

HuanHusted_TSA_Competition

Kelsey Husted

2023-03-31

Time Series Competition

```
knitr::opts_chunk$set(echo = TRUE, tidy.opts=list(width.cutoff=80), tidy=FALSE)
```

```
##Load packages
```

```
library(readxl)
library(dplyr)
library(lubridate)
library(openxlsx)
library(ggplot2)
library(forecast)
library(Kendall)
library(tseries)
library(outliers)
library(tidyverse)
library(smooth)
library(zoo)
library(kableExtra)
```

Data wrangling and processing from 2005-2009

Hourly data was transformed into daily data with aggregate functions and pipes (i.e., tidyverse).

```
#Import data
df <- read_xlsx("./Competition/Data/load.xlsx")

#Wrangle data from hourly to daily
#Wrangling date column 2005 to 2009
df_daily <- df %>%
  mutate( Date = ymd(date)) %>%
  filter(Date < '2010-01-01')

#removing no numeric columns so rowMeans() functino will work
df_dailyV2 <- df %>%
  mutate( Date = ymd(date)) %>%
  filter(Date < '2010-01-01') %>%
  select(3:26)

#Creating daily data
df_processed <- df_dailyV2 %>%
  mutate(rowMeans(df_dailyV2)) %>%
  rename(Daily_data = "rowMeans(df_dailyV2)") %>%
```

```

select(25)

#Combining date and daily data
date <- df_daily[,2]
df_processed <- cbind(date, df_processed)

nobs = nrow(df_daily)

```

Data wrangling and processing from 2005-2010

The data needs to be formatted to include 2010 as well since the objective is to forecast for 2011. Instead of making two separate datasets, I should use the `window()` function for future reference.

```

#Wrangle data from hourly to daily
#Wrangling date column 2005 to 2010
#removing no numeric columns so rowMeans() function will work
df_daily2010 <- df %>%
  mutate( Date = ymd(date)) %>%
  select(3:26)

#Creating daily data
df_processed2010 <- df_daily2010 %>%
  mutate(rowMeans(df_daily2010)) %>%
  rename(Daily_data = "rowMeans(df_daily2010)") %>%
  select(25)

#Combining data and daily data
date <- df[,2]
df_processed2010 <- cbind(date, df_processed2010)

nobs2010 = nrow(df_processed2010)

```

Time series object transformation

```

#ts transformation 2005 to 2009
ts_daily <- msts(df_processed$Daily_data,
  seasonal.periods=c(7,365.25),
  start=c(2005, 01, 01))

#ts transformation 2005 to 2010
ts_daily2010 <- msts(df_processed2010$Daily_data,
  seasonal.periods=c(7,365.25),
  start=c(2005, 01, 01))

```

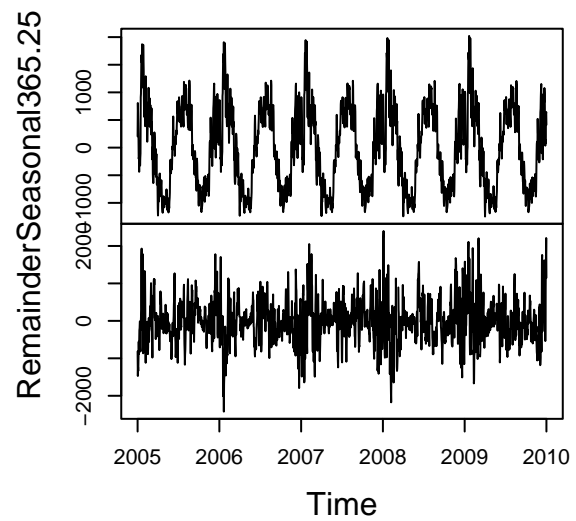
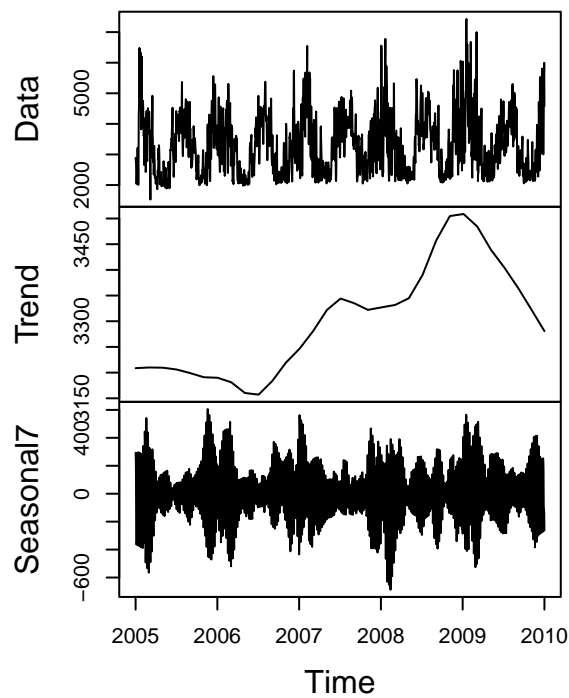
Time series decomposition and plot

```

#Decompose time series
ts_decompose <- ts_daily %>%
  mstl()
plot(ts_decompose)

```

ts_decompose



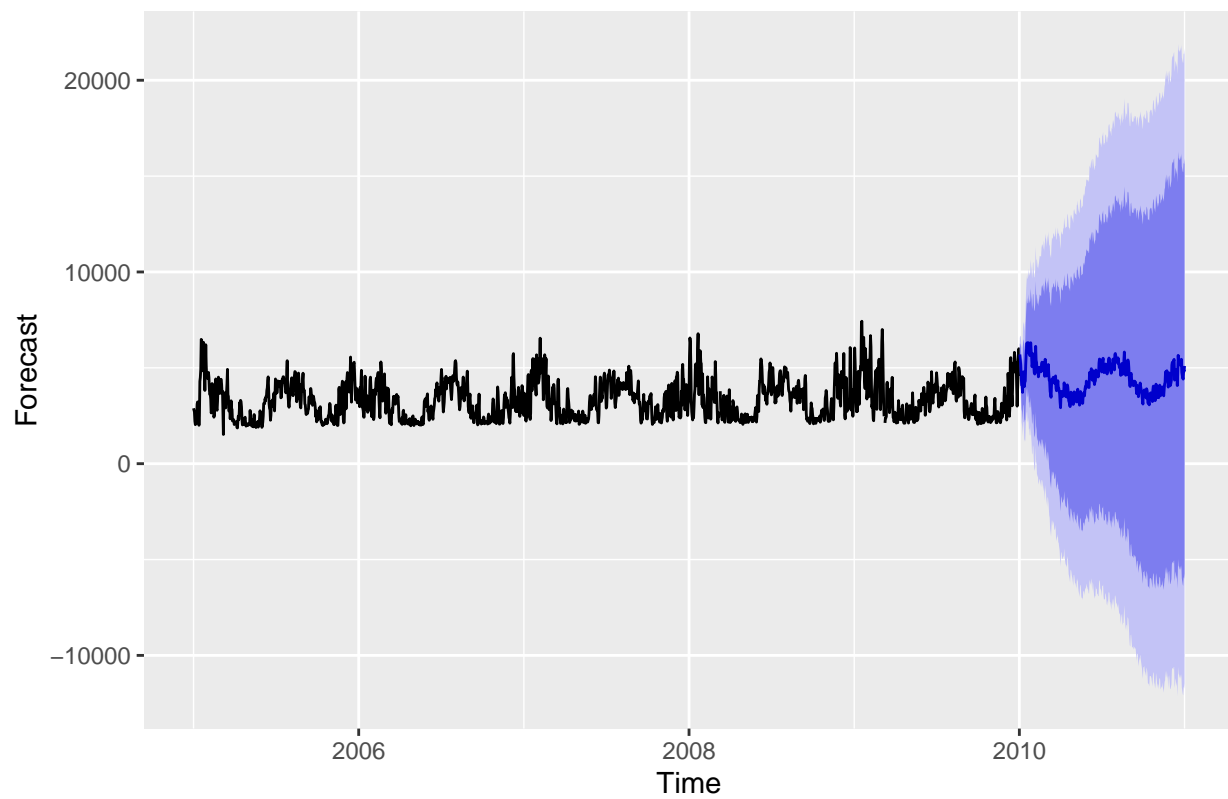
##

Model 1 STL + ETS: Forecast 2010

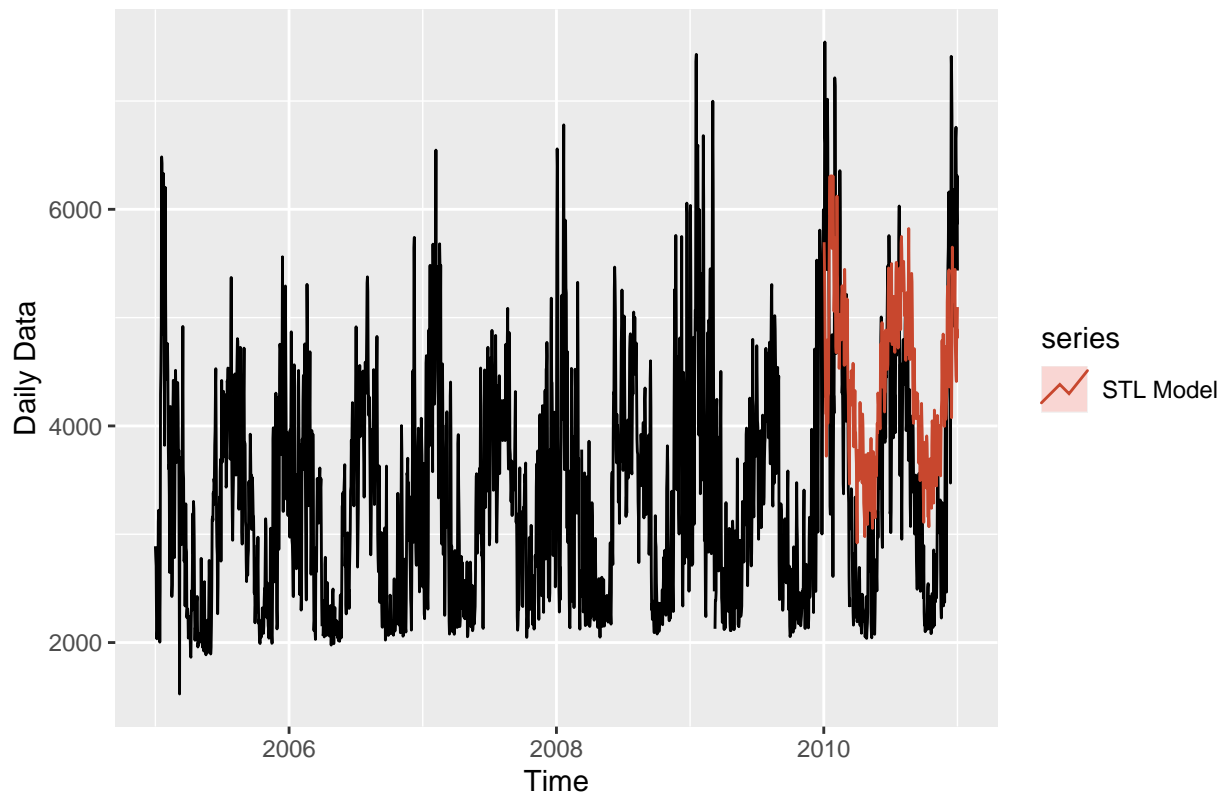
```
#Fit and forecast STL model
ETS_model <- stlf(ts_daily,h=365)

#Plot forecasting
autoplot(ETS_model) + ylab("Forecast")
```

Forecasts from STL + ETS(A,N,N)



```
#Plot model + observed data  
autoplot(ts_daily2010) +  
  autolayer(ETS_model, series="STL Model",PI=FALSE) +  
  ylab("Daily Data")
```



```
#Check accuracy of model
n_for <- 365
observed <- df_processed[(nobs-n_for+1):nobs, "Daily_data"]
ETS_scores <- accuracy(ETS_model$mean,observed)
print(ETS_scores)
```

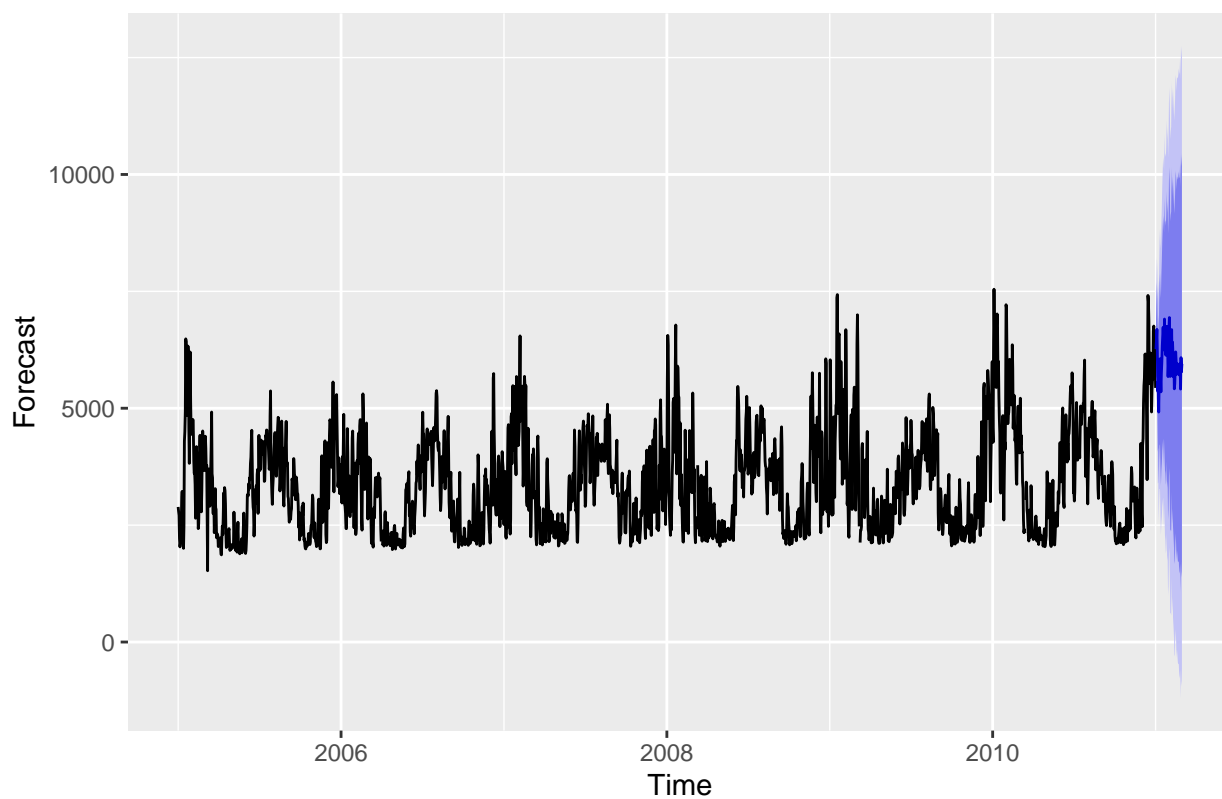
```
##           ME      RMSE      MAE      MPE      MAPE
## Test set -984.5201 1210.625 1079.332 -35.24753 36.80666
```

Model 1 STL + ETS: Forecast 2011

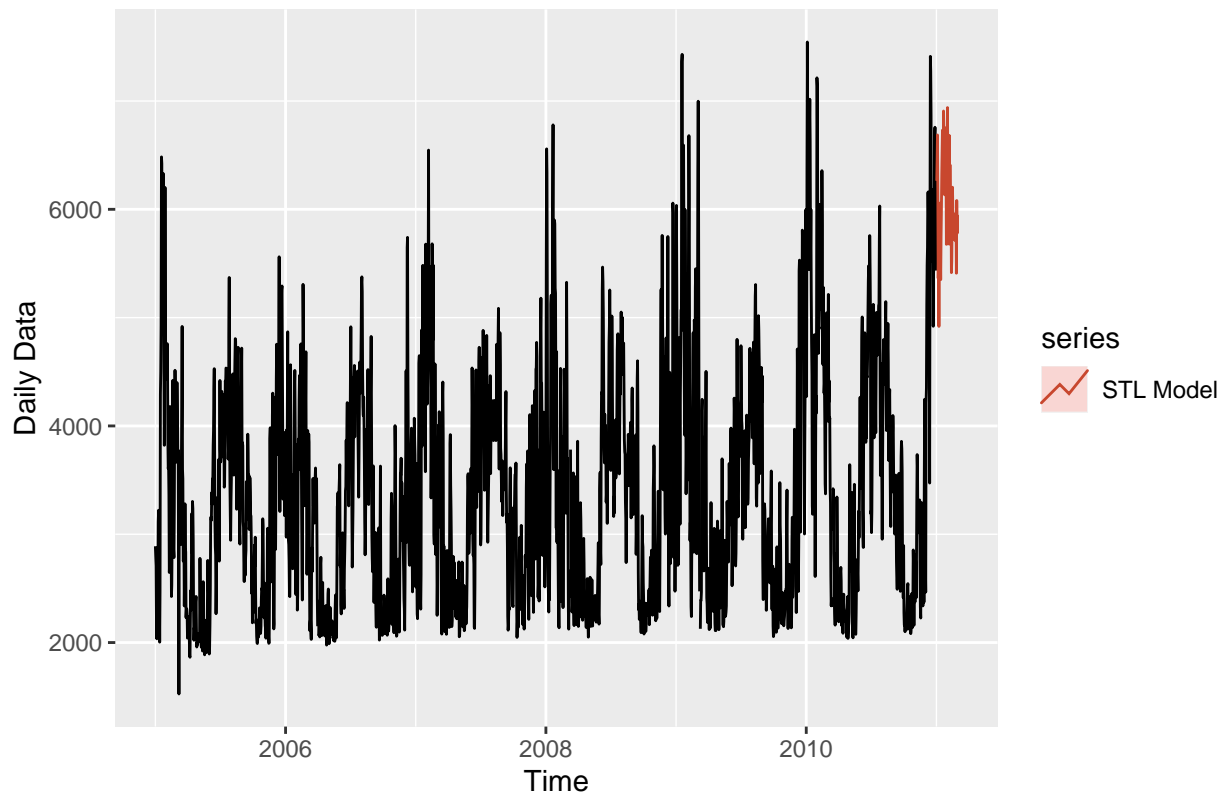
```
#Fit and forecast STL model January 1st to February 28th 2011
ETS_model2011 <- stlf(ts_daily2010,h=59)

#Plot forecasting
autoplot(ETS_model2011) + ylab("Forecast")
```

Forecasts from STL + ETS(A,N,N)



```
#Plot model + observed data  
autoplot(ts_daily2010) +  
  autolayer(ETS_model2011, series="STL Model",PI=FALSE) +  
  ylab("Daily Data")
```



```
#Check accuracy of model
n_for <- 59
observed <- df_processed[(nobs-n_for+1):nobs, "Daily_data"]
ETS_scores <- accuracy(ETS_model2011$mean,observed)
print(ETS_scores)
```

```
##           ME      RMSE      MAE      MPE      MAPE
## Test set -2528.13 2846.706 2535.511 -90.47756 90.60064
```

Model 2 TBATS: Forecast 2010

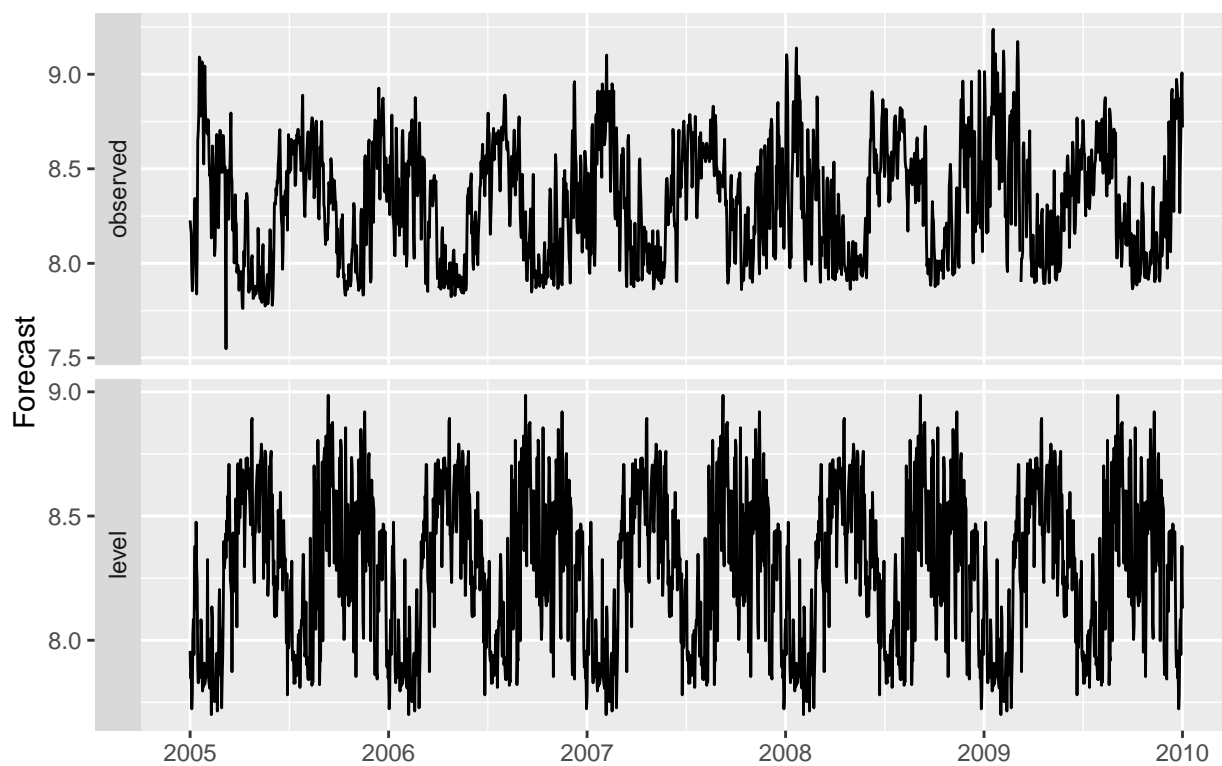
The model looks like a really bad fit visually and will not be used to forecast for 2011.

```
#Fit and forecast TBATS model
TBATS_model <- tbats(ts_daily)

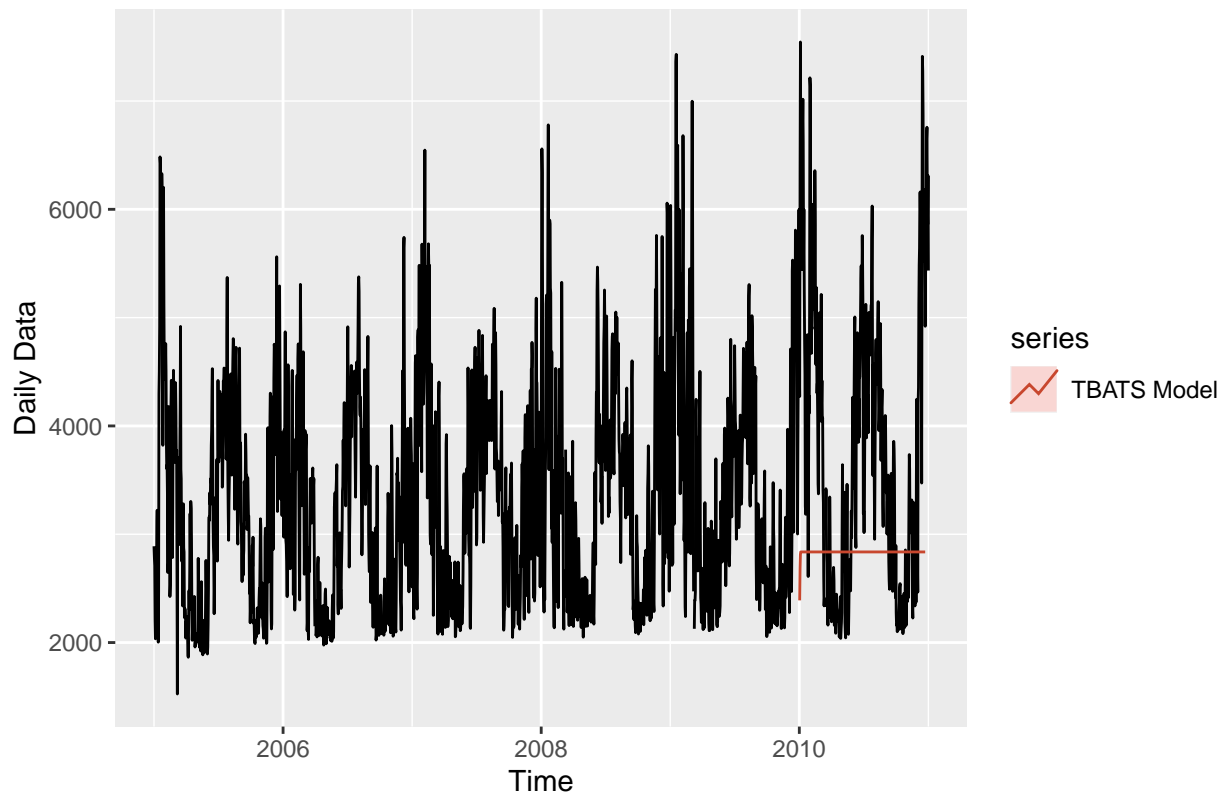
#forecast
TBATS_for <- forecast(TBATS_model,h=356)

#Plot forecasting
autoplot(TBATS_model) + ylab("Forecast")
```

Components of TBATS method



```
#Plot model + observed data  
autoplot(ts_daily2010) +  
  autolayer(TBATS_for, series="TBATS Model",PI=FALSE) +  
  ylab("Daily Data")
```

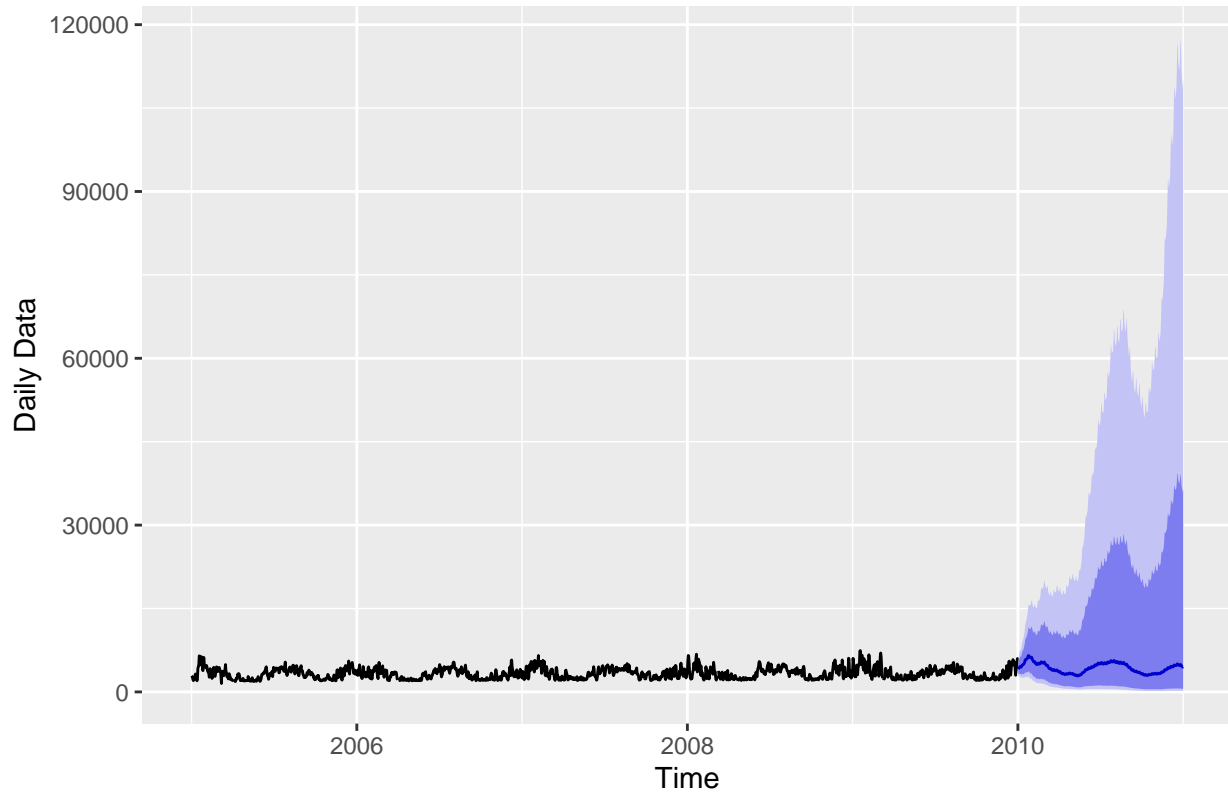
Model 3 ARIMA + FOURIER terms: Forecast 2010

```
#Fit and forecast TBATS model
ARIMA_Four_model <- auto.arima(ts_daily,
                               seasonal=FALSE,
                               lambda=0,
                               xreg=fourier(ts_daily,
                                             K=c(2,12))
                               )

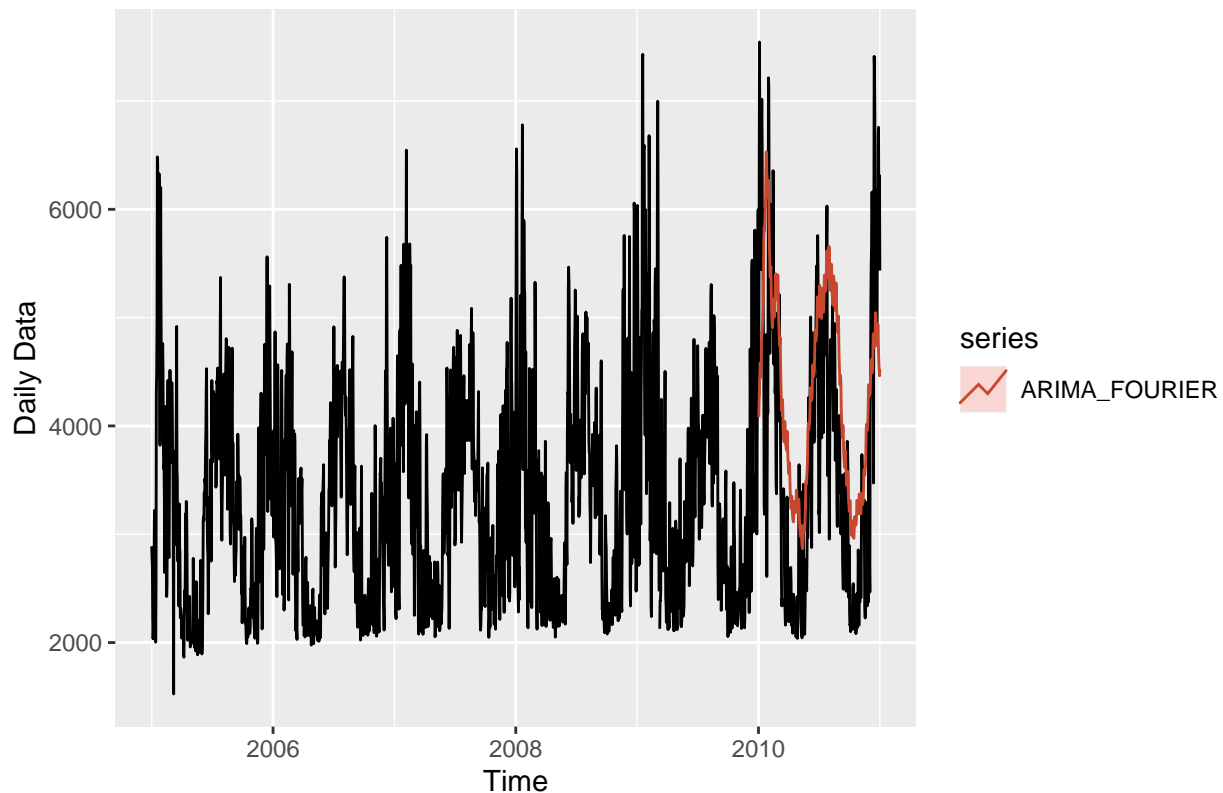
#Forecast
ARIMA_Four_for <- forecast(ARIMA_Four_model,
                           xreg=fourier(ts_daily,
                                         K=c(2,12),
                                         h=365),
                           h=365
                           )

#Plot foresting results
autoplot(ARIMA_Four_for) + ylab("Daily Data")
```

Forecasts from Regression with ARIMA(5,1,0) errors



```
#Plot model + observed data  
autoplot(ts_daily2010) +  
  autolayer(ARIMA_Four_for, series="ARIMA_FOURIER", PI=FALSE) +  
  ylab("Daily Data")
```



```
#Check accuracy of model
n_for <- 365
observed <- df_processed[(nobs-n_for+1):nobs, "Daily_data"]
ARIMA_Four_scores <- accuracy(ARIMA_Four_for$mean,observed)
print(ARIMA_Four_scores)
```

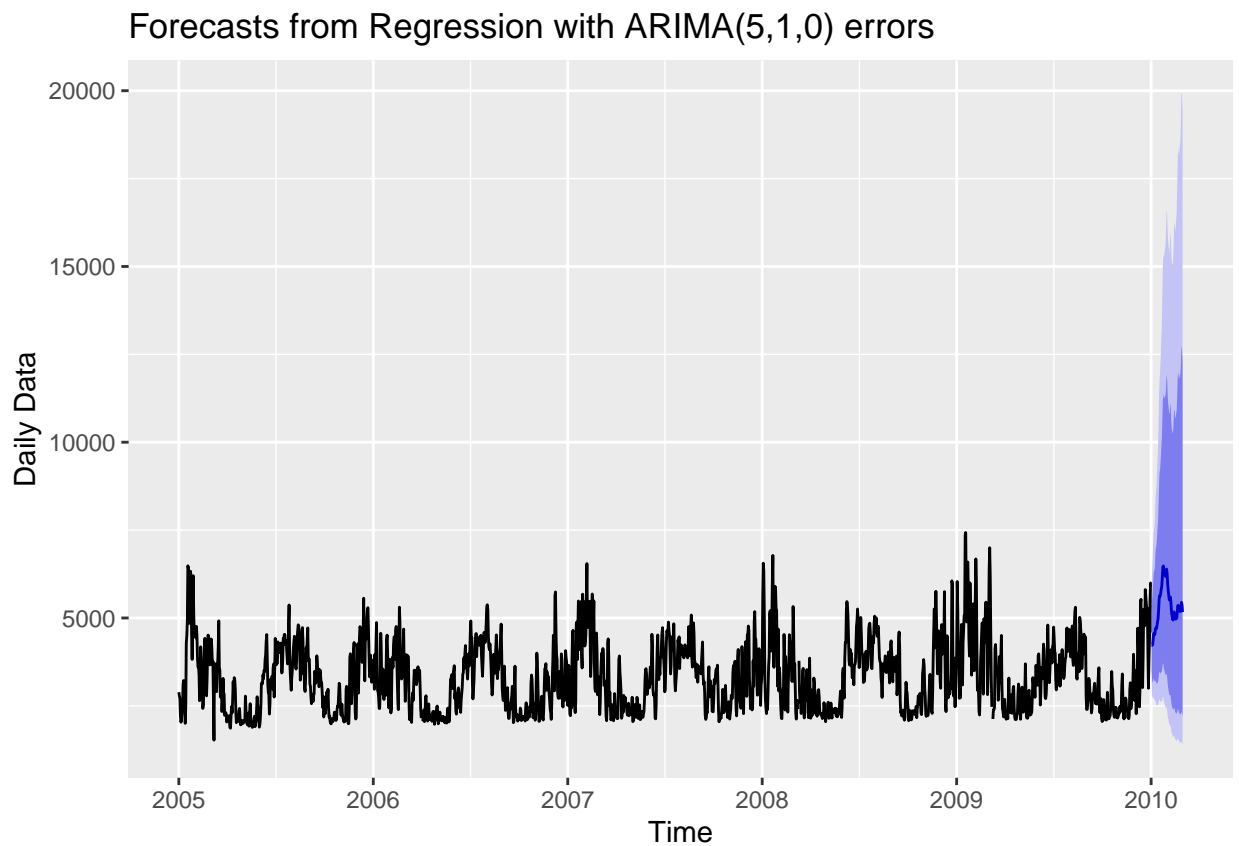
```
##              ME      RMSE      MAE      MPE      MAPE
## Test set -857.4621 1162.004 994.7404 -30.51384 32.8675
```

Model 3 ARIMA + FOURIER terms: Forecast 2011

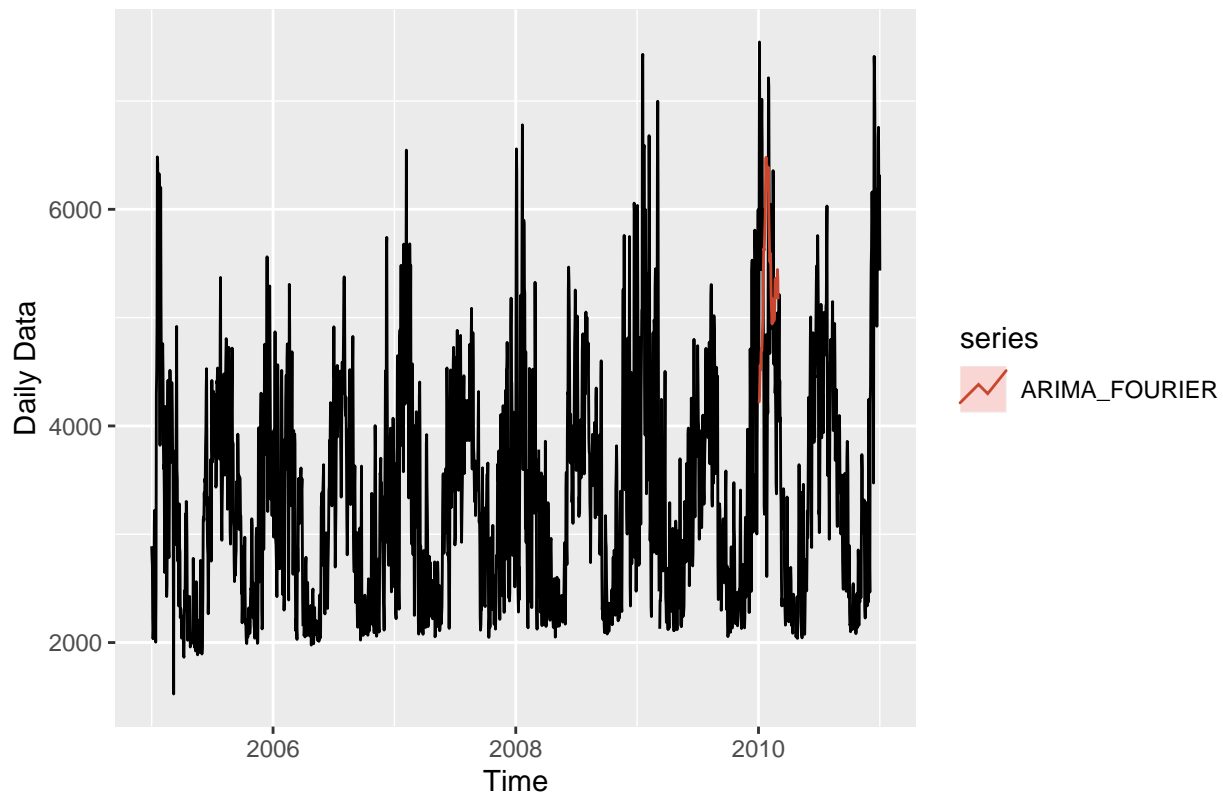
```
#Fit and forecast TBATS model
ARIMA_Four_model2011 <- auto.arima(ts_daily2010,
                                   seasonal=FALSE,
                                   lambda=0,
                                   xreg=fourier(ts_daily2010,
                                                K=c(2,12))
                                   )

#Forecast
ARIMA_Four_for2011 <- forecast(ARIMA_Four_model,
                               xreg=fourier(ts_daily2010,
                                              K=c(2,12),
                                              h=59),
                               h=59
                               )

#Plot forecasting results
autoplot(ARIMA_Four_for2011) + ylab("Daily Data")
```



```
#Plot model + observed data  
autoplot(ts_daily2010) +  
  autolayer(ARIMA_Four_for2011, series="ARIMA_FOURIER",PI=FALSE) +  
  ylab("Daily Data")
```



```
#Check accuracy of model
n_for <- 59
observed <- df_processed[(nobs-n_for+1):nobs, "Daily_data"]
ARIMA_Four_scores <- accuracy(ARIMA_Four_for$mean,observed)
print(ARIMA_Four_scores)
```

```
##                ME      RMSE      MAE      MPE      MAPE
## Test set -1832.672 2258.961 1948.471 -67.73503 69.78746
```

```
#print(ARIMA_Four_for2011$mean)
```

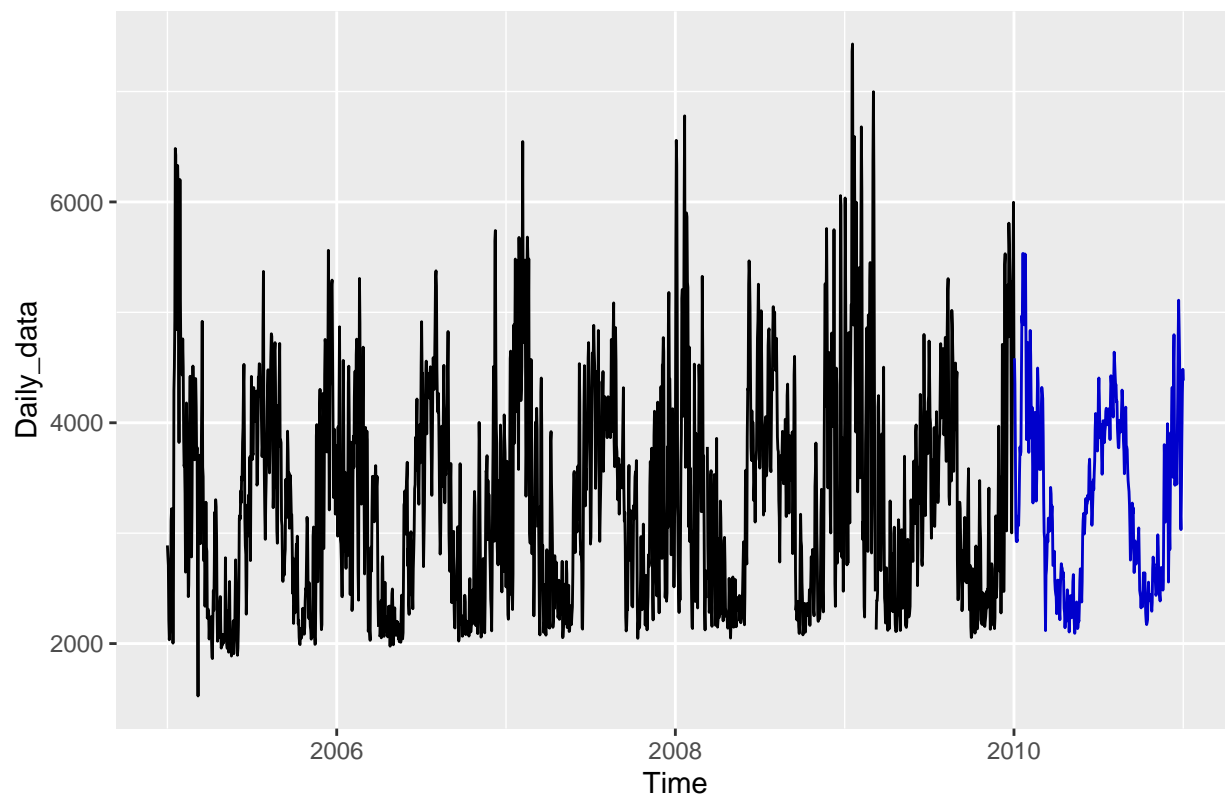
Model 4 Neural Network Time Series: Forecasts 2010

```
#NN_fit <- nnetar(ts_act_power_daily_train,p=1,P=1)
NN_model <- nnetar(ts_daily,decay=0.5, maxit=150, p=1,P=0,xreg=fourier(ts_daily, K=c(2,12)))

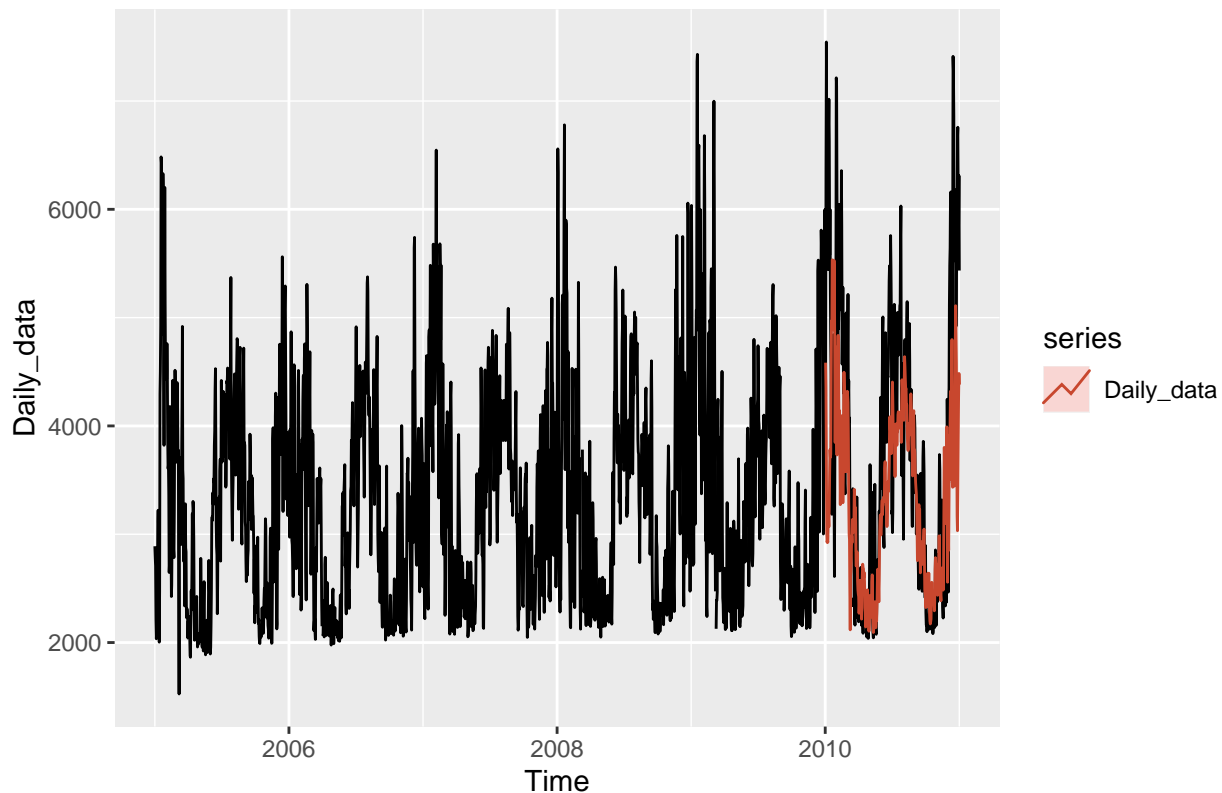
#NN_for <- forecast(NN_fit, h=365)
NN_for <- forecast(NN_model, h=365,xreg=fourier(ts_daily,
                                                K=c(2,12),h=365))

#Plot foresting results
autoplot(NN_for) +
  ylab("Daily_data")
```

Forecasts from NNAR(1,15)



```
#Plot model + observed data  
autoplot(ts_daily2010) +  
  autolayer(NN_for, series="Daily_data",PI=FALSE)+  
  ylab("Daily_data")
```



```
#Checking error variables to decide which model fits the data the best
n_for <- 365
observed <- df_processed[(nobs-n_for+1):nobs, "Daily_data"]
NN_scores1 <- accuracy(NN_for$mean,observed)
print(NN_scores1)
```

```
##              ME      RMSE      MAE      MPE      MAPE
## Test set 127.1238 721.3404 534.6116 0.1142971 14.92668
```

Model 4 Neural Network Time Series: Forecasts 2011

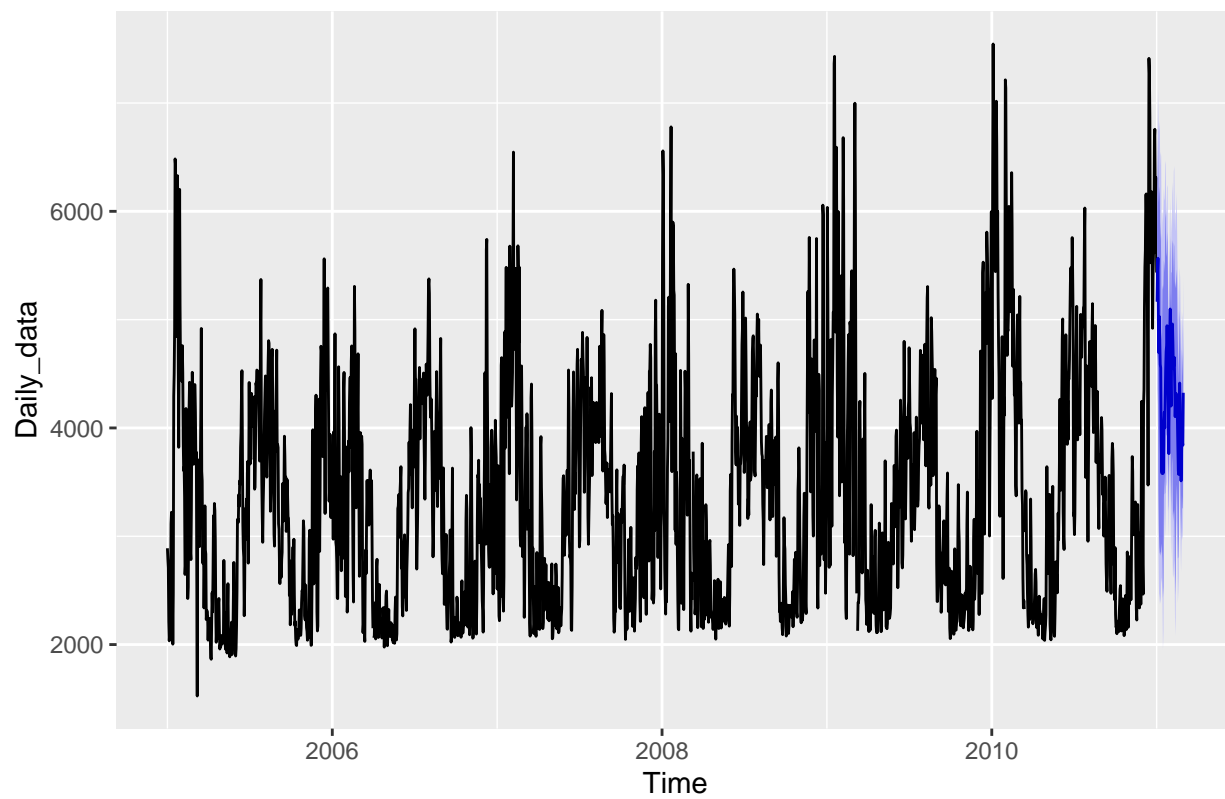
Note: Based on the error variables calculated with the accuracy() function, the Neural Network model seems to fit the data the best.

```
#NN_fit <- nnetar(ts_act_power_daily_train,p=1,P=1)
NN_model2010 <- nnetar(ts_daily2010,lambda = 0.5,p=1,P=0,xreg=fourier(ts_daily2010, K=c(2,12)))

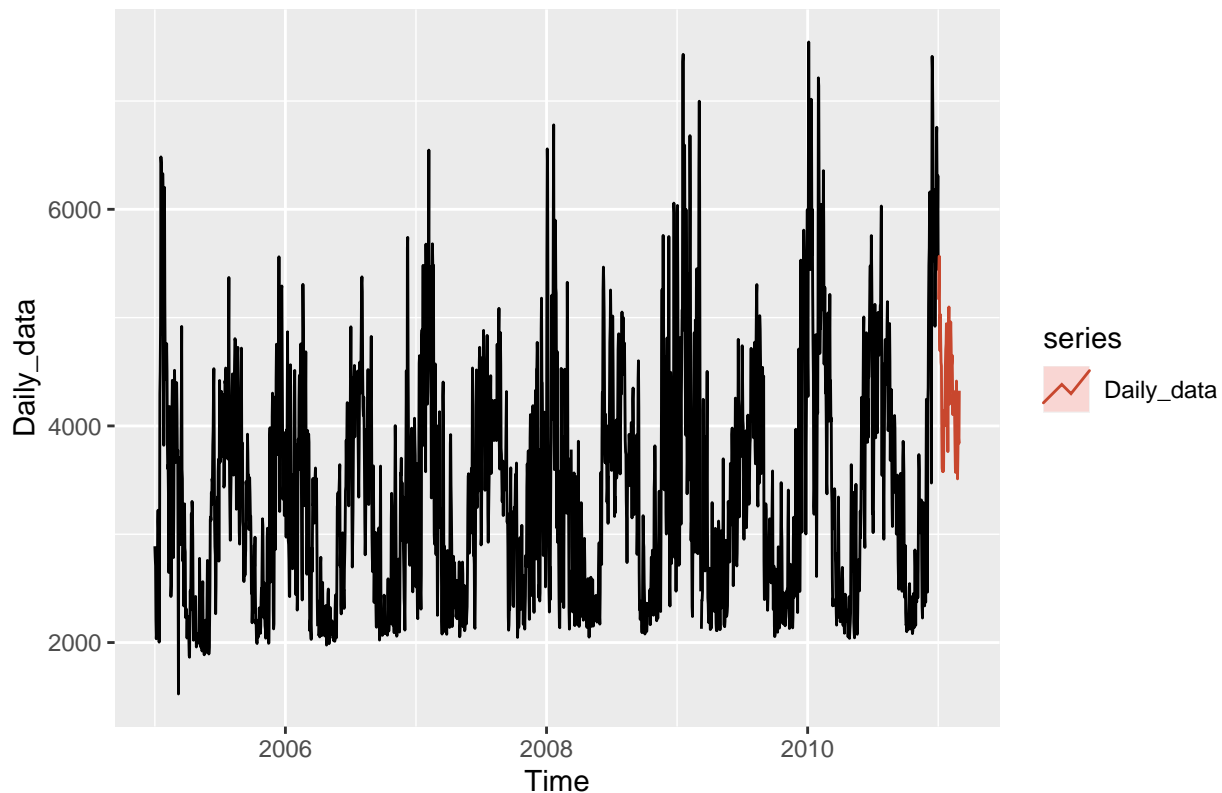
#NN_for <- forecast(NN_fit, h=365)
NN_for2010 <- forecast(NN_model2010,PI=TRUE, h=59,xreg=fourier(ts_daily,
                                                                K=c(2,12),h=59))

#Plot foresting results
autoplot(NN_for2010) +
  ylab("Daily_data")
```

Forecasts from NNAR(1,15)



```
#Plot model + observed data  
autoplot(ts_daily2010) +  
  autolayer(NN_for2010, series="Daily_data",PI=FALSE)+  
  ylab("Daily_data")
```

```
n_for <- 59
observed <- df_processed[(nobs-n_for+1):nobs, "Daily_data"]
NN_scores1 <- accuracy(NN_for2010$mean,observed)
print(NN_scores1)
```

```
##           ME      RMSE      MAE      MPE      MAPE
## Test set -826.9043 1586.724 1402.582 -37.04755 47.72392
```

```
print(NN_for2010$mean)
```

```
## Multi-Seasonal Time Series:
```

```
## Start: 2011 2
```

```
## Seasonal Periods: 7 365.25
```

```
## Data:
```

```
## [1] 5166.186 5449.052 5566.688 4829.817 4693.982 5028.454 4772.463 4587.167
## [9] 4554.188 3964.770 3610.373 3580.220 3802.931 3580.138 3601.748 4058.169
## [17] 4145.582 4000.039 4266.917 4546.132 4706.348 4758.783 4943.783 4761.355
## [25] 4456.957 4190.473 3763.085 4318.888 4855.398 5098.434 4413.309 4202.363
## [33] 4591.779 4758.524 4959.781 4777.864 4547.086 4580.151 4651.508 4404.896
## [41] 4104.143 4267.752 4319.709 4295.005 4279.063 4076.134 3822.421 3602.155
## [49] 3570.328 4131.925 4414.513 4218.860 3800.942 3511.317 3719.701 4010.911
## [57] 3880.124 3840.262 4324.888
```