Configuration Manual

MSc Research Project
Data Analytics

Saptarshi Das
Student ID: x18127355

School of Computing
National College of Ireland

Supervisor: Dr Muhammad Iqbal

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Saptarshi Das
Student ID:	x18127355
Programme:	Data Analytics
Year:	2019
Module:	MSc Research Project
Supervisor:	Dr Muhammad Iqbal
Submission Due Date:	12/12/2019
Project Title:	Configuration Manual
Word Count:	XXX
Page Count:	12

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	12th December 2019

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Saptarshi Das
x18127355

1 Introduction

This report will describe in details all the turns taken to successfully complete the project. Along with the procedure to achieve the goals, the software and hardware platforms configurations will also be laid on paper for getting the similar results every time.

2 System Configuration

2.1 Software Setup

- Microsoft Excel 2019
- RStudio IDE v1.2.5019
- R v3.6.1(64bit)
- Tableau 2018.2

The above above mentioned softwares were used of the specified verision. In RStudio certain packages were used to manipulate the acquired data and implement the forecasting models. A brief summery of the packages along with their purpose in the project has been discussed below.
R Packages Utilised

- ggpubr- Facilitates in producing advanced graphics, the normality test by Q-Q plot was done with the help of this package. ggplot2 package is a required package for this package
- CombMSC- This package provides a convenient functionality of splitting a dataset intro train and test subparts. The splitTrainTest() function was taken from this package.
- prophet- Utilised to implement the prophet model.
- forecast- Holds a range of timeseries analysis facilities. The forecast function along with several models like auto.arima, nnetar, ses are packed in it. To check how good the model fit, residuals were checked by checkresidual() function which also belongs to this package.
- tseries- Besides having several time series analysis facilities, in this project it is mainly utilised for its test of Dickey-Fuller.

- zoo- It is generally used to handle the irregulaties of time series data,and in this project, it helped in converting the data into a zoo object, which is a necessity for aggregate function used to convert the hourly data to daily records.
- ggplot2- This package gets automatically loaded while loading ggpubr and helped in producing various visualizations. In this current project it helped in manipulating the titles and names of both axis.
- xlsx- This pakage was solely used to import the data file into the RStudio, as it was in .xlsx format.
- tidyr- This function helped a lot in spliting the datatime column and extract only the relevant date information.
- dplyr- At certain point to implement a function to multiple columns, select function is most benefiting. This package lets use of the select function.
- Metric- As the errors of the Prophet model are not similar to the ones of ARIMA model, so to measure the accuracy of the Prophet model, separately error measuring functions were used which are available under the Metric package.

2.2 Hardware Setup

From the hardware perspective, all the previously specified softwares were executed on Windows 10 machine with 8 GB RAM and 1 TB of hard drive and everything power by an i5 6th generation chipset.

3 Project Development

In order to structure the complete project, all the tools specified above were used time and again migrating the data forward and backward from the RStudio IDE.

3.1 Data Preparation

The raw data taken from Central Pollution Control Board(CPCB)[1] was initially carrying 5 columns and 33961 rows. For aim of this project which is to conduct a univariate forecasting o wind speed, the columns carrying irrelevant parameters were dropped in the excel sheet only. A sample of the raw data can be seen in Figure 1. For conducting the research, the data recording starting time was not required as well, so "From Date" was also deleted.

[1] `https://app.cpcbccr.com/ccr/#/caaqm-dashboard-all/caaqm-landing/data/%7B%22state%22:%22Rajasthan%22,%22city%22:%22Jaipur%22,%22station%22:%22site_134%22%7D`

CENTRAL POLLUTION CONTROL BOARD						
CONTINUOUS AMBIENT AIR QUALITY						
						Date: Thursday, Dec 12 2019
						Time: 03:19:37 AM
State	Tamil Nadu					
City	Chennai					
Station	Alandur Bus Depot, Chennai - CPCB					
Parameter	BP,WD,Temp,WS,RH,Ozone,CO					
AvgPeriod	1 Hours					
From	01-01-2016T00:00:00Z 00:00					
To	16-11-2017T00:00:59Z 00:00					
Alandur Bus Depot, Chennai - CPCB						
Prescribed Standards		NA	NA	NA	NA	NA
Exceeding Standards		NA	NA	NA	NA	NA
Remarks						
From Date	To Date	BP	WD	Temp	WS	RH
01-01-2016 00:00	01-01-2016 01:00	1001.83	146.24	27.63	1.05	69.37
01-01-2016 01:00	01-01-2016 02:00	1009.01	173.56	27.38	0.98	68.98
01-01-2016 02:00	01-01-2016 03:00	1023.94	217.57	26.88	0.93	68.22
01-01-2016 03:00	01-01-2016 04:00	1037.8	278.77	26.61	0.9	67.79
01-01-2016 04:00	01-01-2016 05:00	1062.24	316.79	26.08	0.88	66.99
01-01-2016 05:00	01-01-2016 06:00	1061.44	308.99	26.14	0.9	67.08
01-01-2016 06:00	01-01-2016 07:00	1060.96	302.32	26.21	0.91	67.19
01-01-2016 07:00	01-01-2016 08:00	1067.46	299.61	26.44	0.91	67.55
01-01-2016 08:00	01-01-2016 09:00	1036.25	209.96	27.44	1.05	69.07
01-01-2016 09:00	01-01-2016 10:00	982.22	131.15	28.57	1.3	70.81
01-01-2016 10:00	01-01-2016 11:00	979.6	135.87	28.77	1.32	71.12
01-01-2016 11:00	01-01-2016 12:00	971.51	139.48	29.14	1.41	71.69

Figure 1: Raw Data

Now the data is imported to RStudio and the preprocessing happens as specified in the report. In Figure 2 the preprocessing of the data along with converting it to time series data by R coding can be seen. For the specific requirement of Prophet model, to have a temporal data with specific column names, a seperate copy of the data was kept aside and the process can be also seen at the end of Figure 2.

```
####################################################################################################
############################################     Maharashtra      ##################################
####################################################################################################
####################################################################################################

#Loading wind speed data of Maharashtra
MH <- read.xlsx("Maharashtra_Chandrapur.xlsx", 1)
#split column
MH = separate(MH, col = To.Date, into = c("Date"), sep = 10, remove=T)
#converting to POSIXlt date format to make it interpretable for zoo and other time series models
MH$Date=strptime(MH$Date, format= "%d-%m-%Y")
MH$WS = as.character(MH$WS)
MH$WS = as.numeric(MH$WS)

temp1= zoo(MH %>% select(2), order.by = MH$Date)
#replacing the outliers and null values with locally smoothed values
temp1$WS <- tsclean(temp1$WS) #forecast
#agregating the hourly data to daily format in oder to make mid-term predictions successfully
MH_ag= aggregate(temp1, as.Date, mean)
MH_ag[,1]=round(MH_ag[,1],digits = 2)

MH.ts <- ts(MH_ag, start= c(2016, 01, 01),  frequency=365)

MH.df <- as.data.frame(MH.ts)
names(MH.df)[1] <- "y"
write.table(MH.df, "C:/Users/HP/Downloads/Research Project/Data/MH_all.csv",
            sep=" ,", row.names=FALSE)

```

Figure 2: Data Preprocessing

Once the preprocessed data is ready, the data is analyzed by drawing several plots and the components like seasonality and trend are checked. Further on several tests are done to check if the data is at all containing knowledge or just a random noise. The stationarity of data was tested using Dickey Fuller test and Normality by Shapiro Wilk test. The codes can be seen in Figure 3.

```
autoplot(decompose(MH.ts))
autoplot(MH.ts) +
  ggtitle("Wind Speed data of Maharashtra(2016-2019)") +
  xlab("Year") +
  ylab("Wind Speed(m/sec)")

#Dickey-Fuller test for Stationarity
apply(MH.ts, 2, adf.test)

#From the pvalue, it can be seen that the data is not stationary


#Ljung-Box Test for white noise
Box.test(MH.ts, lag = 24, fitdf = 0, type = "Ljung")
# The pvalue very less than 0.05 suggests that the data is not white noise
#spliting 75% of the original data for training the models
Maharashtra<-splitTrainTest(MH.ts, numTrain = length(MH.ts) - 354)
#checking normality
ggqqplot(MH.df$y)+
  ggtitle("Maharashtra Data Linearity")
#Shapiro-Wilk normality test
shapiro.test(MH.df$y)
##though from the qqplot the data seems a bit non-linear, but from the
#p value <0.05 of Shapiro-Wilk normality test, it can be said that the data is normally distributed
#As the data is normally distributed, BoxCox transformation is not required
```

Figure 3: Data Analyzing

3.2 Model Fitting

A total of five models were applied to the ready dataset. The subsequent sections will show the functions used and codings done in order to make the project a success,

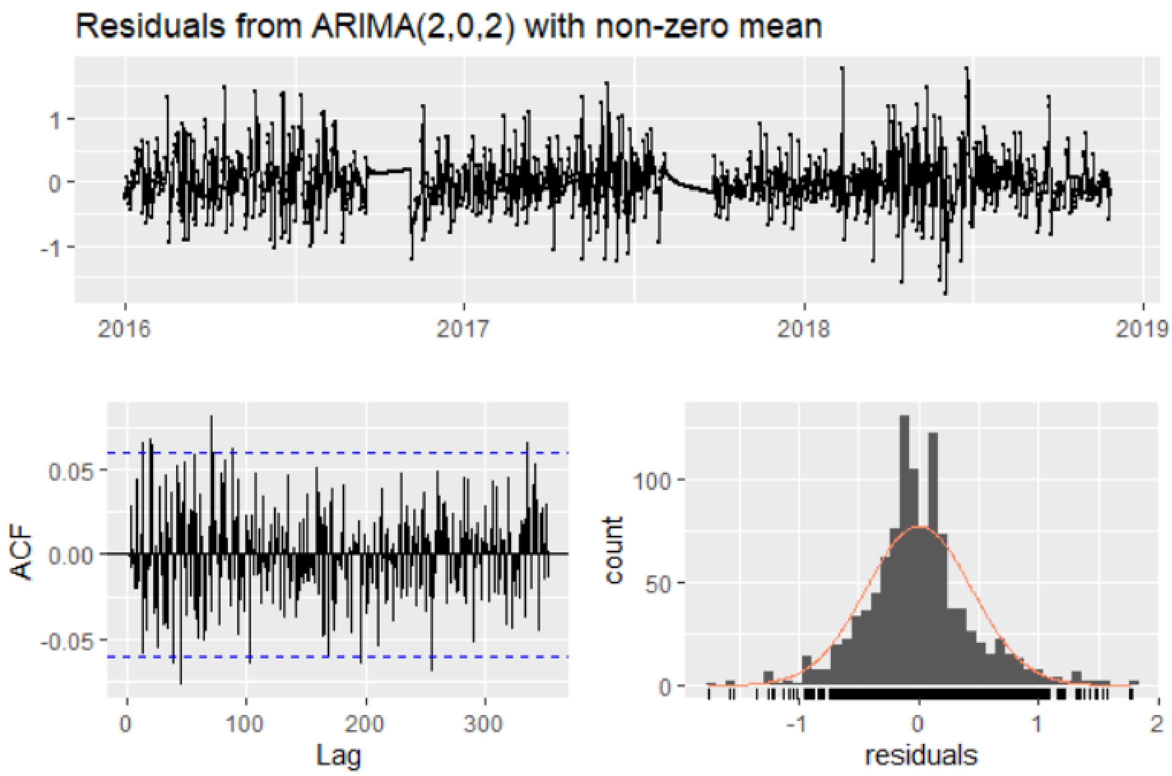
3.2.1 ARIMA Fitting

Below is the code of how the ARIMA model was implemented and also how the accuracy was checked is shown in Figure 4

```
############################ ARIMA ######################################

MHarima <-auto.arima(Maharashtra$train,D=0,lambda=NULL, stationary = T,
                      stepwise=F, trace=F, approximation=F, biasadj=F)
checkresiduals(MHarima)

#by looking at the residuals, it can be seen that it is white noise, so the model is a fit
MHarima.forecast <-forecast(MHarima,h=354)
autoplot(MHarima.forecast) + ylab("Wind Speed(m/sec)")

MHarima.acc<-accuracy(MHarima.forecast)

```

Figure 4: Coding for ARIMA

Also how suited is the model chosen by auto.arima is was checked by checking the residuals which is shown in Figure 5




```
> checkresiduals(MHarima)

        Ljung-Box test

data:  Residuals from ARIMA(2,0,2) with non-zero mean
Q* = 222.58, df = 207, p-value = 0.2177

Model df: 5.   Total lags used: 212
```

Figure 5: Cheching the Residuals

3.2.2 Neural Network Fitting

The fitted neural network is shown in Figure 6 and the process of checking the accuracy of the model is the same as ARIMA.

```
####################### Neural Net ####################################################
set.seed(18127355)
MHnn=nnetar(Maharashtra$train,p=41,P=2,size =19,repeats=50,
            lambda = NULL, scale.inputs=F)

MHnn.forecast=forecast(MHnn, h=354)

MHnn.acc<-accuracy(MHnn.forecast)

autoplot(MHnn.forecast) + ylab("Wind Speed(m/sec)")
```

Figure 6: Coding for Neural Netwrok

3.2.3 Simple Exponential Smoothing Model Fitting

The implemented SES model is shown in Figure 7 and can be seen that in case of SES the forecast function is not required to make future predictions.

```
#################### Simple Exponential Smoothening ############################################

MHses <- ses(Maharashtra$train, h = 354, lambda = NULL, initial="optimal", biasadj=F)

# checking the error rates to evaluate the model

MHses.acc<-accuracy(MHses)

# Add the one-step forecasts for the training data to the plot
autoplot(Maharashtra$train) + autolayer(fitted(MHses), series = "ses") + ylab("Wind Speed(m/sec)")
```

Figure 7: Coding for Simple Exponential Smoothing

3.2.4 Prophet Model Fitting

As it can be seen that for Prophet model, the way to calculate the accuracy measures are different. To produce the forecast, this model requires a blank temporal column can be seen in the code to be created, before making the predictions.

```
########################## Prophet Model ##########################################
library(Metrics)
MH.temporal <- read.csv("MH_all.csv")

MH.prophet<-prophet(MH.temporal[1:1062,], changepoints = NULL,
                    seasonality.mode = 'additive', daily.seasonality=F,fit = T)
MHfuture <- make_future_dataframe(MH.prophet, periods = 354)
MH.prophetforecast <- predict(MH.prophet, MHfuture, type="response")

dyplot.prophet(MH.prophet, MH.prophetforecast)

####### Errors
MHactual<- MH.temporal[1063:1416,2]

MHprophet.acc<-data.frame(rmse=rmse(MHactual, MH.prophetforecast$yhat),
                          mae=mae(MHactual, MH.prophetforecast$yhat))
detach("package:Metrics", unload = TRUE)
```

Figure 8: Coding for Prophet

3.2.5 Dynamic Harmonic Regression Model Fitting

For the Dynamic Harmonic Regression model, it can be see that the k value for the two datasets were chosen to be different. This is because of the fact that the AICc value of the datasets stabilized at different values of k. The first chunk of code in Figure 9 is for data of Maharashtra and below one is for Tamil Nadu.

```
###################Dynamic Harmonic Regression##################

MHdhr<- auto.arima(Maharashtra$train, xreg= fourier(Maharashtra$train, K=1),
                   lambda = 0, stationary = T,stepwise=F, trace=F, approximation=F, biasadj=T)
summary(MHdhr) # checking for the minimum AICc value to determine the value of K
MHdhr.forecast<-forecast(MHdhr, xreg= fourier(MH.ts, K=1, h=354))
MHdhr.acc<-accuracy(MHdhr.forecast)
checkresiduals(MHdhr)
autoplot(MHdhr.forecast) + ylab("Wind Speed(m/sec)")
```

```
###################Dynamic Harmonic Regression##################

TNdhr<- auto.arima(Tamil$train, xreg= fourier(Tamil$train, K=3), lambda = 0,
                   stepwise=F, trace=F, approximation=F, biasadj=F)

summary(TNdhr) # checking the AICc value from summary of the fit model, the value of K is fixed with t
TNdhr.forecast<-forecast(TNdhr, xreg= fourier(TN.ts, K=3, h=354))
TNdhr.acc<-accuracy(TNdhr.forecast)

autoplot(TNdhr.forecast) + ylab("Wind Speed(m/sec)")
```

Figure 9: Coding for Dynamic Harmonic Regression

3.3 Calculation of Accuracy

The accuracy values of all the models were put together into a single table for comparison purpose the code can be seen in Figure 10.

```
###########################################################################################################
############### Accuracy Table #####################
MH.evaluation_table <- data.frame("Models" = c("ARIMA","NN", "SES", "PROPHET", "DHR"), "RMSE" =0, "MAE" =0)

MH.evaluation_table[1,2:3] <- MHarima.acc[1,c(2,3)]
MH.evaluation_table[2,2:3] <- MHnn.acc[1,c(2,3)]
MH.evaluation_table[3,2:3] <- MHses.acc[1,c(2,3)]
MH.evaluation_table[4,2:3] <- MHprophet.acc[1,1:2]
MH.evaluation_table[5,2:3] <- MHdhr.acc[1,c(2,3)]
write.table(MH.evaluation_table, "C:/Users/HP/Downloads/Research Project/Data/MH_accuracy.csv",
          sep=" ,", row.names=FALSE)
```

Figure 10: Accuracy Table

3.4 Wind Power Calculation

At the end, with the results from both the states data, the forecasted wind speed data is used in the wind power generation equation shown in Figure 11. The calculated wind power is then exported out of RStudio to analyze the results.

```
#########Theoritically Calculating Wind Power with probable parameter values################################

###########################################################################################################
#WP=0.5*Cp*q*A*V^3
#[Cp=Max power coefficient(theoretical max-> 0.59)]
#[q=air density(1.225kg/m^3 at sea level)]
#[A=Swept area of the rotor (largest rotor diameter used in the current time is 129m, so swept area=13070m^2) ]
#[V=Wind Speed(m/s), which in this case have been forecasted]
FutureWP<-data.frame("Tamil Nadu Wind"=((0.5 * 0.59 * 1.225 * 13070 * TNWS.forecast$mean^3)/1000),
                     "Maharashtra Wind Power"=((0.5 * 0.59 * 1.225 * 13070 * MHWS.forecast$mean^3)/1000))

write.table(FutureWP, "C:/Users/HP/Downloads/Research Project/Data/Future_WP.csv",
            sep=" ,", row.names=FALSE)
```

Figure 11: Wind Power Calculation

All the code snippets shown before were each having a similar counter code for the other state. As the code were the same, so they are not shown in this summary. The Visualizations obtained from forecasts of the fitted models are attached in the following section.

4 Forecasted Outcomes

The Forecasted visualizations are as follows-

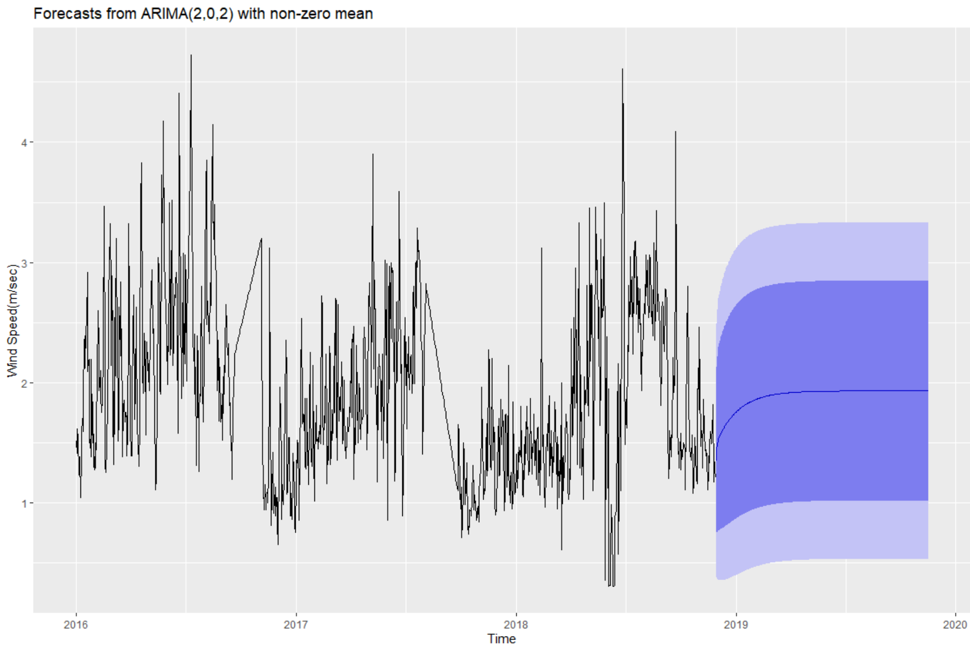


Figure 12: ARIMA Forecast of Maharashtra

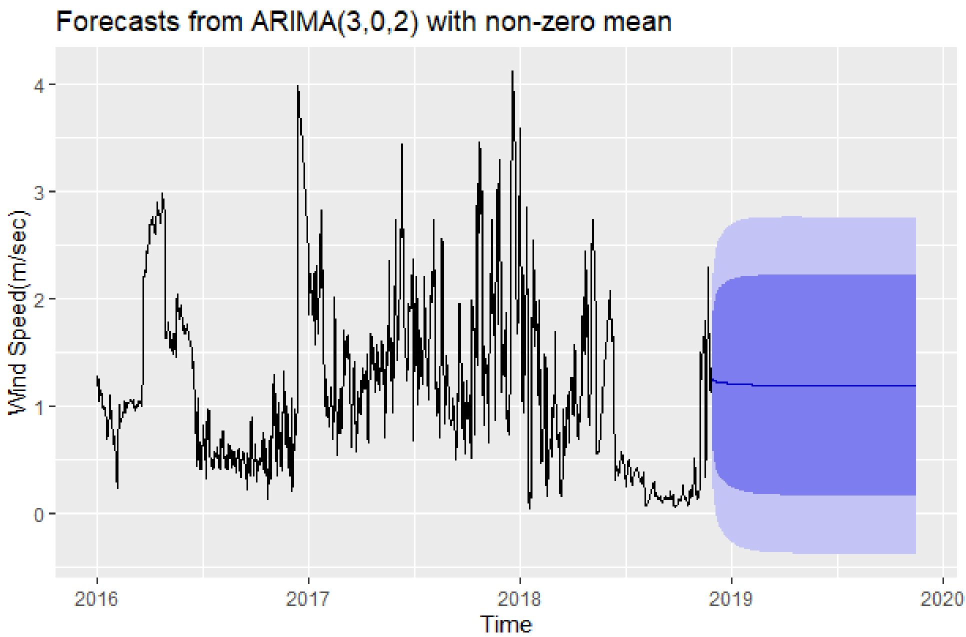


Figure 13: ARIMA Forecast of Tamil Nadu

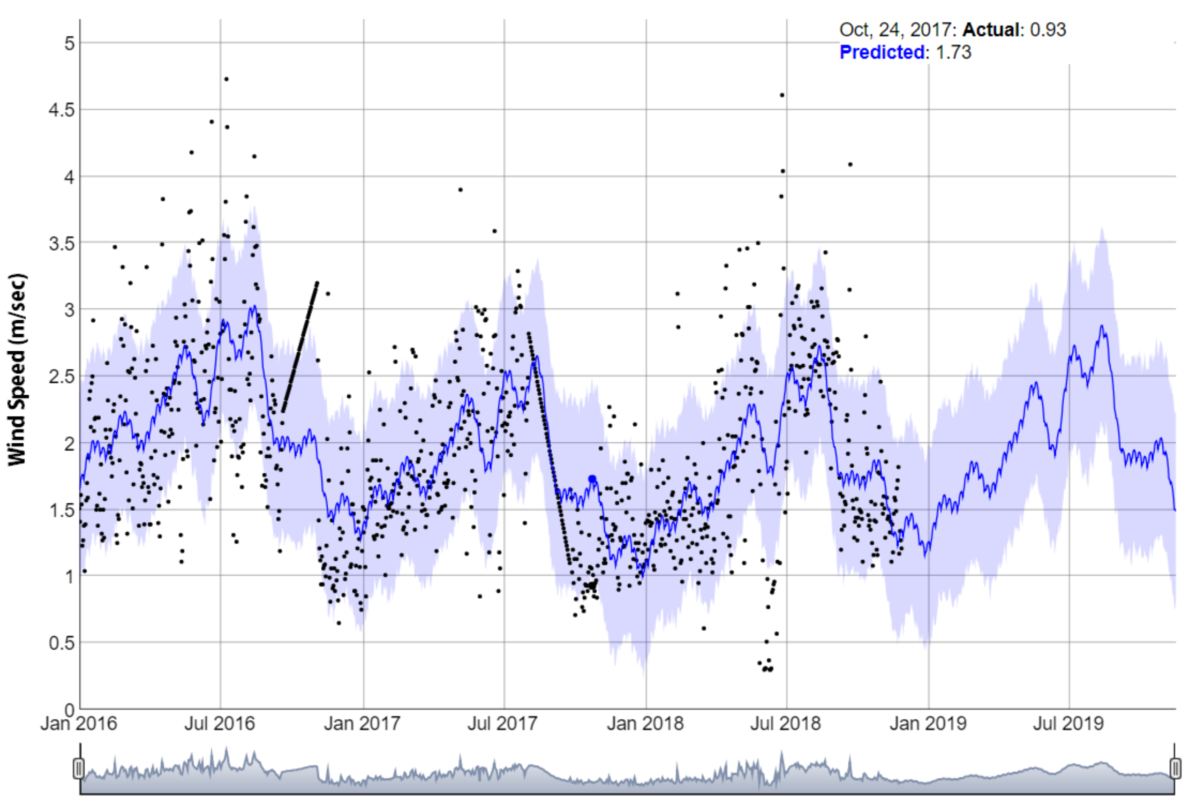


Figure 14: Prophet Forecast of Maharashtra

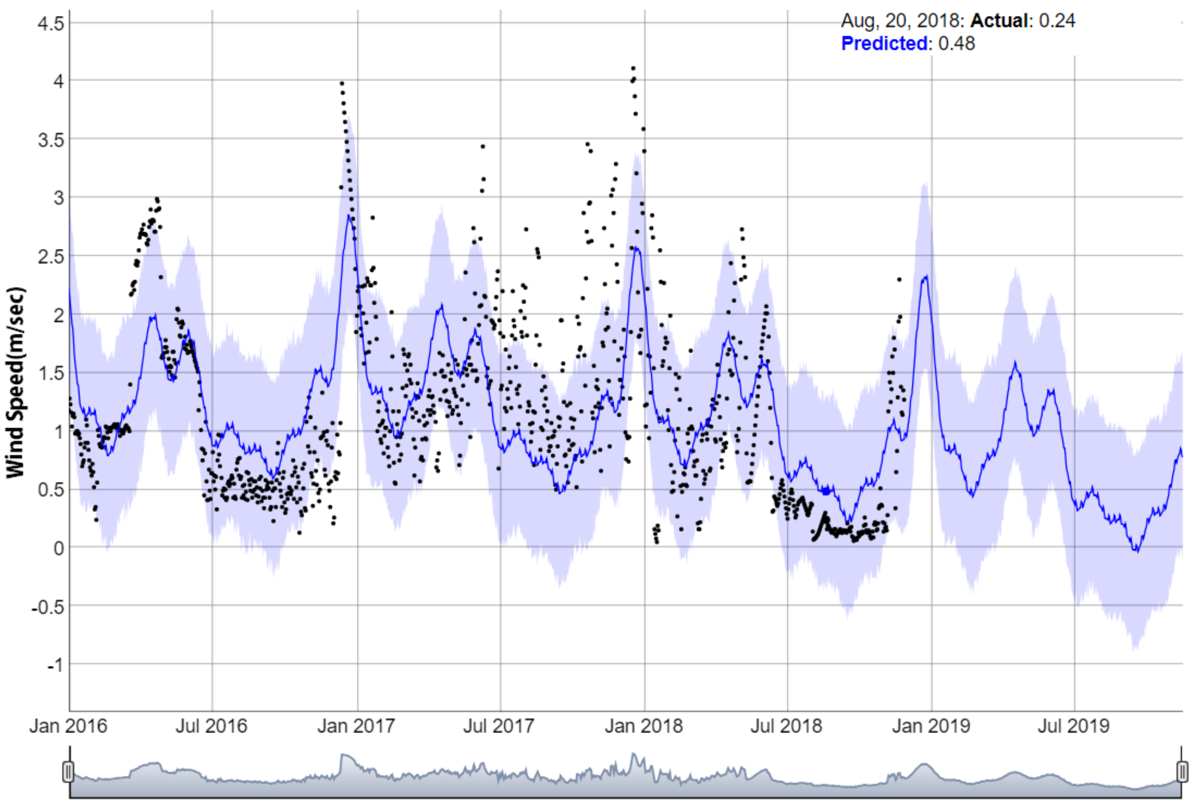


Figure 15: Prophet Forecast of Tamil Nadu

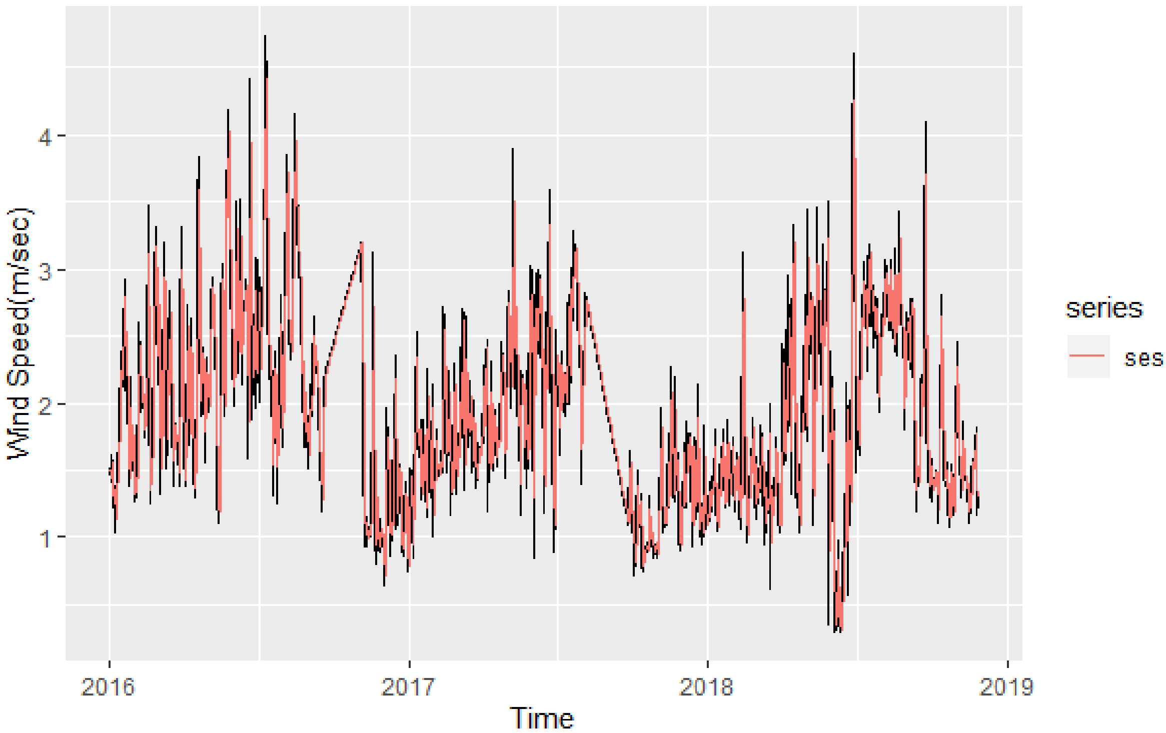


Figure 16: Simple Exponential Smoothng Forecast of Maharashtra

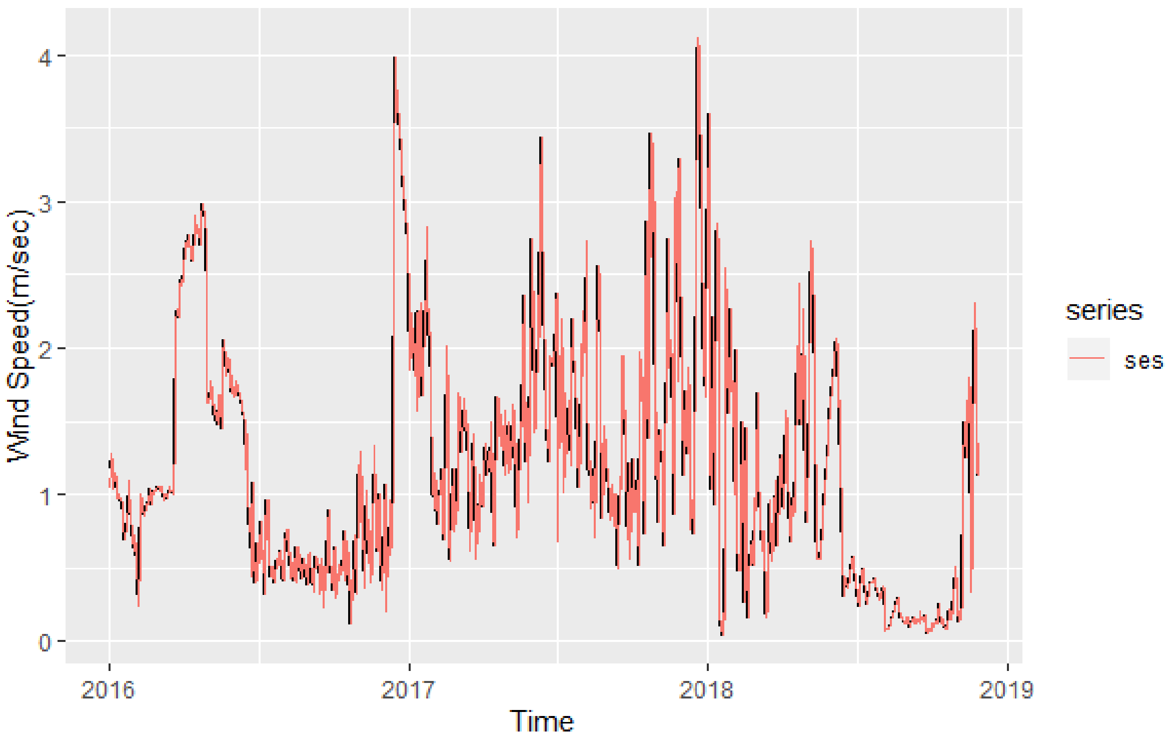


Figure 17: Simple Exponential Smoothing Forecast of Tamil Nadu

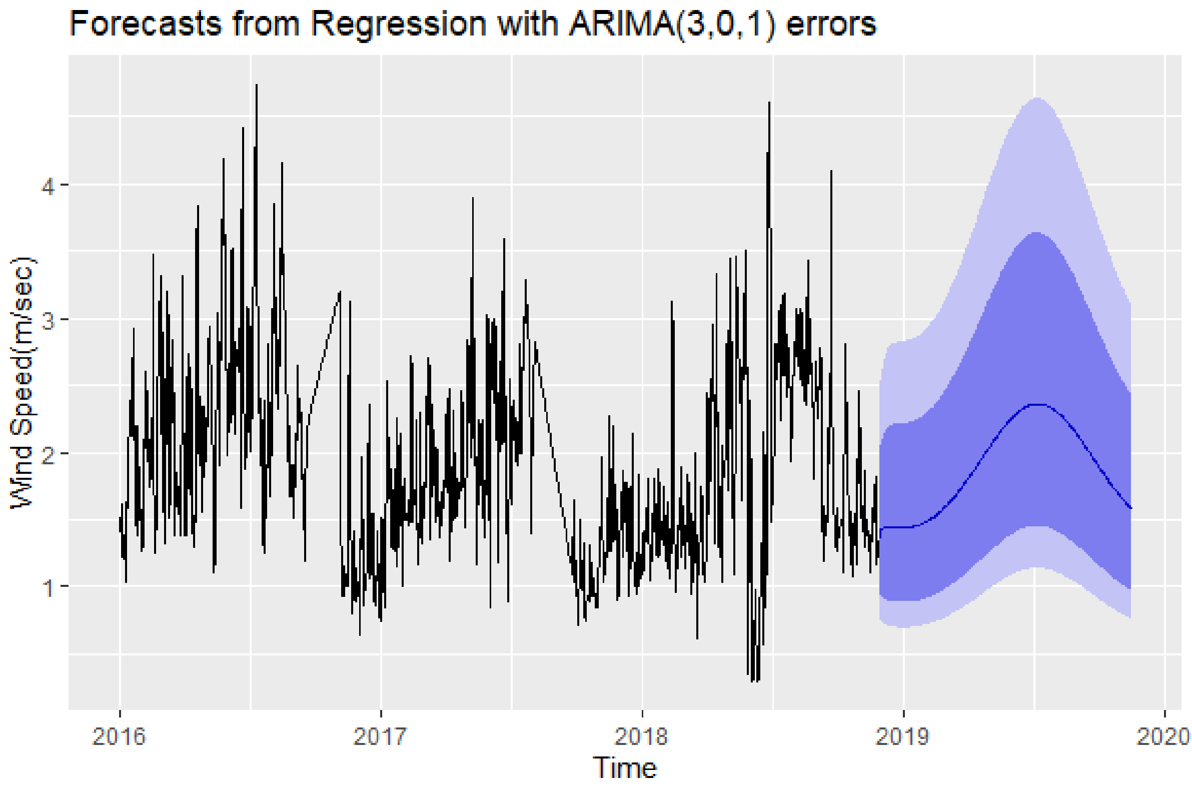


Figure 18: Dynamic Harmonic Rwgression Forecast of Maharashtra

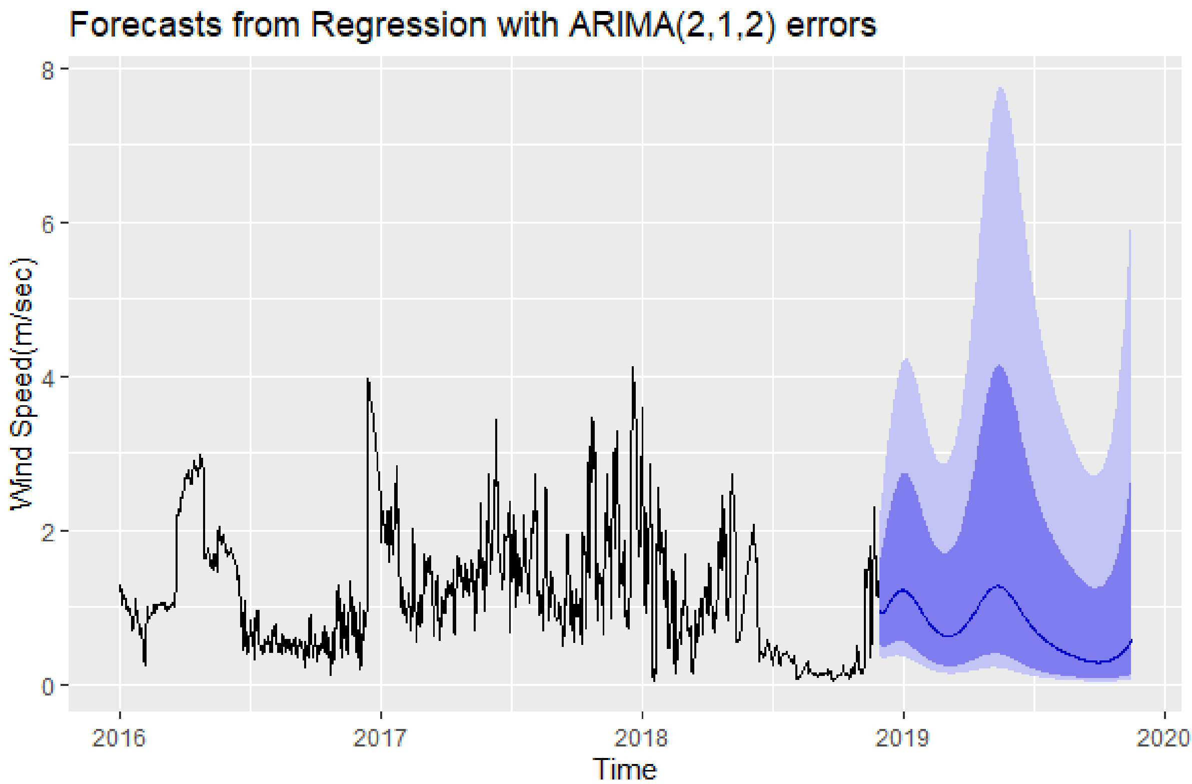


Figure 19: Dynamic Harmonic Rwgression Forecast of Tamil Nadu

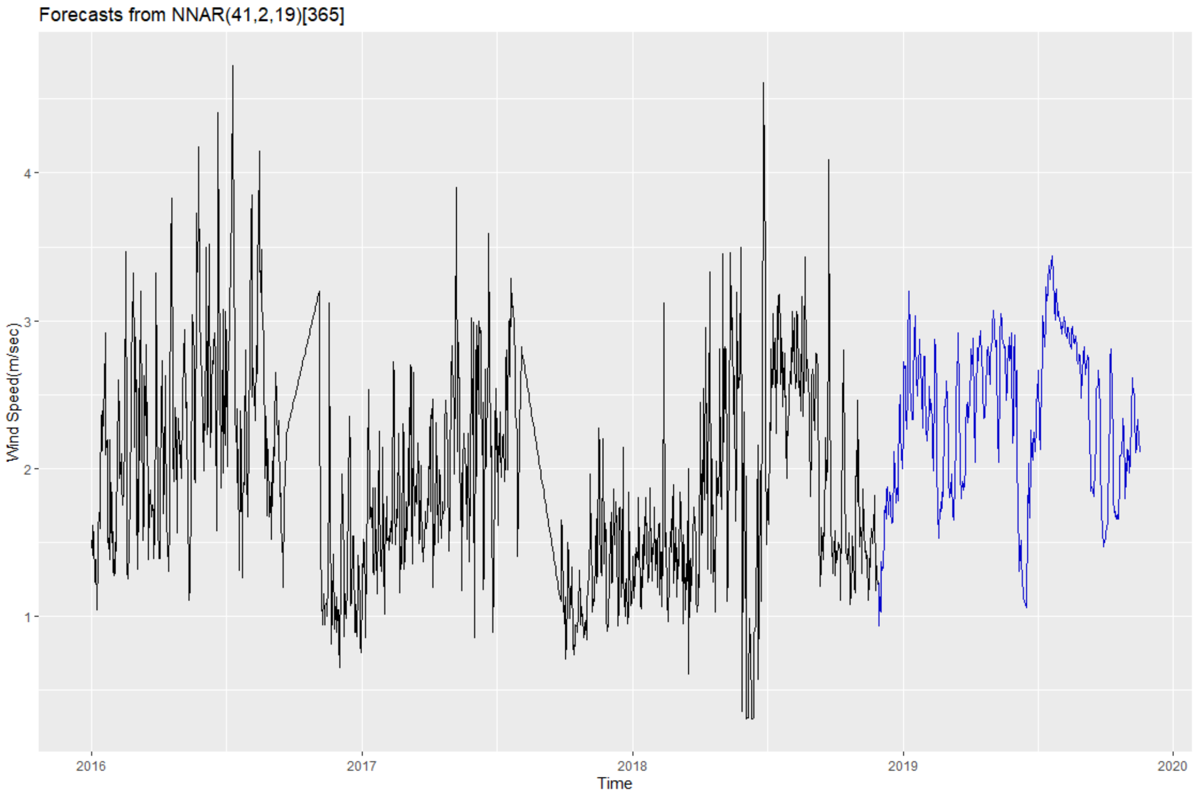


Figure 20: Dynamic Harmonic Rwgression Forecast of Maharashtra

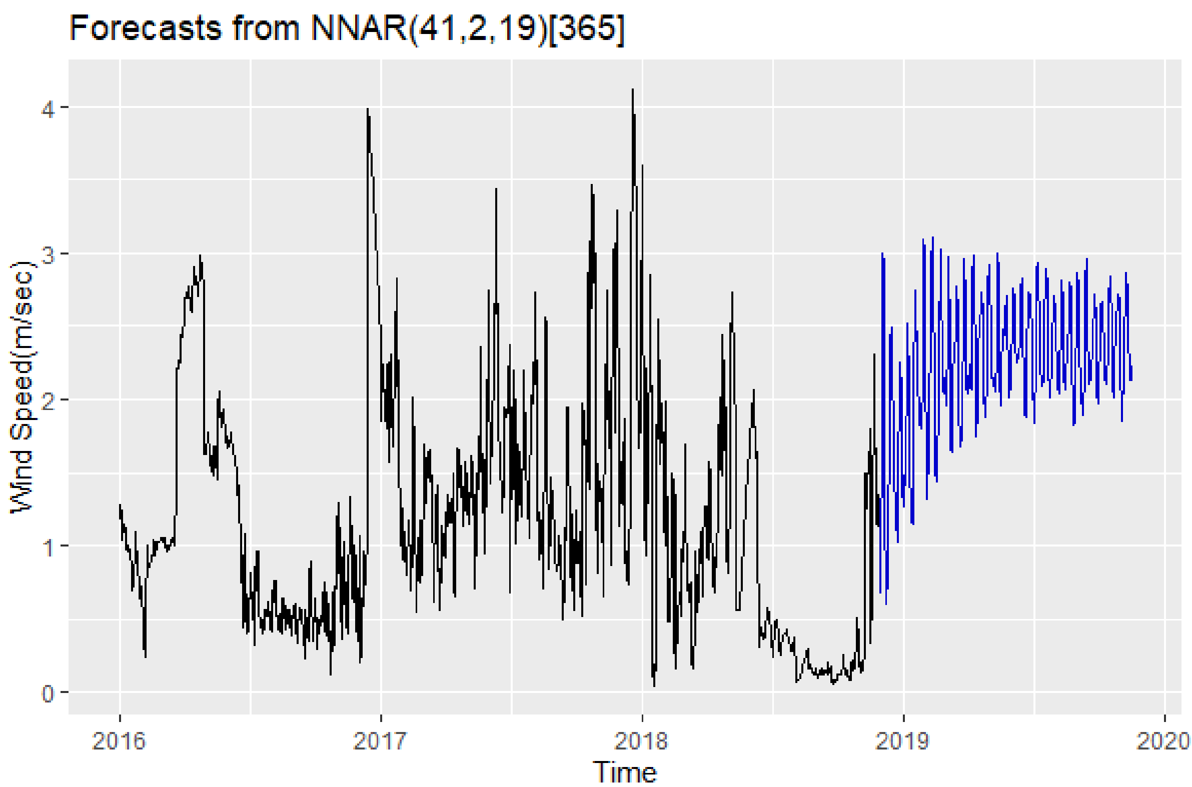


Figure 21: Dynamic Harmonic Rwgression Forecast of Tamil Nadu