2024/2025

MSCI570: Forecasting and Predictive Analytics

Individual Coursework

36686277 - Dataset NN5-025

Executive Summary

Our analysis explored different time series modelling methods for the NN5-025 dataset. Other than the machine learning models, for each of the other modelling methods manual models were created intuitively and automated models were generated by the system. Following table highlights the models generated and the best selected out of them

Table 0: Summary Table of all Models and Best Fits

	Manual	**Automated**	**Best Fit**
Exponential Smoothing	ANA, AAA	ANN	ANA
ARIMA	ARIMA(2,1,2)(3,1,1)	ARIMA(2,1,2)	ARIMA(2,1,2)(3,1,1)
Regression	**Model 1:** Thurs, Lag1, Lag7 **Model 2:**Thurs,Lag1, Lag7,Mon,Tue,Fri,Sat,Sun, Trend **Model 3:** Thurs,Lag1, Mon,Tue,Fri,Sat,Sun,Trend	**Model:** Trend, Lag1, Lag7, Mon, Wed, Thurs, Fri, Sat	**Model 3:** Thurs,Lag1, Mon,Tue,Fri,Sat,Sun,Trend
K-Nearest Neighbours	k=2, k=3, k=4, k=5	-	k=4

The models were then compared with each other and with the benchmark models of naive and seasonal naive. Based on sMAPE, it was concluded that the ARIMA(2,1,2)(3,1,1) model is the best fit and should be used for forecasting.

Introduction

This report analyses the different modelling techniques for time series and recommends the best model to be used to forecasted values to help clients better understand and be prepared for future withdrawal transactions from cash machines across the northwest region.

The objective of this report is to build forecasting models using different modelling techniques and to recommend the best fitted model.

The analysis starts by handling missing values in the dataset to avoid errors and dividing the dataset into test and train. The first modelling technique of exponential smoothing is then used to auto generate and manually build exponential models based on the preliminary analysis of trend and seasonality. The best out of these models is then concluded based on residual analysis and forecasting error terms.

Following this an autoARIMA model is fitted and multiple manual ARIMA models are iteratively crested based on ACF and PACF graph analysis. These models are also then evaluated to find the best fit based residual analysis and forecasting error term.

Regression models are also automatically generated and manually iteratively created based on correlation among parameters and with the dependent variable. As earlier, regression models are also evaluated to select the better model.

Finally, the K-Nearest Neighbours model is auto generated by the system. All of the best fit models per modelling technique are then compared to find the overall best fitted model. The concluded model is also evaluated against the benchmark models for better accuracy.

We end by making the recommendation of the best model based on our graphical and statistical analysis.

Summary

In the first assignment the dataset was explored and based on the analysis certain models were proposed. Following is a brief overview of the findings from the previous report before proceeding with model creation. My dataset is NN5-025.

- **Missing Value:** 13 identified missing values were linearly interpolated to avoid errors during analysis.
- **Outliers:** 21 outliers detected because of seasonal peaks/dips which were not changed to prevent loss of seasonality ***(Refer to appendix - 2).***
- **Skewness:** Skewness of 1.12785 was seen indicating non-normality of the dataset. Hence, all statistical tests used were non-parametric ***(Refer to appendix - 1).***
- **Decomposition:** Data fluctuated quite constantly around the mean so additive decomposition was used for the time series ***(Refer to appendix - 3)***
- **Trend:** The decomposition showed a slight upward trend. This was backed by the Mann-Kendall Trend Test ***(Refer to appendix - 4)***.
- **Seasonality:** The ACF and PACF graphs indicated weekly seasonality in the time series.
- **Stationarity:** The time series was not stationary as proven by the presence of trend and seasonality and statistically proven by the KPSS test ***(Refer to appendix - 4).***

Initialising the Data

Before starting analysis, the dataset is split into a train set and test set. Since the objective is to forecast a 14 day window, the test set contains the last 14 data points while the train set consists of the first 721 data points.

Exponential Smoothing

Model Creation

The train set is used to allow the system to automatically generate an exponential smoothing model of best fit. The system recommends the model “ANN” which indicates that the time series does not have trend or seasonality.

However, based on the preliminary analysis, it is concluded that the time series has a trend and weekly seasonality. Since the trend pattern is only slightly upward, both models having a trend i.e. AAA and not having a trend i.e. ANA are explored.

Model Accuracy

Model Analysis

To find the best manually fitted model, the AIC and BIC values are compared.

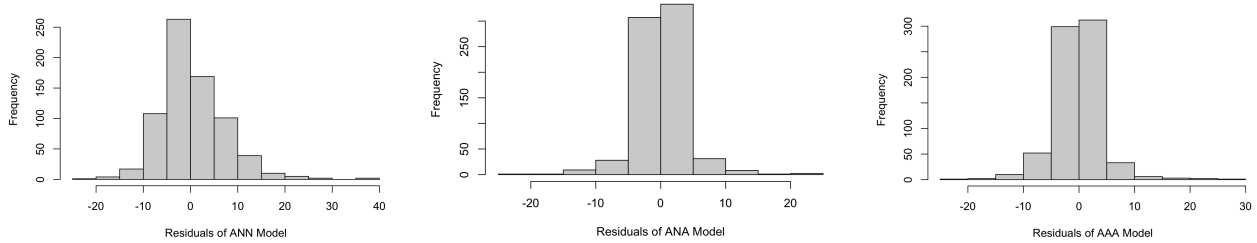
Table 1: AIC values for Exponential Smoothing Models

	ANN (Auto)	ANA	AAA
AIC	4837.044	4632.369	4828.153

From the output it is observed that the ANA model has lower AIC values as compared to the AAA model and ANN, the automatically generated model. Hence, statistically ANA might be the best fitted model but the forecast and the residuals need to be analysed before deciding.

Residual Analysis

Figure 1: Histograms for the Residuals of ANN, ANA and AAA Models



From Figure 1, it is seen that the histogram for the residuals of the ANA model are normally distributed whereas for the AAA model are slightly skewed to the right. The histogram for ANN model however, shows high right skewness. This shows that among all models (auto and manual), ANA is the best fit.

Forecasting Analysis

Table 2: Forecasting Error values for manually fitted Exponential Smoothing Models

	ANN (Auto)	ANA	AAA
MAE	6.157	2.308	2.542
RMSE	8.04	3.613	4.127

From Table2, it is observed that the forecasting errors for the ANA model are lower as compared to the AAA model and ANN model. Hence, the ANA model might be a better fit.

Conclusion

Therefore, among manually built models, AIC, normality of residuals and forecasting errors indicate that ANA is a better fitted model than AAA. Despite a slight upward trend, ANA better captures the features of the dataset.

When compared with the auto model of ANN, the AIC, normality of residuals and forecasting errors indicate that the best manual model, **ANA**, is a better fit among manual models as well as among manual and automated models.

ARIMA

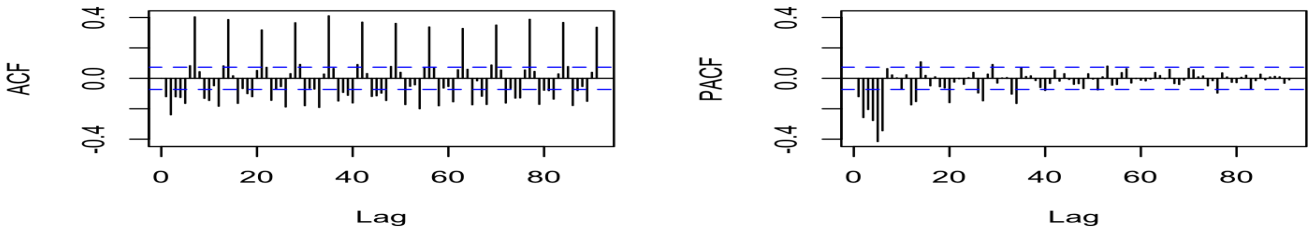
Automated ARIMA Model

Firstly an ARIMA model is fit in R using auto.arima() which recommends ARIMA(2,1,2) model. However, from the initial analysis, weekly seasonality is observed in the time series. Hence, more models are explored to determine the one with the best fitting.

Iterative Model Creation for Non Seasonal Components with Analysis

The modelling starts by the KPSS ***(Refer to appendix - 4)*** test which indicates non-stationarity of the time series. Thus, first order differencing is done on the time series and the KPSS test ***(Refer to appendix - 4)*** is rerun which now indicates that the time series is stationary. This is the first ARIMA model i.e. **Model 1: ARIMA(0,1,0)**

Figure 2: ACF and PACF graphs for First Order Differenced Time Series



Upon evaluating the PACF graphs from figure 3, it can be concluded that the possible AR(p) values are 1,2 since those are the identified significant lags. Thus, another two possible models would be **Model 2: ARIMA(1,1,0)** and **Model 3: ARIMA(2,1,0).**

Table 3: AIC values for manually fitted ARIMA Models with only AR(p) and I(d)

	ARIMA(0,1,0)	**ARIMA(1,1,0)**	**ARIMA(2,1,0)**
AIC	4935.96	4927.39	4879.61

Figure 3: Residual analysis for manually fitted ARIMA Models with only AR(p) and I(d)

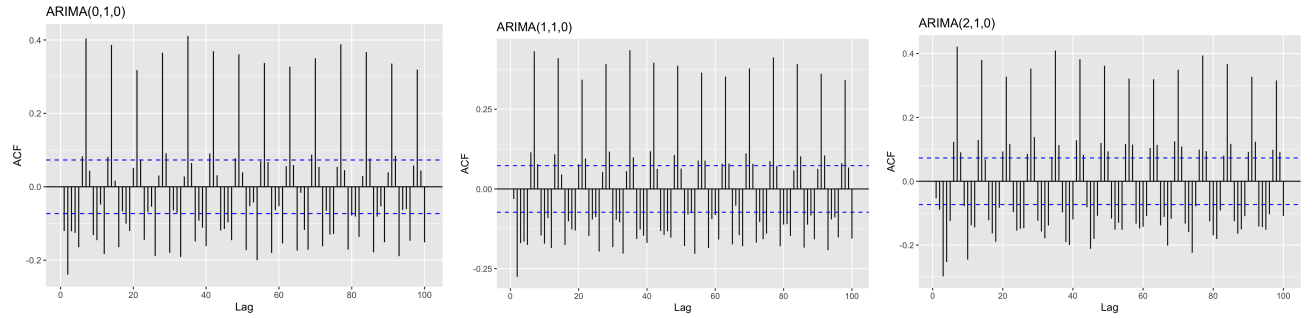


Table 4: Forecasting Error values for manually fitted ARIMA Models with only AR(p) and I(d)

	ARIMA(0,1,0)	**ARIMA(1,1,0)**	**ARIMA(2,1,0)**
MAE	10.749	9.586	9.599
RMSE	13.062	11.730	11.886

From figure 4, it is evident that there are significant lags in the ACF plots for all three models, hence AIC and forecasting error values against the test set are compared to determine the best model. Since AIC values are lowest for the ARIMA(2,1,0) model, it is the best fit among the three. Although the forecasting error values for the ARIMA(1,1,0) model are slightly lower, the differences relative to the ARIMA(2,1,0) model are minimal. Therefore, the **ARIMA(2,1,0)** model is selected as the best fit. Now that the AR(p) term has been determined, the next step is to model the MA(q) terms. Based on the ACF graph in figure 3, it can be said that the possible MA(q) values are 1,2 since those are identified as significant lags. Thus, using the best fit, ARI(2,1,0) model and possible MA(q) values, it is determined that the feasible ARIMA models are: **Model 4: ARIMA(2,1,1)** and **Model 5: ARIMA(2,1,2).**

Table 5: AIC values for manually fitted ARIMA Models with all components

	ARIMA(2,1,1)	**ARIMA(2,1,2)**
AIC	4659.89	4652.51

Figure 4: Residual analysis for manually fitted ARIMA Models with all components

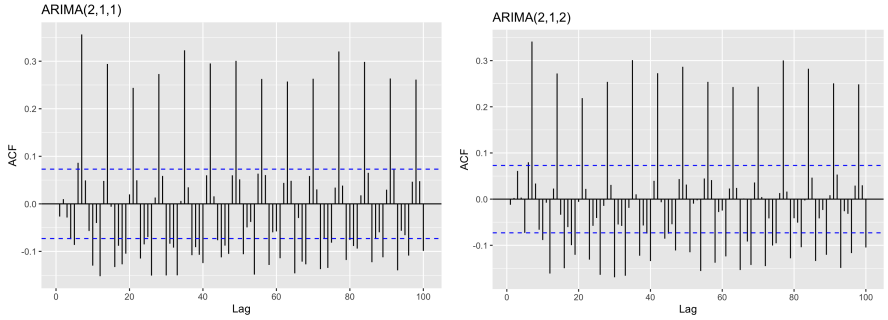


Table 6: Forecasting Error values for manually fitted ARIMA Models with all components

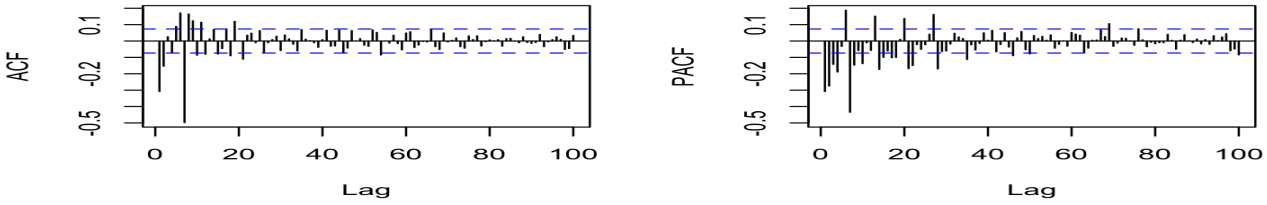
	ARIMA(2,1,1)	ARIMA(2,1,2)
MAE	6.650	6.537
RMSE	8.703	8.546

From figure 5 it is observed that all the ACF plots have significant lags hence, the analysis for the best model is conducted using AIC and forecasting error values against the test set. It is evident from table 5 and 6 that the model with the lowest AIC and error values is **ARIMA(2,1,2)** and hence it is concluded to be the best fit between the two.
Thus, the **ARIMA(2,1,2)** model is the best fit for the non-seasonal components.

Iterative Model Creation for Seasonal Components with Analysis

From figure 3, ACF graph it is observed that every 7th lag is highly significant which indicates weekly seasonality. Thus, seasonal differencing is conducted on the first order differenced time series which gives **Model 7: ARIMA(2,1,2)(0,1,0).**

Figure 5: ACF and PACF graphs for Seasonal Differenced Time Series



Upon evaluating the PACF graphs from figure 6, it can be concluded that the possible SAR(P) values are 1,2 and 3 since those are the identified significant lags. Thus, two possible models would be **Model 7: ARIMA(2,1,2)(1,1,0), Model 3: ARIMA(2,1,2)(2,1,0)** and **Model 3: ARIMA(2,1,2)(3,1,0).**

Table 7: AIC values for manually fitted SARIMA Models with only SAR(P) and SI(D)

	ARIMA(2,1,2)(0,1,0)	ARIMA(2,1,2)(1,1,0)	ARIMA(2,1,2)(2,1,0)	ARIMA(2,1,2)(3,1,0)
AIC	4746.49	4568.81	4528.14	4473

Figure 6: Residual analysis for manually fitted SARIMA Models with only SAR(P) and SI(D)

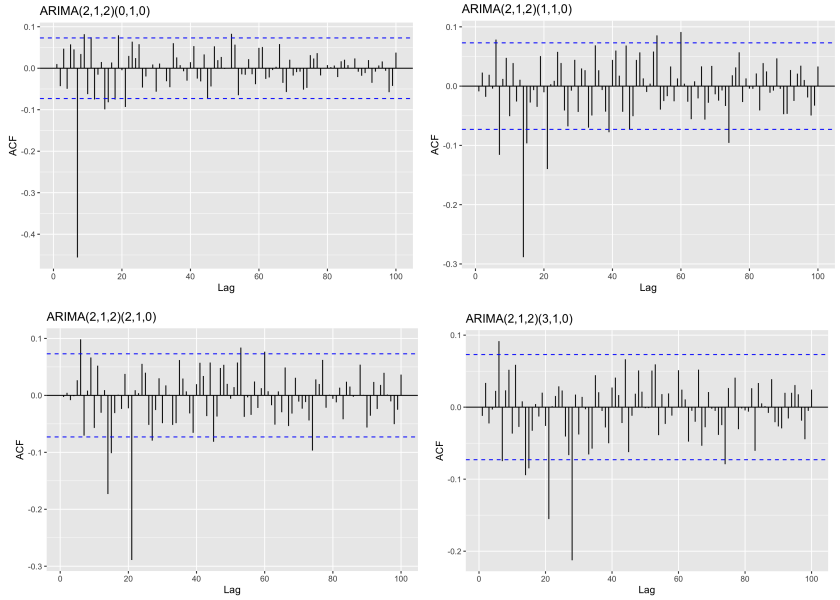


Table 8: Forecasting Error values for manually fitted ARIMA Models with only SAR(P) and SI(D)

	ARIMA(2,1,2)(0,1,0)	ARIMA(2,1,2)(1,1,0)	ARIMA(2,1,2)(2,1,0)	ARIMA(2,1,2)(3,1,0)
MAE	7.023	7.148	6.303	5.423
RMSE	10.009	10.685	9.898	9.343

From figure 7, it is clear that there are significant lags in the ACF plots for all four models, hence AIC and forecasting error values against the test set are compared to determine the best model. Since AIC and error values are lowest for the ARIMA(2,1,2)(3,1,0) model, it is the best fit.

Now that the SAR(P) term has been determined, the next step is to model the SMA(Q) terms. Based on the ACF graph in figure 6, it can be said that the possible SMA(Q) values are 1,2 since those are identified as significant lags. Thus, using the best fit, ARIMA(2,1,2)(3,1,0) model and possible SMA(Q) values, it is determined that the feasible ARIMA models are: **Model 4: ARIMA(2,1,2)(3,1,1)** and **Model 5: ARIMA(2,1,2)(3,1,2).**

Table 9: AIC values for manually fitted SARIMA Models with all seasonal components

	ARIMA(2,1,2)(3,1,1)	ARIMA(2,1,2)(3,1,2)
AIC	4336.95	4338.8

Figure 7: Residual analysis for manually fitted SARIMA Models with all seasonal components

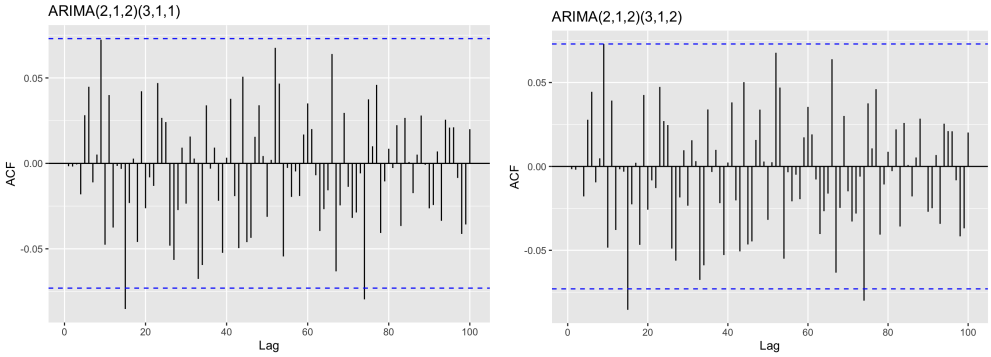


Table 10: Forecasting Error values for manually fitted ARIMA Models with all seasonal components

	ARIMA(2,1,2)(3,1,1)	ARIMA(2,1,2)(3,1,2)
MAE	4.834	4.843
RMSE	7.547	7.536

From figure 8, it is clear there are acceptable numbers of significant lags in the ACF plots for both models considering 95% confidence hence AIC and forecasting error values against the test set are compared to determine the best model. Since AIC and error values are lowest for the **ARIMA(2,1,2)(3,1,1)** model, it is the best fit.

Best Fit Manual ARIMA Model vs Automated ARIMA Model

Table 11: AIC values for Best Fit Manual Model and Automated Model

	ARIMA(2,1,2)(3,1,1)	Auto ARIMA(2,1,2)
AIC	4336.95	4652.51

Figure 8: Residual analysis for Best Fit Manual Model and Automated Model

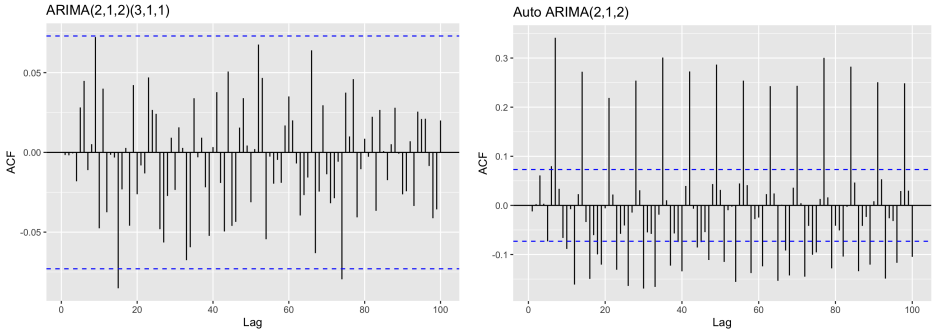


Table 12: Forecasting Error values for Best Fit Manual Model and Automated Model

	ARIMA(2,1,2)(3,1,1)	Auto ARIMA(2,1,2)
MAE	4.834	6.537
RMSE	7.547	8.546

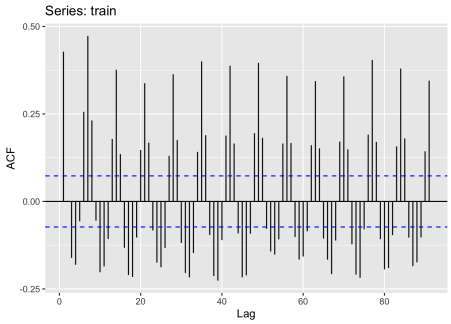
From figure 9, it is clear that the auto-ARIMA model has significant lags in the ACF plots whereas the best fit manual ARIMA model contains 95% confidence worth of lags in the interval. In addition, AIC and error values are lower for the best fit manual ARIMA model, hence overall ARIMA(2,1,2)(3,1,0) model, it is the best fit.

Conclusion

Thus, based on all the analysis it is concluded that ARIMA(2,1,2)(3,1,1) is the best fitted manual model. In comparison to the auto-ARIMA model as well, the best fit manual model of ARIMA(2,1,2)(3,1,1) is the better model. Thus, overall **ARIMA(2,1,2)(3,1,1**) is the best fit.

Regression
Data Preparation and Preliminary Analysis

Figure 9: ACF Graph for Train Set



From preliminary analysis, it was concluded that the time series has a trend and weekly seasonality. Further, based on figure 10, lag 1 and lag 7 are the most evident. Thereby, all of the parameters are included as independent variables during regression analysis.

train_vector - train data set vector used to evaluate forecasting performance
trend - sequential number capturing increasing changes of the time series
DayOfWeek - categorical variable showing each day of the week due to weekly seasonality
lag1 - previous day's value which captures time series dependency
lag7 - previous week's value which captures weekly seasonality

From the spread plot and correlation coefficients ***(Refer to appendix - 5,6)***, it is evident that the train time series is correlated to all of the parameters. DayOfWeek, accounting for weekly seasonality as seasonal dummies, is highly correlated to the train set followed by lag7 and lag1 whereas trend has a weak correlation to it.

Automated Regression Model

The automated regression model recommended by the system is :

train_vector=β0+β1(trend)+β2(DayOfWeek)+β3(lag1)+β4(lag7)+ϵ

Iterative Model Creation

Since "thurs", "lag1" and "lag7" have the highest correlation with the dependent variable, the first model created is **Model 1: train_vector=β0+β1*thurs+β2*lag1+β3*lag7+ϵ .** However for this model, the RSE is high and adjusted R^2 is low hence we try to add more parameters to the model.
For the next model, we add in "mon","tue","fri","sat","sun" and "trend" since they have a correlation with the dependent variable as well. Thus,
Model 2:
trainvector=β0+β1*thurs+β2*lag1+β3*lag7+β4*mon+β5*tue+β6*fri+β7*sat+β8*sun+β9*trend+ϵ
While the RSE decreases and adjusted R^2 significantly improves, it is observed that "lag7" and "thurs" are highly correlated leading to multicollinearity. Therefore, "lag7" is removed from the next model and following is the same **Model 3:**
train_vector=β0+β1*thurs+β2*lag1+β3*mon+β4*tue+β5*fri+β6*sat+β7*sun+β8*trend+ϵ . This step slightly increases the adjusted R^2 and lowers the RSE.

Manual Model Analysis

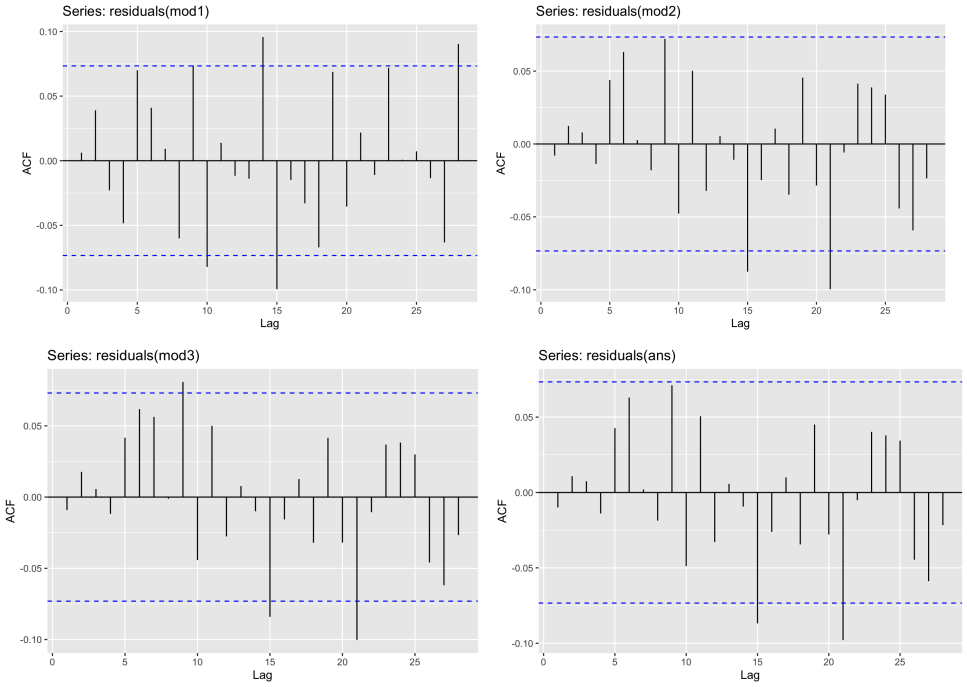
Table 13: Residual Standard Error for all models

	Model 1	Model 2	Model 3	Auto Regression
RSE	5.264	4.911	4.909	4.911
R^2 **Adjusted**	0.4332	0.5067	0.5074	0.5066

Table 14: Forecasting Error values for all Regression Models

	Model 1	Model 2	Model 3	Auto Regression
ME	5.094	3.31	3.071	3.292

Figure 10: ACF Graph for all Regression Models



From table 13 and 14 it is seen that the RSE, ME values are lowest and adjusted R^2 is highest for Model 3 out of all the manually built models. Further, from figure 11 it is seen that the number of significant lags in Model 2 and Model 3 are acceptable according to a 95% confidence interval. Thus, overall among manually fit models, Model 3 is the best fit.

On comparing best fit manual regression model, Model 3 with the automated model, it is seen figure 11 that both ACF graphs have the acceptable limit of number of significant lags in a 95% confidence interval, hence adjusted R^2 RSE and ME are compared. From table 13 and 14, it is concluded that the adjusted R^2 is higher and RSE, ME values are lower for the best fit manual model. Thus, between the automated model and manual best fit model, the Model 3, best fit manual model is the better choice.

AI and Machine Learning Models

Neural Network

The nntar() function is used to automatically generate the best fitting neural network model which in this case is **NNAR(28,1,15)[364]**

K-Nearest Neighbours

The knn_forecasting() function is used to automatically generate multiple k-nearest neighbour models using a different number of neighbours(k) which is iterated from 2 to 5.

Table 15: Forecasting Error values for K-Nearest Neighbour Models

	k=2	k=3	k=4	k=5
MAE	5.832	6.691	5.224	6.454
RMSE	9.099	8.908	7.251	8.565

From table 15 it is evident that when the number of neighbours is set to 4, the forecasting errors generated are the lowest as compared to the others. Hence, the best model k-nearest neighbours model is with k=4.

Comparative Analysis of Forecasting Models

For the purpose of comparing models across different forecasting modelling techniques, the sMAPE will be used since it is independent of scaling, no problem with the denominator being zero and avoids large biases.

The models that are being compared are Automated Exponential Smoothing, Manual Exponential Smoothing, Automated ARIMA, Manual ARIMA, Automated Regression, Manual Regression, K-Nearest Neighbours, Naive and Seasonal Naive. The last two models are used as a benchmark to compare other models.

The testing is conducted on In-Sample and Out-of-Sample data however, the values for out of sample are more relevant since the behaviour of the model on unseen data is more relevant to a forecasting model.

Table 16: sMAPE values across all models

		ANN	ANA	Auto ARIMA	Manual ARIMA	Auto - Regression	Manual Regression	KNN	Naive	Season Naive
sMAPE	**Test**	35.44	33.11	37.15	28.46	31.95	35.88	30.64	49.65	47.15
	Train	30.99	14.99	25.39	19.02	20.22	25.94	NaN	NaN	NaN

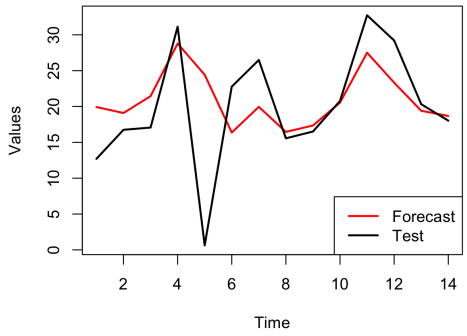
From table 16, it is evident that the manually fitted ARIMA(2,1,2)(3,1,1) is the best model with the lowest sMAPE value. Although, it is the second lowest when compared to the train test, the accuracy in forecasting is more important and hence, the out-of-sample sMAPE results are prioritised.

Since, Naive and Seasonal Naive are used as benchmarks, the difference between the sMAPE values for the test are very evident. Clearly, Naive and Seasonal naive have a higher error which indicates its not a great fit for the time series.Thereby, compared to the benchmark, ARIMA(2,1,2)(3,1,1) is indeed the best model.

Following is the forecasted values and graph for the best model **ARIMA(2,1,2)(3,1,1)**

14-day forecast: 9.92947, 19.09173, 21.44318, 28.78003, 24.42554, 16.36903, 19.95277, 16.45900, 17.34964, 20.54634, 27.50705, 23.33064, 19.40678, 18.67863

Figure 11: Forecasting fit for ARIMA(2,1,2)(3,1,1)



Conclusion

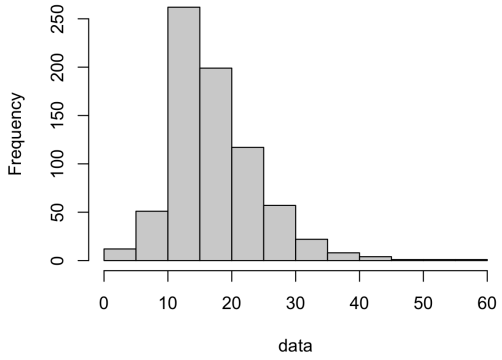
Multiple modelling methods were used throughout the report to find the best fitted model for the time series NN5-025. The exponential smoothing model concluded that the ANA model was the best fit as compared to the other models of ANN and AAA because of lower AIC and forecasting errors. Similarly for ARIMA modelling, multiple models were built and analysed to find the best fit model which was ARIMA(2,1,2)(3,1,1). This model also outperformed the automated model built by the system because of the better fitting ACF graphs. A few models were built using Regression as well and the best was the one including all days of the week except wednesday, lag1 and trend. This was built by iteratively adding in parameters and analysing the fit by adjusted R^2. Finally, the K-Nearest Neighbours model was built by using various numbers of neighbours but the one with 4 neighbours was the best fit. All of these models were then compared and it was concluded based on sMAPE that the manually built ARIMA(2,1,2)(3,1,1) model is the best fit. This model was also compared with the naive models to gain an understanding with regards to the benchmark and it was concluded that the model of best fit is far better performing that the naive models.
Thus, it is recommended that the ARIMA(2,1,2)(3,1,1) model be used for forecasting for better results and accuracy.

List of Graphs

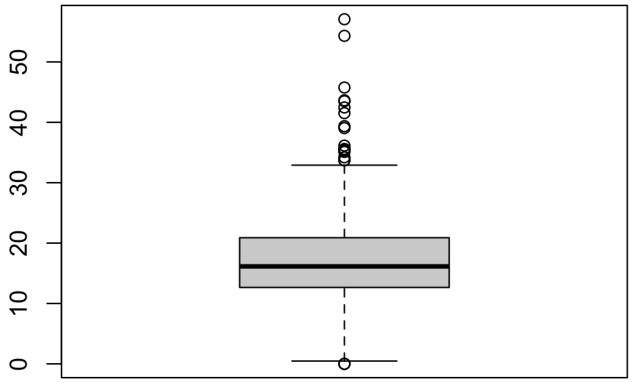
List of Tables

Appendix

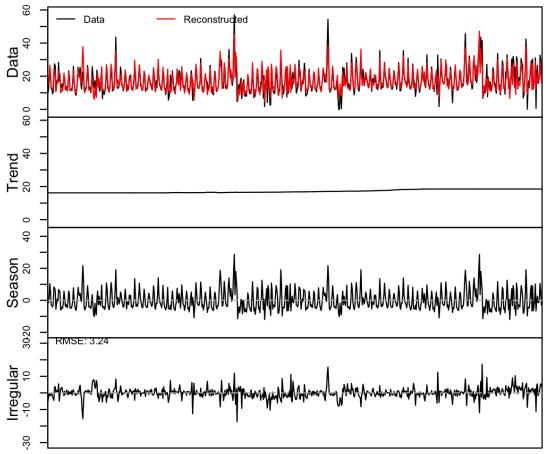
1. **Histogram to show skewness**



2. Boxplot to show outliers



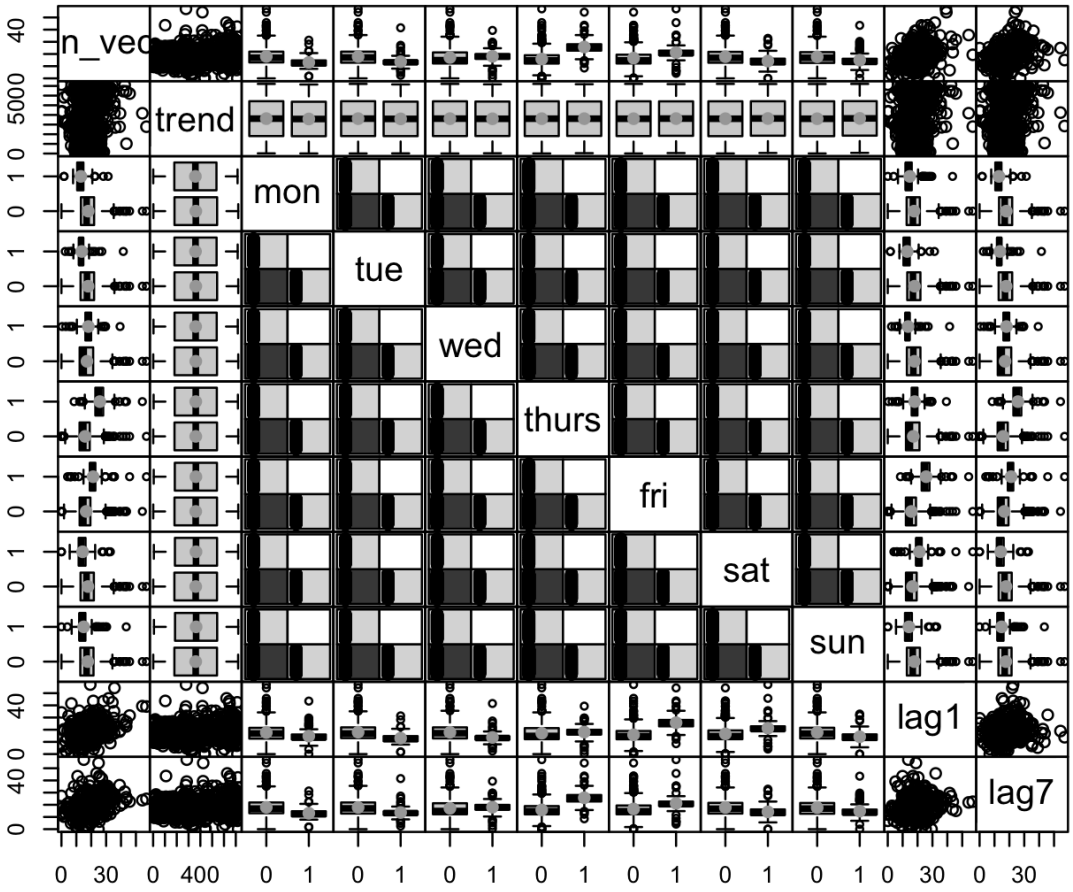
3. Additive Decomposition of Time Series



4. Test for presence of trend

	P value	Tau	Conclusion
Mann Kendall of Train	5.4836e-06	0.112	Presence of upward trend line
KPSS of Train	0.01	-	Non-stationarity of time series
KPSS after 1st Order Differencing	0.1	-	Stationarity of time series

5. Spread Matrix for Parameters of Regression Model



6. Correlation matrix for parameters of Regression Model

values:

	train_vector	trend	mon	tue	wed	thurs	fri	sat	sun	lag1	lag7
train_vector	1.0000	0.1640	0.2472	0.2118	0.0474	0.4964	0.2324	0.1798	0.1374	0.4292	0.4773
trend	0.1640	1.0000	0.0059	0.0039	0.0020	0.0000	0.0020	0.0039	0.0059	0.1602	0.1570
mon	0.2472	0.0059	1.0000	0.1567	0.1567	0.1567	0.1567	0.1567	0.1567	0.1447	0.2474
tue	0.2118	0.0039	0.1567	1.0000	0.1567	0.1567	0.1567	0.1567	0.1567	0.2467	0.2094
wed	0.0474	0.0020	0.1567	0.1567	1.0000	0.1567	0.1567	0.1567	0.1567	0.2113	0.0485
thurs	0.4964	0.0000	0.1567	0.1567	0.1567	1.0000	0.1567	0.1567	0.1567	0.0486	0.4995
fri	0.2324	0.0020	0.1567	0.1567	0.1567	0.1567	1.0000	0.1567	0.1567	0.4987	0.2304
sat	0.1798	0.0039	0.1567	0.1567	0.1567	0.1567	0.1567	1.0000	0.1567	0.2340	0.1777
sun	0.1374	0.0059	0.1567	0.1567	0.1567	0.1567	0.1567	0.1567	1.0000	0.1792	0.1439
lag1	0.4292	0.1602	0.1447	0.2467	0.2113	0.0486	0.4987	0.2340	0.1792	1.0000	0.2590
lag7	0.4773	0.1570	0.2474	0.2094	0.0485	0.4995	0.2304	0.1777	0.1439	0.2590	1.0000

```
#------------ Import Libraries ------------
library(forecast)
library(smooth)
library(tsutils)
library(readxl)
library(zoo)
library(e1071)
library(ggplot2)
library(tseries)
library(lmtest)
library(dplyr)
library(car)
library(caret)
library(tsfknn)

#------------ Import Dataset ------------
data_s=read_excel("Documents/MSc Business
Analytics/MSCI570_Forecasring and Predictive
Analytics/coursework/Individual/data_s.xlsx")

#------------ Interpolate Missing Values ------------
dataset=data_s$Values
data=na.approx(dataset)

#------------ Ordering data by date ------------
data_s$Dates=as.Date(data_s$Dates)
data_s=data_s[order(data_s$Dates), ]

#------------ Converting to Time Series ------------
data_series=ts(data, frequency=364, start = c(1996, 77))

#------------ Setting Forecasting Parameters ------------
h=14
data_length=length(data_series)

#------------ Create a Train set ------------
train=ts(data_series[1:(data_length - h)],
frequency=frequency(data_series),start=start(data_series))

#------------ Create a Test set ------------
test=data_series[(data_length - h + 1):data_length]

#------------ Exponential Smoothing: Model Creation ------------
```

```
#Automated Model
es_ZZZ=es(train, model="ZZZ")

#Model without Trend
es_ANA=es(train, model="ANA")

#Model with Trend
es_AAA=es(train, model="AAA")

#------------- Exponential Smoothing: Model Analysis ------------
#Automated Model
summary(es_ZZZ)

#Model without Trend
summary(es_ANA)

#Model with Trend
summary(es_AAA)

#------------- Exponential Smoothing: Residual Analysis ------------
#Automated Model
hist(residuals(es_ZZZ), xlab="Residuals of ANN Model")

#Model without Trend
hist(residuals(es_ANA), xlab="Residuals of ANA Model")

#Model with Trend
hist(residuals(es_AAA), xlab="Residuals of AAA Model")

#------------- Exponential Smoothing: Forecasting Analysis
------------
#Automated Model
forecast_ZZZ=forecast(es_ZZZ, 14)
accuracy(forecast_ZZZ$mean,test)

#Model without Trend
forecast_ANA=forecast(es_ANA, h=14)
accuracy(forecast_ANA$mean,test)

#Model with Trend
forecast_AAA=forecast(es_AAA, h=14)
accuracy(forecast_AAA$mean,test)
```

```
#------------ ARIMA: Auto Model ------------
#Fitting the auto model
auto_fit=auto.arima(train)
summary(auto_fit)

#Analysing fitted model residuals
ggAcf(residuals(auto_fit), lag=100, main="Auto ARIMA(2,1,2)")

#Forecasting using the fitted model
auto_forecast=forecast(auto_fit, h=14)

#Evaluating forecasted values accuracy against test values
accuracy(auto_forecast$mean, test)

#------------ ARIMA Data Analysis ------------
#Testing stationary of train set
kpss.test(train)

#First order differencing of train set
diff_train=diff(train)

#Testing stationary of train set after differencing
kpss.test(diff_train)

#ACF and PACF graphs for differenced train set to identify possible
p,q
tsdisplay(diff_train, lag=91)

#------------ ARIMA: Model1------------
#Fitting the ARIMA(0,1,0)
fit1=Arima(train, order=c(0,1,0))
summary(fit1)

#Analysing fitted model residuals
ggAcf(residuals(fit1), lag=100, main="ARIMA(0,1,0)")

#Forecasting using the fitted model
forecast1=forecast(fit1, h=14)

#Evaluating forecasted values accuracy against test values
accuracy(forecast1$mean,test)

#------------ ARIMA: Model2------------
```

```
#Fitting the ARIMA(1,1,0)
fit2=Arima(train, order=c(1,1,0))
summary(fit2)

#Analysing fitted model residuals
ggAcf(residuals(fit2), lag=100, main="ARIMA(1,1,0)")

#Forecasting using the fitted model
forecast2=forecast(fit2, h=14)

#Evaluating forecasted values accuracy against test values
accuracy(forecast2$mean, test)


#------------- ARIMA: Model3------------
#Fitting the ARIMA(2,1,0)
fit3=Arima(train, order=c(2,1,0))
summary(fit3)

#Analysing fitted model residuals
ggAcf(residuals(fit3), lag=100, main="ARIMA(2,1,0)")

#Forecasting using the fitted model
forecast3=forecast(fit3, h=14)

#Evaluating forecasted values accuracy against test values
accuracy(forecast3$mean, test)

#------------- ARIMA: Model4------------
#Fitting the ARIMA(2,1,1)
fit4=Arima(train, order=c(2,1,1))
summary(fit4)

#Analysing fitted model residuals
ggAcf(residuals(fit4), lag=100, main="ARIMA(2,1,1)")

#Forecasting using the fitted model
forecast4=forecast(fit4, h=14)

#Evaluating forecasted values accuracy against test values
accuracy(forecast4$mean, test)

#------------- ARIMA: Model5------------
```

```
#Fitting the ARIMA(2,1,2)
fit5=Arima(train, order=c(2,1,2))
summary(fit5)

#Analysing fitted model residuals
ggAcf(residuals(fit5), lag=100, main="ARIMA(2,1,2)")

#Forecasting using the fitted model
forecast5=forecast(fit5, h=14)

#Evaluating forecasted values accuracy against test values
accuracy(forecast5$mean, test)

#------------- Seasonal differencing------------
szn_data=diff(diff_train, lag=7)

#ACF and PACF graphs for differenced train set to identify possible
p,q
tsdisplay(szn_data, lag=91)

#------------- ARIMA: Model6------------
#Fitting the ARIMA(2,1,2)(0,1,0)
fit6=Arima(train, order=c(2,1,2), seasonal = list(order = c(0,1,0),
period=7))
summary(fit6)

#Analysing fitted model residuals
ggAcf(residuals(fit6), lag=100, main="ARIMA(2,1,2)(0,1,0)")

#Forecasting using the fitted model
forecast6=forecast(fit6, h=14)

#Evaluating forecasted values accuracy against test values
accuracy(forecast6$mean, test)

#------------- ARIMA: Model7------------
#Fitting the ARIMA(2,1,2)(1,1,0)
fit7=Arima(train, order=c(2,1,2), seasonal = list(order = c(1,1,0),
period=7))
summary(fit7)

#Analysing fitted model residuals
ggAcf(residuals(fit7), lag=100, main="ARIMA(2,1,2)(1,1,0)")
```

```
#Forecasting using the fitted model
forecast7=forecast(fit7, h=14)

#Evaluating forecasted values accuracy against test values
accuracy(forecast7$mean, test)

#------------- ARIMA: Model8------------
#Fitting the ARIMA(2,1,2)(2,1,0)
fit8=Arima(train, order=c(2,1,2), seasonal = list(order = c(2,1,0),
period=7))
summary(fit8)

#Analysing fitted model residuals
ggAcf(residuals(fit8), lag=100, main="ARIMA(2,1,2)(2,1,0)")

#Forecasting using the fitted model
forecast8=forecast(fit8, h=14)

#Evaluating forecasted values accuracy against test values
accuracy(forecast8$mean, test)

#------------- ARIMA: Model9------------
#Fitting the ARIMA(2,1,2)(3,1,0)
fit9=Arima(train, order=c(2,1,2), seasonal = list(order = c(3,1,0),
period=7))
summary(fit9)

#Analysing fitted model residuals
ggAcf(residuals(fit9), lag=100, main="ARIMA(2,1,2)(3,1,0)")

#Forecasting using the fitted model
forecast9=forecast(fit9, h=14)

#Evaluating forecasted values accuracy against test values
accuracy(forecast9$mean, test)

#------------- ARIMA: Model10------------
#Fitting the ARIMA(2,1,2)(3,1,1)
fit10=Arima(train, order=c(2,1,2), seasonal = list(order = c(3,1,1),
period=7))
summary(fit10)
```

```
#Analysing fitted model residuals
ggAcf(residuals(fit10), lag=100, main="ARIMA(2,1,2)(3,1,1)")

#Forecasting using the fitted model
forecast10=forecast(fit10, h=14)

#Evaluating forecasted values accuracy against test values
accuracy(forecast10$mean, test)

#------------- ARIMA: Model11------------
#Fitting the ARIMA(2,1,2)(3,1,2)
fit11=Arima(train, order=c(2,1,2), seasonal = list(order = c(3,1,2),
period=7))
summary(fit11)

#Analysing fitted model residuals
ggAcf(residuals(fit11), lag=100, main="ARIMA(2,1,2)(3,1,2)")

#Forecasting using the fitted model
forecast11=forecast(fit11, h=14)

#Evaluating forecasted values accuracy against test values
accuracy(forecast11$mean, test)


#------------- Regression: Train set------------
#ACF graph
ggAcf(train, lag=91)

#Train Data Preparation
train_vector=as.numeric(train)
df=data.frame(train_vector)
df$trend=c(1:length(train_vector))
df$mon=rep(c(1,0,0,0,0,0,0),103)
df$tue=rep(c(0,1,0,0,0,0,0),103)
df$wed=rep(c(0,0,1,0,0,0,0),103)
df$thurs=rep(c(0,0,0,1,0,0,0),103)
df$fri=rep(c(0,0,0,0,1,0,0),103)
df$sat=rep(c(0,0,0,0,0,1,0),103)
df$sun=rep(c(0,0,0,0,0,0,1),103)
df$lag1=lag(df$train_vector, 1)
df$lag7=lag(df$train_vector, 7)
```

```
#------------- Regression: Test set------------
#Test Data Preparation
test_vector=as.numeric(test)
df_test=data.frame(test_vector)
df_test$trend=c(1:length(test_vector))
df_test$mon=rep(c(1,0,0,0,0,0,0),2)
df_test$tue=rep(c(0,1,0,0,0,0,0),2)
df_test$wed=rep(c(0,0,1,0,0,0,0),2)
df_test$thurs=rep(c(0,0,0,1,0,0,0),2)
df_test$fri=rep(c(0,0,0,0,1,0,0),2)
df_test$sat=rep(c(0,0,0,0,0,1,0),2)
df_test$sun=rep(c(0,0,0,0,0,0,1),2)
df_test$lag1=lag(df_test$test_vector, 1)
df_test$lag7=lag(df_test$test_vector, 7)

#------------- Regression: Variable correlations------------
association(df)

#----------Auto Regression-----------
#Model with no parameter
auto_null=lm(train_vector~1, data=df)

#Model with all parameters
auto_full=lm(train_vector~., data=df)

#Iterate in both directions to get the best fit
interate=step(auto_full, direction="both")

#Automated Regression fit
ans=lm(formula = train_vector ~ trend + lag1 + lag7 + mon + wed +
thurs + fri +
         sat, data = df)
summary(ans)

#Analysing fitted model residuals
ggAcf(residuals(ans))

#Predict values based on test set
auto_predict_test=predict(ans, df_test)

#Errors of the forecasted values
accuracy(auto_predict_test,df_test$test_vector)
```

```
#------------- Regression: Model1------------
#Fitting the train_vector=β0+β1*thurs+β2*lag1+β3*lag7+ϵ
mod1=lm(train_vector~thurs+lag1+lag7, data=df)
summary(mod1)

#Analysing fitted model residuals
ggAcf(residuals(mod1))

#Forecasting using the fitted model
mod1_predict=predict(mod1, df_test)

#Evaluating forecasted values accuracy against test values
accuracy(mod1_predict,df_test$test_vector)

#------------- Regression: Model 2------------
#Fitting the
train_vector=β0+β1*thurs+β2*lag1+β3*lag7+β4*mon+β5*tue+β6*fri+β7*sat+
β8*sun+β9*trend+ϵ
mod2=lm(train_vector~thurs+lag1+lag7+mon+tue+fri+sat+sun+trend,
data=df)
summary(mod2)

#Analysing fitted model residuals
ggAcf(residuals(mod2))

#Forecasting using the fitted model
mod2_predict=predict(mod2, df_test)

#Evaluating forecasted values accuracy against test values
accuracy(mod2_predict,df_test$test_vector)

#------------- Regression: Model 3------------
#Fitting the
train_vector=β0+β1*thurs+β2*lag1+β3*mon+β4*tue+β5*fri+β6*sat+β7*sun+β
8*trend+ϵ
mod3=lm(train_vector~thurs+lag1+mon+tue+fri+sat+sun+trend, data=df)
summary(mod3)

#Analysing fitted model residuals
ggAcf(residuals(mod3))

#Forecamod1#Forecastmod1#Forecasting using the fitted model
mod3_predict=predict(mod3, df_test)
```

```
#Evaluating forecasted values accuracy against test values
accuracy(mod3_predict,df_test$test_vector)

#------------- K-Nearest Neighbors------------
#Fitting the k=2 model
knn_fit1=knn_forecasting(train, h=14, lags=1:7, k=2, msas="MIMO")
#Evaluating forecasted values accuracy against test values
accuracy(knn_fit1$prediction,test)

#Fitting the k=3 model
knn_fit2=knn_forecasting(train, h=14, lags=1:7, k=3, msas="MIMO")
#Evaluating forecasted values accuracy against test values
accuracy(knn_fit2$prediction,test)

#Fitting the k=4 model
knn_fit3=knn_forecasting(train, h=14, lags=1:7, k=4, msas="MIMO")
#Evaluating forecasted values accuracy against test values
accuracy(knn_fit3$prediction,test)

#Fitting the k=5 model
knn_fit4=knn_forecasting(train, h=14, lags=1:7, k=5, msas="MIMO")
#Evaluating forecasted values accuracy against test values
accuracy(knn_fit4$prediction,test)

#------------- Naive ------------
#Fitting the Naive model
naive=naive(train, h=14)

#Evaluating forecasted values accuracy against test values
accuracy(naive$mean,test)

#------------- Seasonal Naive ------------
#Fitting the Seasonal Naive model
snaive=snaive(train,h=14)

#Evaluating forecasted values accuracy against test values
accuracy(snaive$mean,test)

#------------- Best Fit Model Forecast ------------
forecast10$mean
plot(as.numeric(forecast10$mean), col = "red", lwd = 2, type = "l",
```

```
     main="Forecasting fit for the ARIMA(2,1,2)(3,1,1)",xlab =
"Time", ylab = "Values", ylim = range(c(forecast_ZZZ$mean, test)))
lines(test, col = "black", lwd = 2)
legend("bottomright", legend = c("Forecast", "Test"), col = c("red",
"black"), lwd = 2)

#------------- Appendix ------------
hist(data, main="Histogram to show skewness")
boxplot(data, main="Boxplot to see outliers")
decomp(data_series, decomposition="additive", outplot=TRUE)
Kendall::MannKendall(train)
kpss.test(train)
spread(df)
association(df)
```