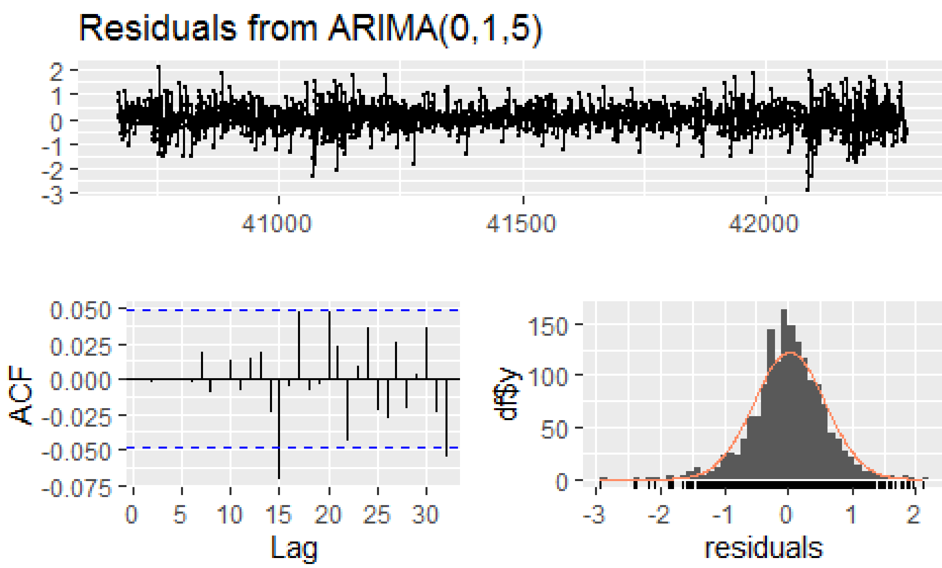
S06_Var05




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

```
##
## data:  Residuals from ARIMA(5,1,0)
## Q* = 6.3899, df = 5, p-value = 0.2701
##
## Model df: 5.   Total lags used: 10
```

S06_Var07




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,5)
## Q* = 1.0044, df = 5, p-value = 0.9622
##
## Model df: 5.   Total lags used: 10
```

Codes

```
# Loading packages
library(fpp2)
library(dplyr)
library(tidyverse)
library(corrplot)
library(MASS)
library(imputeTS)
library(patchwork)
library(ggplot2)



# Loading the data set
raw_df <- readxl::read_excel("Data Set for Class.xls")
raw_df <- as.data.frame(raw_df)
raw_df$group <- as.factor(raw_df$group)

# Summary of the first 1622 periods. The remaining 140 periods are blank and need to be foreca
sted
raw_summary <- summary(raw_df[c(1:(1622*6)),])
raw_summary



# Missing values in the observations
raw_na <- raw_df[c(1:(1622*6)),][apply(is.na(raw_df[c(1:(1622*6)),]),1,any),]
raw_na



# Checking the correlations between variables. If the correlations are high, we can use linear mo
dels to impute the missing values of one variable using another variable.
corrplot(cor(raw_df[,c(3:7)], use = "na.or.complete"),
        method = 'number', order = "hclust",
        type = 'lower', diag = FALSE, tl.srt = 0.1)



# Impute the missing values of Var03, Var05, Var07, where Var01 is available, using linear mod
```

```
els

var03_lm <- lm(Var03~Var01,raw_df)
var05_lm <- lm(Var05~Var01,raw_df)
var07_lm <- lm(Var07~Var01,raw_df)

raw_df$Var03[!is.na(raw_df$Var01) & is.na(raw_df$Var03)] <-
  predict(var03_lm,raw_df[!is.na(raw_df$Var01) & is.na(raw_df$Var03),])
raw_df$Var05[!is.na(raw_df$Var01) & is.na(raw_df$Var05)] <-
  predict(var03_lm,raw_df[!is.na(raw_df$Var01) & is.na(raw_df$Var05),])
raw_df$Var07[!is.na(raw_df$Var01) & is.na(raw_df$Var07)] <-
  predict(var03_lm,raw_df[!is.na(raw_df$Var01) & is.na(raw_df$Var07),])



# Gather data into one data frame, with one column per group per selected variable
S01_Var01 <- raw_df %>% filter(group=="S01") %>% dplyr::select("SeriesInd","Var01")
S01_Var02 <- raw_df %>% filter(group=="S01") %>% dplyr::select("SeriesInd","Var02")
S02_Var02 <- raw_df %>% filter(group=="S02") %>% dplyr::select("SeriesInd","Var02")
S02_Var03 <- raw_df %>% filter(group=="S02") %>% dplyr::select("SeriesInd","Var03")
S03_Var05 <- raw_df %>% filter(group=="S03") %>% dplyr::select("SeriesInd","Var05")
S03_Var07 <- raw_df %>% filter(group=="S03") %>% dplyr::select("SeriesInd","Var07")
S04_Var01 <- raw_df %>% filter(group=="S04") %>% dplyr::select("SeriesInd","Var01")
S04_Var02 <- raw_df %>% filter(group=="S04") %>% dplyr::select("SeriesInd","Var02")
S05_Var02 <- raw_df %>% filter(group=="S05") %>% dplyr::select("SeriesInd","Var02")
S05_Var03 <- raw_df %>% filter(group=="S05") %>% dplyr::select("SeriesInd","Var03")
S06_Var05 <- raw_df %>% filter(group=="S06") %>% dplyr::select("SeriesInd","Var05")
S06_Var07 <- raw_df %>% filter(group=="S06") %>% dplyr::select("SeriesInd","Var07")

main_df <- data.frame(S01_Var01=S01_Var01[,2],
                      S01_Var02=S01_Var02[,2],
                      S02_Var02=S02_Var02[,2],
                      S02_Var03=S02_Var03[,2],
                      S03_Var05=S03_Var05[,2],
                      S03_Var07=S03_Var07[,2],
                      S04_Var01=S04_Var01[,2],
                      S04_Var02=S04_Var02[,2],
                      S05_Var02=S05_Var02[,2],
                      S05_Var03=S05_Var03[,2],
                      S06_Var05=S06_Var05[,2],
```

```
                   S06_Var07=S06_Var07[,2])
row.names(main_df) <- S01_Var01$SeriesInd

main_df




# Boxplots of the variables for checking outliers and skewness

main_df_pre_process <- main_df

par(mfrow=c(3,4))
for (i in c(1:length(main_df_pre_process))) {
  boxplot(main_df_pre_process[,i], main=colnames(main_df_pre_process)[i])
}




# remove the extreme outliers to be imputed later
main_df$S02_Var03[which.max(main_df$S02_Var03)] <- NA
main_df$S06_Var05[which.max(main_df$S06_Var05)] <- NA
main_df$S06_Var07[which.max(main_df$S06_Var07)] <- NA




# Finding lambda for Box-Cox Transformation for Var02
boxcox(lm(raw_df$Var02 ~ 1))
# A number near 0 suggested a log transformation should be used
# Boxplot of Var02 after log transformation
par(mfrow=c(1,4))
boxplot(log(main_df_pre_process$S01_Var02), main="S01_Var02_Log")
boxplot(log(main_df_pre_process$S02_Var02), main="S02_Var02_Log")
boxplot(log(main_df_pre_process$S04_Var02), main="S04_Var02_Log")
boxplot(log(main_df_pre_process$S05_Var02), main="S05_Var02_Log")




# For remaining missing value, we will perform linear interpolation.
# The following are examples of missing values before linear interpolation.
main_df_pre_interpolation <-  main_df[c(1535:1540),]
```

```r
main_df_pre_interpolation

# perform linear interpolation
for (i in c(1:ncol(main_df))) {
  main_df[c(1:1622),i] <- na_interpolation(main_df[c(1:1622),i])
}

# The following are the values after imputation by linear interpolation.
main_df_post_interpolation <- main_df[c(1535:1540),]
main_df_post_interpolation




# Data is ready for modeling
# Create time series objects
S01_Var01_ts <- ts(main_df$S01_Var01[1:1622],start=as.integer(raw_df$SeriesInd[1]), frequency = 1)
S01_Var02_ts <- ts(main_df$S01_Var02[1:1622],start=as.integer(raw_df$SeriesInd[1]), frequency = 1)
S02_Var02_ts <- ts(main_df$S02_Var02[1:1622],start=as.integer(raw_df$SeriesInd[1]), frequency = 1)
S02_Var03_ts <- ts(main_df$S02_Var03[1:1622],start=as.integer(raw_df$SeriesInd[1]), frequency = 1)
S03_Var05_ts <- ts(main_df$S03_Var05[1:1622],start=as.integer(raw_df$SeriesInd[1]), frequency = 1)
S03_Var07_ts <- ts(main_df$S03_Var07[1:1622],start=as.integer(raw_df$SeriesInd[1]), frequency = 1)
S04_Var01_ts <- ts(main_df$S04_Var01[1:1622],start=as.integer(raw_df$SeriesInd[1]), frequency = 1)
S04_Var02_ts <- ts(main_df$S04_Var02[1:1622],start=as.integer(raw_df$SeriesInd[1]), frequency = 1)
S05_Var02_ts <- ts(main_df$S05_Var02[1:1622],start=as.integer(raw_df$SeriesInd[1]), frequency = 1)
S05_Var03_ts <- ts(main_df$S05_Var03[1:1622],start=as.integer(raw_df$SeriesInd[1]), frequency = 1)
S06_Var05_ts <- ts(main_df$S06_Var05[1:1622],start=as.integer(raw_df$SeriesInd[1]), frequency = 1)
S06_Var07_ts <- ts(main_df$S06_Var07[1:1622],start=as.integer(raw_df$SeriesInd[1]), frequency = 1)
```

```r
# Time Plot
autoplot(S01_Var01_ts) +
autoplot(S01_Var02_ts) +
autoplot(S02_Var02_ts) +
autoplot(S02_Var03_ts) +
autoplot(S03_Var05_ts) +
autoplot(S03_Var07_ts) +
autoplot(S04_Var01_ts) +
autoplot(S04_Var02_ts) +
autoplot(S05_Var02_ts) +
autoplot(S05_Var03_ts) +
autoplot(S06_Var05_ts) +
autoplot(S06_Var07_ts) +
  plot_layout(ncol = 1, guides = "collect")



# ACF and PACF
ggAcf(S01_Var01_ts) + ggPacf(S01_Var01_ts) +
ggAcf(S01_Var02_ts) + ggPacf(S01_Var02_ts) +
ggAcf(S02_Var02_ts) + ggPacf(S02_Var02_ts) +
ggAcf(S02_Var03_ts) + ggPacf(S02_Var03_ts) +
ggAcf(S03_Var05_ts) + ggPacf(S03_Var05_ts) +
ggAcf(S03_Var07_ts) + ggPacf(S03_Var07_ts) +
ggAcf(S04_Var01_ts) + ggPacf(S04_Var01_ts) +
ggAcf(S04_Var02_ts) + ggPacf(S04_Var02_ts) +
ggAcf(S05_Var02_ts) + ggPacf(S05_Var02_ts) +
ggAcf(S05_Var03_ts) + ggPacf(S05_Var03_ts) +
ggAcf(S06_Var05_ts) + ggPacf(S06_Var05_ts) +
ggAcf(S06_Var07_ts) + ggPacf(S06_Var07_ts) +
  plot_layout(ncol = 2, guides = "collect")



# Buildling models
# For each time series, we build an optimal ETS model an an optimal ARIMA model based on th
e AIC scores.
```

```r
S01_Var01_ets <- ets(S01_Var01_ts)
S01_Var01_arima <- auto.arima(S01_Var01_ts, stepwise=FALSE, approximation=FALSE)

S01_Var02_ets <- ets(S01_Var02_ts, lambda = 0)
S01_Var02_arima <- auto.arima(S01_Var02_ts, lambda = 0, stepwise=FALSE, approximation=FALSE)

S02_Var02_ets <- ets(S02_Var02_ts, lambda = 0)
S02_Var02_arima <- auto.arima(S02_Var02_ts, lambda = 0, stepwise=FALSE, approximation=FALSE)

S02_Var03_ets <- ets(S02_Var03_ts)
S02_Var03_arima <- auto.arima(S02_Var03_ts, stepwise=FALSE, approximation=FALSE)

S03_Var05_ets <- ets(S03_Var05_ts)
S03_Var05_arima <- auto.arima(S03_Var05_ts, stepwise=FALSE, approximation=FALSE)

S03_Var07_ets <- ets(S03_Var07_ts)
S03_Var07_arima <- auto.arima(S03_Var07_ts, stepwise=FALSE, approximation=FALSE)

S04_Var01_ets <- ets(S04_Var01_ts)
S04_Var01_arima <- auto.arima(S04_Var01_ts, stepwise=FALSE, approximation=FALSE)

S04_Var02_ets <- ets(S04_Var02_ts, lambda = 0)
S04_Var02_arima <- auto.arima(S04_Var02_ts, lambda = 0, stepwise=FALSE, approximation=FALSE)

S05_Var02_ets <- ets(S05_Var02_ts, lambda = 0)
S05_Var02_arima <- auto.arima(S05_Var02_ts, lambda = 0, stepwise=FALSE, approximation=FALSE)

S05_Var03_ets <- ets(S05_Var03_ts)
S05_Var03_arima <- auto.arima(S05_Var03_ts, stepwise=FALSE, approximation=FALSE)

S06_Var05_ets <- ets(S06_Var05_ts)
S06_Var05_arima <- auto.arima(S06_Var05_ts, stepwise=FALSE, approximation=FALSE)

S06_Var07_ets <- ets(S06_Var07_ts)
S06_Var07_arima <- auto.arima(S06_Var07_ts, stepwise=FALSE, approximation=FALSE)
```

```
# Perform Cross-Validation for both Exponential Smoothing (ETS) and ARIMA models
# The process takes more than an hour so the pre-calculated results at the end of the block can b
e used to save time

# fets <- function(x, h) {
#   forecast(ets(x), h = h)
# }
#
# farima <- function(x, h) {
#   forecast(auto.arima(x), h=h)
# }
#
# fets2 <- function(x, h) {
#   forecast(ets(x, lambda=0), h = h)
# }
#
# farima2 <- function(x, h) {
#   forecast(auto.arima(x, lambda=0), h=h)
# }
#
# #Function to calculate the MAPE using the result from the tsCV() function
# mape <- function(cv_e, ts) {
#   return(
#     mean(abs(cv_e[1:(length(cv_e)-sum(is.na(cv_e)))]/
#              ts[-sum(is.na(cv_e))]))*100 #multiply 100 to represent the number in percentage, whi
ch is consistent with the output from a time series model
#   )
# }
#
# e1 <- tsCV(S01_Var01_ts, fets, h=1)
# e2 <- tsCV(S01_Var01_ts, farima, h=1)
# S01_Var01_ets_cv <- mape(e1, S01_Var01_ts)
# S01_Var01_arima_cv <- mape(e2, S01_Var01_ts)
#
# e1 <- tsCV(S01_Var02_ts, fets2, h=1)
# e2 <- tsCV(S01_Var02_ts, farima2, h=1)
# S01_Var02_ets_cv <- mape(e1, S01_Var02_ts)
# S01_Var02_arima_cv <- mape(e2, S01_Var02_ts)
```

```
#
# e1 <- tsCV(S02_Var02_ts, fets2, h=1)
# e2 <- tsCV(S02_Var02_ts, farima2, h=1)
# S02_Var02_ets_cv <- mape(e1, S02_Var02_ts)
# S02_Var02_arima_cv <- mape(e2, S02_Var02_ts)
#
# e1 <- tsCV(S02_Var03_ts, fets, h=1)
# e2 <- tsCV(S02_Var03_ts, farima, h=1)
# S02_Var03_ets_cv <- mape(e1, S02_Var03_ts)
# S02_Var03_arima_cv <- mape(e2, S02_Var03_ts)
#
# e1 <- tsCV(S03_Var05_ts, fets, h=1)
# e2 <- tsCV(S03_Var05_ts, farima, h=1)
# S03_Var05_ets_cv <- mape(e1, S03_Var05_ts)
# S03_Var05_arima_cv <- mape(e2, S03_Var05_ts)
#
# e1 <- tsCV(S03_Var07_ts, fets, h=1)
# e2 <- tsCV(S03_Var07_ts, farima, h=1)
# S03_Var07_ets_cv <- mape(e1, S03_Var07_ts)
# S03_Var07_arima_cv <- mape(e2, S03_Var07_ts)
#
# e1 <- tsCV(S04_Var01_ts, fets, h=1)
# e2 <- tsCV(S04_Var01_ts, farima, h=1)
# S04_Var01_ets_cv <- mape(e1, S04_Var01_ts)
# S04_Var01_arima_cv <- mape(e2, S04_Var01_ts)
#
# e1 <- tsCV(S04_Var02_ts, fets2, h=1)
# e2 <- tsCV(S04_Var02_ts, farima2, h=1)
# S04_Var02_ets_cv <- mape(e1, S04_Var02_ts)
# S04_Var02_arima_cv <- mape(e2, S04_Var02_ts)
#
# e1 <- tsCV(S05_Var02_ts, fets2, h=1)
# e2 <- tsCV(S05_Var02_ts, farima2, h=1)
# S05_Var02_ets_cv <- mape(e1, S05_Var02_ts)
# S05_Var02_arima_cv <- mape(e2, S05_Var02_ts)
#
# e1 <- tsCV(S05_Var03_ts, fets, h=1)
# e2 <- tsCV(S05_Var03_ts, farima, h=1)
# S05_Var03_ets_cv <- mape(e1, S05_Var03_ts)
# S05_Var03_arima_cv <- mape(e2, S05_Var03_ts)
```

```r
#
# e1 <- tsCV(S06_Var05_ts, fets, h=1)
# e2 <- tsCV(S06_Var05_ts, farima, h=1)
# S06_Var05_ets_cv <- mape(e1, S06_Var05_ts)
# S06_Var05_arima_cv <- mape(e2, S06_Var05_ts)
#
# e1 <- tsCV(S06_Var07_ts, fets, h=1)
# e2 <- tsCV(S06_Var07_ts, farima, h=1)
# S06_Var07_ets_cv <- mape(e1, S06_Var07_ts)
# S06_Var07_arima_cv <- mape(e2, S06_Var07_ts)

# The followings are the pre-calculated results

S01_Var01_ets_cv <- 0.9177038
S01_Var01_arima_cv <- 0.9208322
S01_Var02_ets_cv <- 25.73889
S01_Var02_arima_cv <- 24.89347
S02_Var02_ets_cv <- 26.99113
S02_Var02_arima_cv <- 25.61222
S02_Var03_ets_cv <- 1.371266
S02_Var03_arima_cv <- 1.389502
S03_Var05_ets_cv <- 1.331599
S03_Var05_arima_cv <- 1.337125
S03_Var07_ets_cv <- 1.231979
S03_Var07_arima_cv <- 1.236644
S04_Var01_ets_cv <- 1.22556
S04_Var01_arima_cv <- 1.277181
S04_Var02_ets_cv <- 30.88217
S04_Var02_arima_cv <- 29.31935
S05_Var02_ets_cv <- 19.29705
S05_Var02_arima_cv <- 18.863
S05_Var03_ets_cv <- 0.814632
S05_Var03_arima_cv <- 0.8129907
S06_Var05_ets_cv <- 1.132529
S06_Var05_arima_cv <- 1.141299
S06_Var07_ets_cv <- 1.147087
S06_Var07_arima_cv <- 1.152259
```

```
# Gather the performance results in one dataframe for comparison
# The table includes the MAPE from the training data and the MAPE from the Cross-Validations
model_compare <-
  data.frame(Group=c("S01","S01","S01","S01",
                    "S02","S02","S02","S02",
                    "S03","S03","S03","S03",
                    "S04","S04","S04","S04",
                    "S05","S05","S05","S05",
                    "S06","S06","S06","S06"),
             Variable=c("Var01","Var01","Var02","Var02",
                       "Var02","Var02","Var03","Var03",
                       "Var05","Var05","Var07","Var07",
                       "Var01","Var01","Var02","Var02",
                       "Var02","Var02","Var03","Var03",
                       "Var05","Var05","Var07","Var07"),
             Model_Type=c("Exponential Smoothing","ARIMA","Exponential Smoothing","ARIMA
",
                         "Exponential Smoothing","ARIMA","Exponential Smoothing","ARIMA",
                         "Exponential Smoothing","ARIMA","Exponential Smoothing","ARIMA",
                         "Exponential Smoothing","ARIMA","Exponential Smoothing","ARIMA",
                         "Exponential Smoothing","ARIMA","Exponential Smoothing","ARIMA",
                         "Exponential Smoothing","ARIMA","Exponential Smoothing","ARIMA"),
             Model=c(as.character(S01_Var01_ets),as.character(S01_Var01_arima),
                    as.character(S01_Var02_ets),as.character(S01_Var02_arima),
                    as.character(S02_Var02_ets),as.character(S02_Var02_arima),
                    as.character(S02_Var03_ets),as.character(S02_Var03_arima),
                    as.character(S03_Var05_ets),as.character(S03_Var05_arima),
                    as.character(S03_Var07_ets),as.character(S03_Var07_arima),
                    as.character(S04_Var01_ets),as.character(S04_Var01_arima),
                    as.character(S04_Var02_ets),as.character(S04_Var02_arima),
                    as.character(S05_Var02_ets),as.character(S05_Var02_arima),
                    as.character(S05_Var03_ets),as.character(S05_Var03_arima),
                    as.character(S06_Var05_ets),as.character(S06_Var05_arima),
                    as.character(S06_Var07_ets),as.character(S06_Var07_arima)),
             CV_MAPE=c(S01_Var01_ets_cv, S01_Var01_arima_cv,
                      S01_Var02_ets_cv, S01_Var02_arima_cv,
                      S02_Var02_ets_cv, S02_Var02_arima_cv,
                      S02_Var03_ets_cv, S02_Var03_arima_cv,
                      S03_Var05_ets_cv, S03_Var05_arima_cv,
                      S03_Var07_ets_cv, S03_Var07_arima_cv,
```

```
                    S04_Var01_ets_cv, S04_Var01_arima_cv,
                    S04_Var02_ets_cv, S04_Var02_arima_cv,
                    S05_Var02_ets_cv, S05_Var02_arima_cv,
                    S05_Var03_ets_cv, S05_Var03_arima_cv,
                    S06_Var05_ets_cv, S06_Var05_arima_cv,
                    S06_Var07_ets_cv, S06_Var07_arima_cv),
           Train_MAPE=c(accuracy(S01_Var01_ets)[5],accuracy(S01_Var01_arima)[5],
                      accuracy(S01_Var02_ets)[5],accuracy(S01_Var02_arima)[5],
                      accuracy(S02_Var02_ets)[5],accuracy(S02_Var02_arima)[5],
                      accuracy(S02_Var03_ets)[5],accuracy(S02_Var03_arima)[5],
                      accuracy(S03_Var05_ets)[5],accuracy(S03_Var05_arima)[5],
                      accuracy(S03_Var07_ets)[5],accuracy(S03_Var07_arima)[5],
                      accuracy(S04_Var01_ets)[5],accuracy(S04_Var01_arima)[5],
                      accuracy(S04_Var02_ets)[5],accuracy(S04_Var02_arima)[5],
                      accuracy(S05_Var02_ets)[5],accuracy(S05_Var02_arima)[5],
                      accuracy(S05_Var03_ets)[5],accuracy(S05_Var03_arima)[5],
                      accuracy(S06_Var05_ets)[5],accuracy(S06_Var05_arima)[5],
                      accuracy(S06_Var07_ets)[5],accuracy(S06_Var07_arima)[5]))

# Adding the p=value from the ljung-box test to compare the goodness of fit for each model
model_compare$Ljung_Box_p[1] <- checkresiduals(S01_Var01_ets, plot=FALSE)$p.value
model_compare$Ljung_Box_p[2] <- checkresiduals(S01_Var01_arima, plot=FALSE)$p.value
model_compare$Ljung_Box_p[3] <- checkresiduals(S01_Var02_ets, plot=FALSE)$p.value
model_compare$Ljung_Box_p[4] <- checkresiduals(S01_Var02_arima, plot=FALSE)$p.value
model_compare$Ljung_Box_p[5] <- checkresiduals(S02_Var02_ets, plot=FALSE)$p.value
model_compare$Ljung_Box_p[6] <- checkresiduals(S02_Var02_arima, plot=FALSE)$p.value
model_compare$Ljung_Box_p[7] <- checkresiduals(S02_Var03_ets, plot=FALSE)$p.value
model_compare$Ljung_Box_p[8] <- checkresiduals(S02_Var03_arima, plot=FALSE)$p.value
model_compare$Ljung_Box_p[9] <- checkresiduals(S03_Var05_ets, plot=FALSE)$p.value
model_compare$Ljung_Box_p[10] <- checkresiduals(S03_Var05_arima, plot=FALSE)$p.value
model_compare$Ljung_Box_p[11] <- checkresiduals(S03_Var07_ets, plot=FALSE)$p.value
model_compare$Ljung_Box_p[12] <- checkresiduals(S03_Var07_arima, plot=FALSE)$p.value
model_compare$Ljung_Box_p[13] <- checkresiduals(S04_Var01_ets, plot=FALSE)$p.value
model_compare$Ljung_Box_p[14] <- checkresiduals(S04_Var01_arima, plot=FALSE)$p.value
model_compare$Ljung_Box_p[15] <- checkresiduals(S04_Var02_ets, plot=FALSE)$p.value
model_compare$Ljung_Box_p[16] <- checkresiduals(S04_Var02_arima, plot=FALSE)$p.value
model_compare$Ljung_Box_p[17] <- checkresiduals(S05_Var02_ets, plot=FALSE)$p.value
model_compare$Ljung_Box_p[18] <- checkresiduals(S05_Var02_arima, plot=FALSE)$p.value
model_compare$Ljung_Box_p[19] <- checkresiduals(S05_Var03_ets, plot=FALSE)$p.value
model_compare$Ljung_Box_p[20] <- checkresiduals(S05_Var03_arima, plot=FALSE)$p.value
```

```
model_compare$Ljung_Box_p[21] <- checkresiduals(S06_Var05_ets, plot=FALSE)$p.value
model_compare$Ljung_Box_p[22] <- checkresiduals(S06_Var05_arima, plot=FALSE)$p.value
model_compare$Ljung_Box_p[23] <- checkresiduals(S06_Var07_ets, plot=FALSE)$p.value
model_compare$Ljung_Box_p[24] <- checkresiduals(S06_Var07_arima, plot=FALSE)$p.value

model_compare




# Prepare data to plot the MAPE.
# Since Var02 has a number scale much larger than the other variables, we have to scale the RM
SE for Var02 models by multiplying 1/20 so they can be plotted in the same graph.
model_compare2 <- model_compare
model_compare2$CV_MAPE <- ifelse(model_compare2$Variable=="Var02",
                          model_compare2$CV_MAPE/20,
                          model_compare2$CV_MAPE)
model_compare2$Train_MAPE <- ifelse(model_compare2$Variable=="Var02",
                             model_compare2$Train_MAPE/20,
                             model_compare2$Train_MAPE)

# Plot the training data MAPE.
ggplot(model_compare2, aes(x=paste0(Group, Variable), y=Train_MAPE, group=Model_Type))
 +
  geom_line(aes(linetype=Model_Type))+
  geom_point(aes(shape=Model_Type))+
  theme(axis.text.x = element_text(angle = 90))+
  labs(title="MAPE of models",
       caption = "MAPE for Var02 time series are scaled by 1/20") +
  xlab("")

# Plot the Cross-Validation MAPE.
ggplot(model_compare2, aes(x=paste0(Group, Variable), y=CV_MAPE, group=Model_Type)) +
  geom_line(aes(linetype=Model_Type))+
  geom_point(aes(shape=Model_Type))+
  theme(axis.text.x = element_text(angle = 90))+
  labs(title="MAPE of Cross-Validations",
       caption = "MAPE for Var02 time series are scaled by 1/20") +
  xlab("")

# Plot the ljung-box test p-value
```

```
ggplot(model_compare2, aes(x=paste0(Group, Variable), y=Ljung_Box_p, group=Model_Typ
e)) +
  geom_line(aes(linetype=Model_Type))+
  geom_point(aes(shape=Model_Type))+
  theme(axis.text.x = element_text(angle = 90))+
  xlab("")
# The RMSE for ETS and ARIMA models are very close, with the ARIMA models perform slightl
y better.
# The the ljung-box test p-values, the ARIMA models are fitting to the data better so we will choo
se the ARIMA models for forecasting



# Forcasting
S01_Var01_forecast <- S01_Var01_arima %>% forecast(h=140)
S01_Var02_forecast <- S01_Var02_arima %>% forecast(h=140)
S02_Var02_forecast <- S02_Var02_arima %>% forecast(h=140)
S02_Var03_forecast <- S02_Var03_arima %>% forecast(h=140)
S03_Var05_forecast <- S03_Var05_arima %>% forecast(h=140)
S03_Var07_forecast <- S03_Var07_arima %>% forecast(h=140)
S04_Var01_forecast <- S04_Var01_arima %>% forecast(h=140)
S04_Var02_forecast <- S04_Var02_arima %>% forecast(h=140)
S05_Var02_forecast <- S05_Var02_arima %>% forecast(h=140)
S05_Var03_forecast <- S05_Var03_arima %>% forecast(h=140)
S06_Var05_forecast <- S06_Var05_arima %>% forecast(h=140)
S06_Var07_forecast <- S06_Var07_arima %>% forecast(h=140)



# Forecast Plot
S01_Var01_forecast %>% autoplot()
S01_Var02_forecast %>% autoplot()
S02_Var02_forecast %>% autoplot()
S02_Var03_forecast %>% autoplot()
S03_Var05_forecast %>% autoplot()
S03_Var07_forecast %>% autoplot()
S04_Var01_forecast %>% autoplot()
S04_Var02_forecast %>% autoplot()
S05_Var02_forecast %>% autoplot()
S05_Var03_forecast %>% autoplot()
```

```
S06_Var05_forecast %>% autoplot()
S06_Var07_forecast %>% autoplot()
```