TSA: Forecasting Competition

Taro Katayama Miaojun Pang

CREATE A REPOSITORY IN YOUR GITHUB ACCOUNT

1. Go to your user account on GitHub and navigate to the repositories tab.
2. In the upper right corner, click the green "New" button.
3. Name your repository with recommended naming conventions (suggestion: *Lastname1Lastname2_ENV790_TSA_Compe*
   Write a short description of the purpose of the repository. Check the box to initialize the repository with a README. Add a .gitignore for R and add a GNU General Public License v3.0.
4. Invite other group members as collaborators to the repository.

LINK YOUR REPO TO YOUR LOCAL DRIVE WITH RSTUDIO

1. Click the "Clone or download" button for your repository and then the "copy" icon. Make sure the box header lists "Clone with HTTPS" rather than "Clone with SSH." If not, click the "Use HTTPS" button and then copy the link.
2. Launch RStudio and select "New Project" from the File menu. Choose "Version Control" and "Git."
3. Paste the repository URL and give your repository a name and a file path.

IMPORT THE DATASET

In the folder `/Competition/Data` you will find three datasets one with hourly demand, one with hourly temperature and another with relative humidity from January 2005 to December 2010. Your goal is to forecast **daily** demand for the month of January 2011 based on this historical data. You may or may not use the temperature and relative humidity in your models. The temperature and humidity measurement are from stations close to the household meter data you have.

```
library(lubridate)
library(ggplot2)
library(forecast)
library(tseries)
library(outliers)
library(tidyverse)
library(readxl)
library(zoo)
```

```
#load data
getwd()
```

```
## [1] "/Users/tarokatayama/Desktop/Duke_Semester_4/Time_Series_Analysis/TSCompetition/Competition"
```

```
load<- read_excel("../Competition/Data/load.xlsx")
temperature<- read_excel("../Competition/Data/temperature.xlsx")
```

WRANGLE/PROCESS THE DATASET

You will need to transform hourly data into daily data. See the Rmd file from Lesson 11 for instructions on how to aggregate your dataset using pipes.

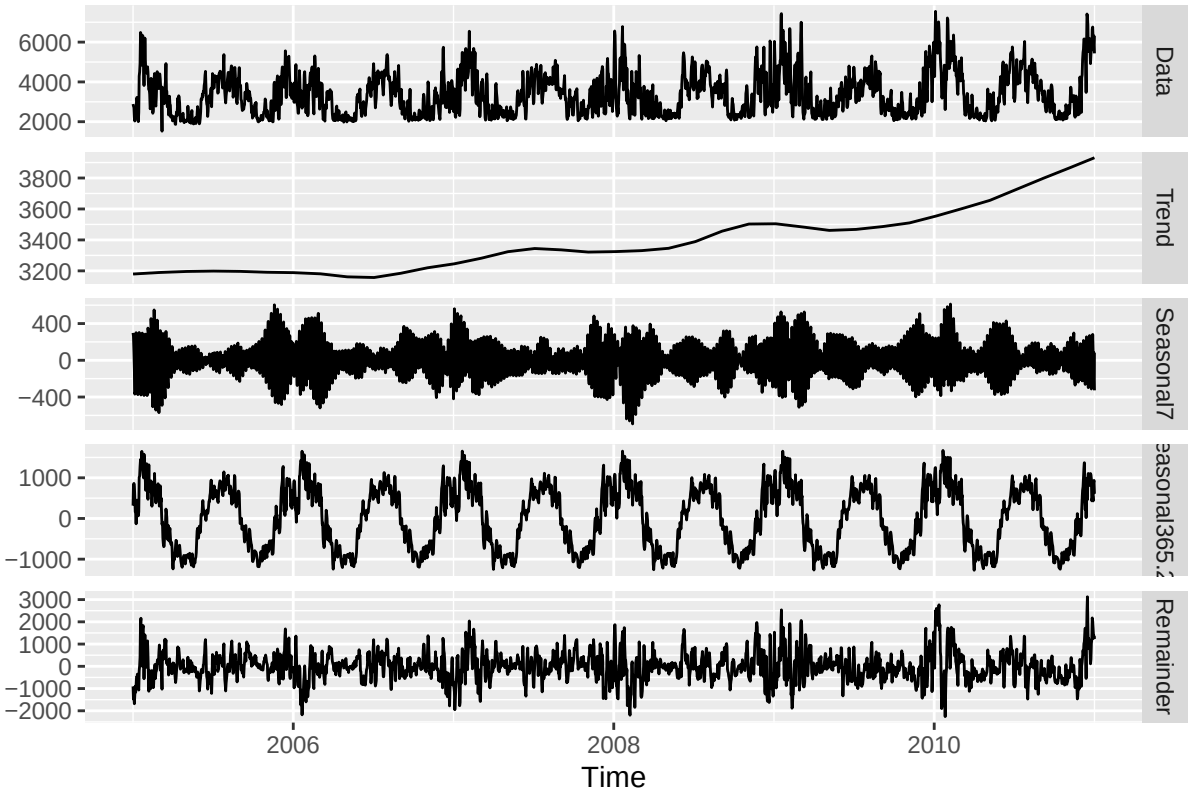
Note that I provided hourly data. You shoudl take teh **average** of the 24 hours to obtain the daily load.

```
#wrangle demand data
load_processed<- load%>%
  mutate(Date = ymd(date))%>%
  mutate( Year = year(Date),
          Month = month(Date),
          Day = day(Date))%>%
  mutate(demand_daily = rowMeans(across(h1:h24)))%>%
  select(Date, Year, Month, Day, demand_daily)

load_processed$demand_daily<-na.approx(load_processed$demand_daily)

#wrangle temp data
temperature_processed<- temperature%>%
  mutate(Date = ymd(date))%>%
  mutate(hourly_mean_temp= rowMeans(across(t_ws1:t_ws28)))%>%
  select(Date, hourly_mean_temp)%>%
  group_by(Date)%>%
  summarise(mean(hourly_mean_temp))

#wrangle hourly demand
load_processed2<- load%>%
  mutate(Date = ymd(date))%>%
  mutate(Year = year(Date),
  Month = month(Date),
  Day = day(Date))%>%
  #mutate(column = (1:2191))%>%
  select(Year, Month, Day,h1:h24)

load_longer<- load_processed2 %>%
  pivot_longer(cols = !Year:Day, names_to = "hour", values_to = "demand")

load_longer<-load_longer%>%
  mutate(Date = as.Date(paste(Year, Month, Day, sep = "-")))%>%
  select(Date, Year, Month, Day, demand)
```

CREATE A TIME SERIES OBJECT

After you process your dataset use the `msts()` function to create a time series object. You need to use `msts()` instead of `ts()` because your daily data will have more than one seasonal component.

```
#Daily time series
ts_daily_demand<- msts(load_processed$demand_daily,
                       seasonal.periods =c(7,365.25),
                       start=c(2005,01,01))

ts_daily_demand%>% mstl()%>% autoplot()
```



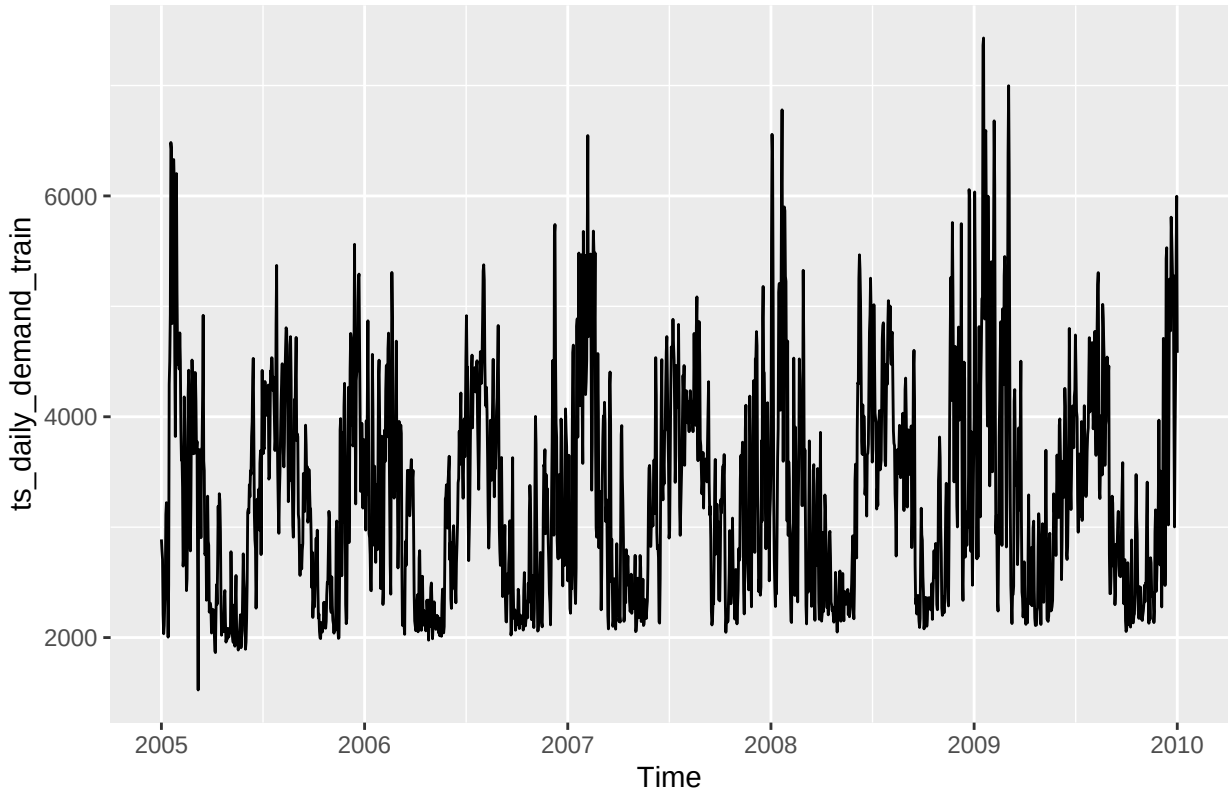

```
ts_temperature<- ts(temperature_processed$`mean(hourly_mean_temp)`, frequency = 365, start = c(2005,01,
```

```
#Hourly Time Series
ts_hourly_demand<- msts(load_longer$demand,
                        seasonal.periods =c(24,168,8766),
                        start=c(2005,01,01))
```

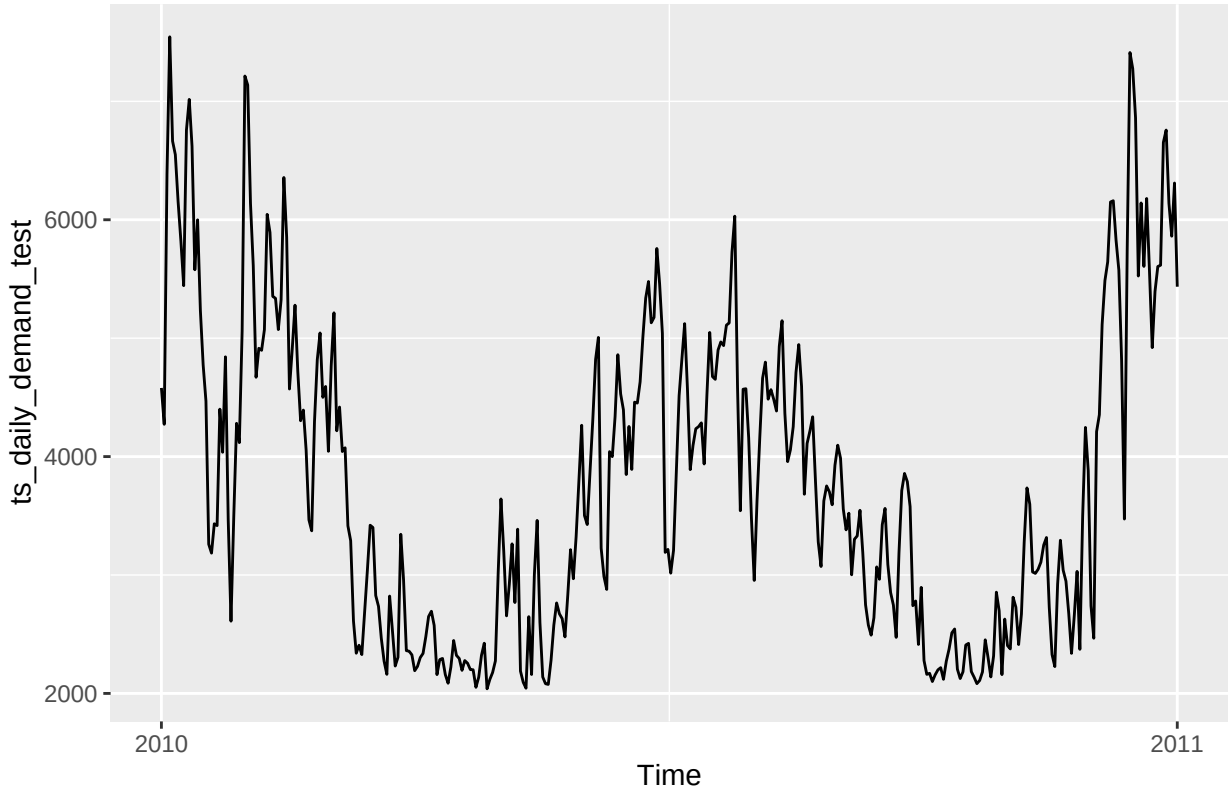
FIT MODELS TO YOUR DATA

Fit models to your dataset considering the period Jan 1st 2005 to Dec 31st 2009.

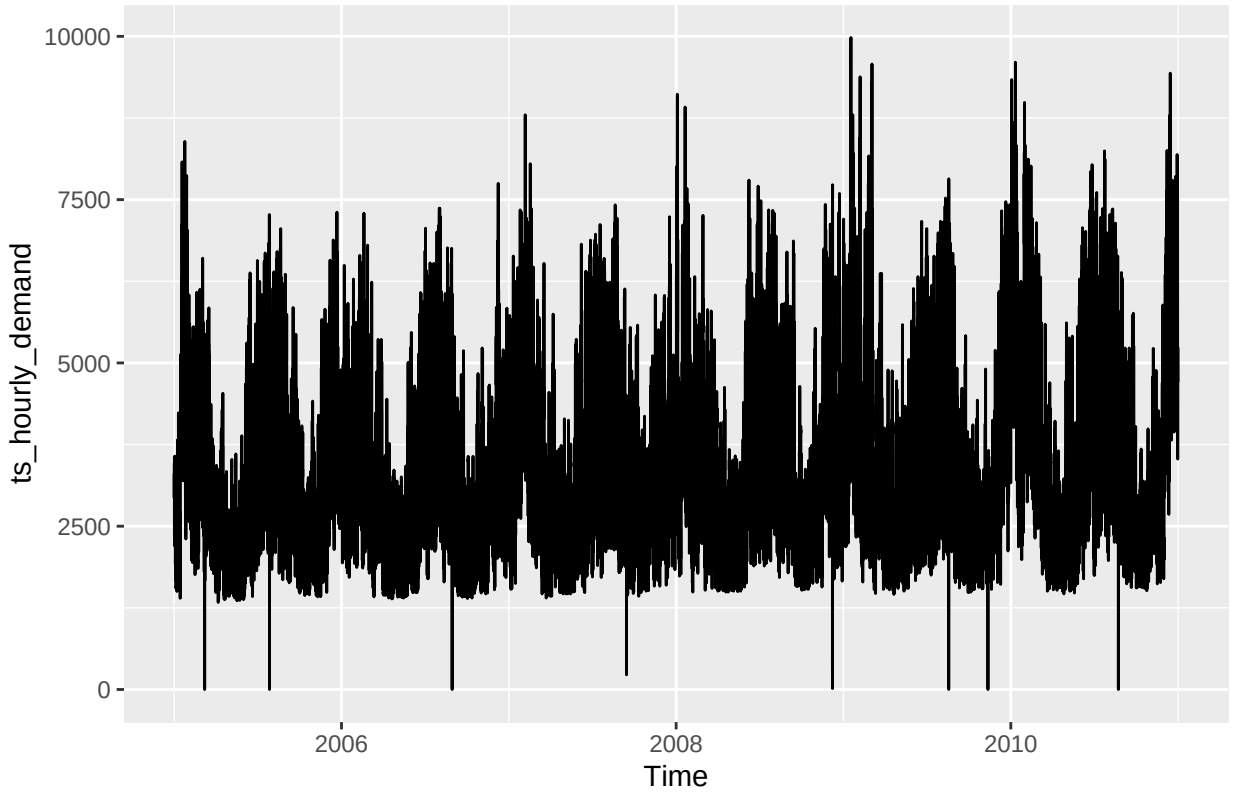
```
#create a subset for training purpose
n_for = 365
ts_daily_demand_train <- subset(ts_daily_demand,
                                end = length(ts_daily_demand)-n_for)

#create a subset for testing purpose
ts_daily_demand_test <- subset(ts_daily_demand,
                                start = length(ts_daily_demand)-n_for)

autoplot(ts_daily_demand_train)
```



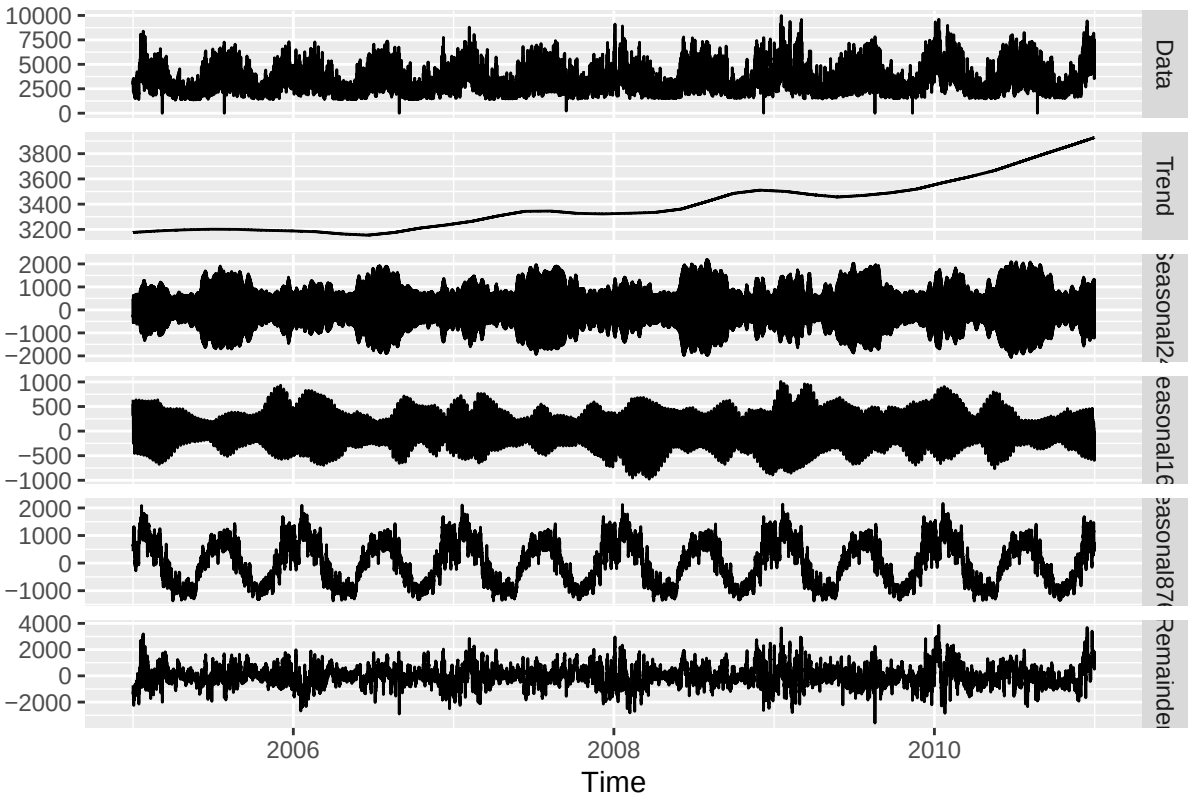

```
autoplot(ts_daily_demand_test)
```



```
autoplot(ts_hourly_demand)
```




```
ts_hourly_demand %>% mstl() %>%
  autoplot()
```

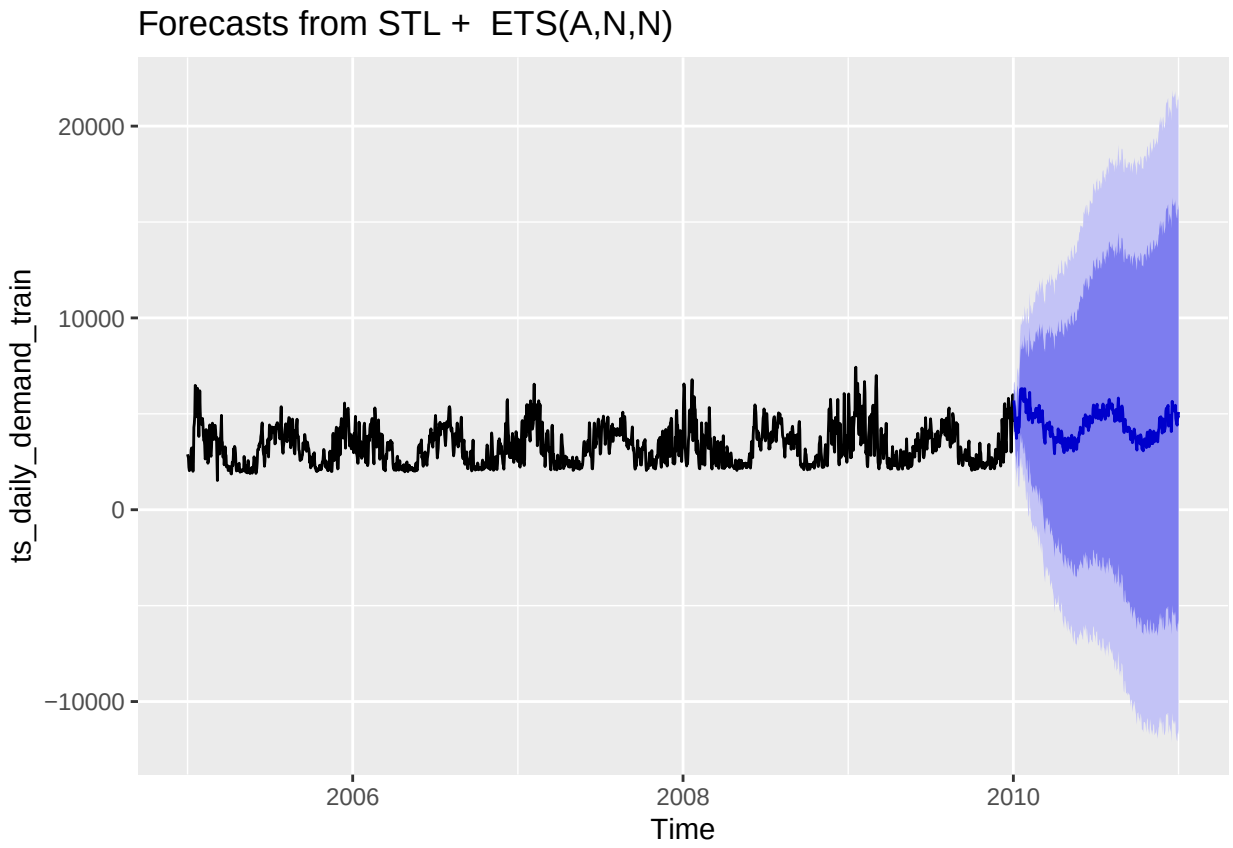


FORECAST DAILY DEMAND FOR 2010

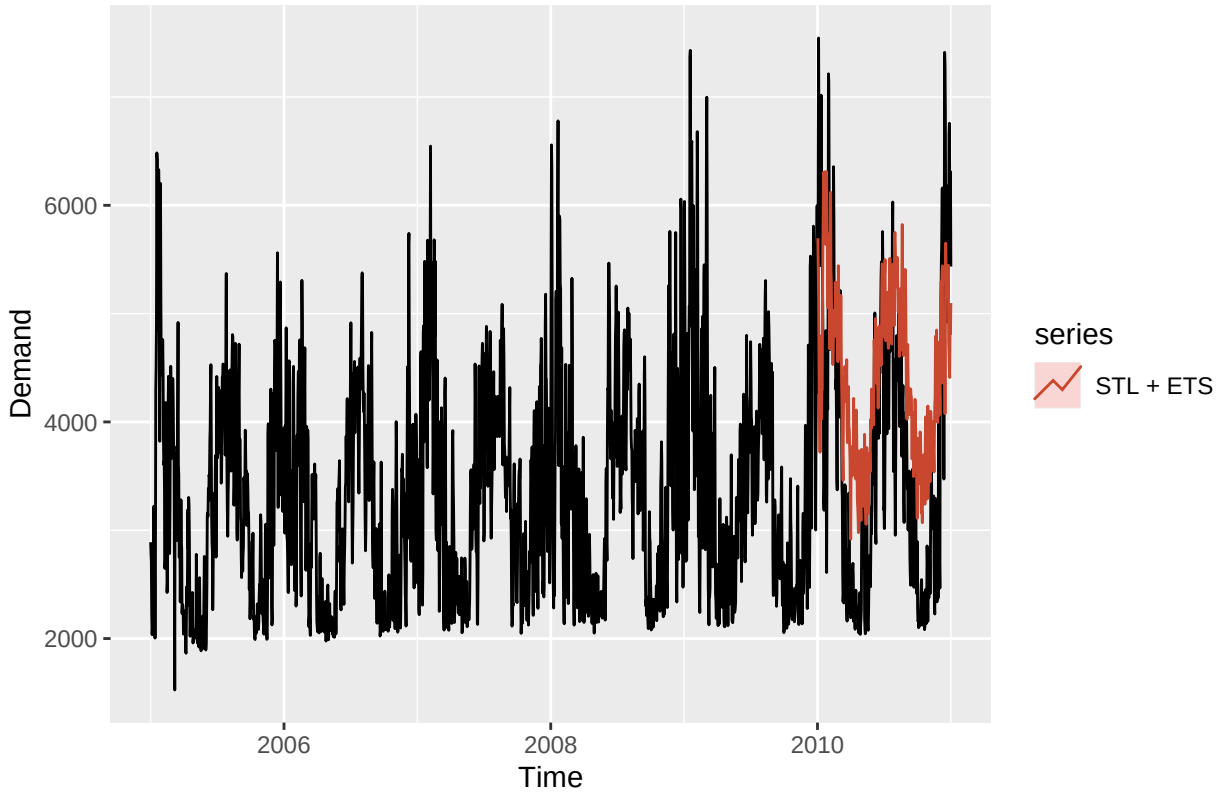
Using the models from previous section, forecast daily demand for the period Jan 1st 2010 to Feb 28 2010. Based on the models you developed which model(s) is(are) generating good forecast?

Model 1: STL + ETS

```
#Fit and forecast STL + ETS model to data
ETS_fit <-  stlf(ts_daily_demand_train,h=365)

#Plot foresting results
autoplot(ETS_fit)
```

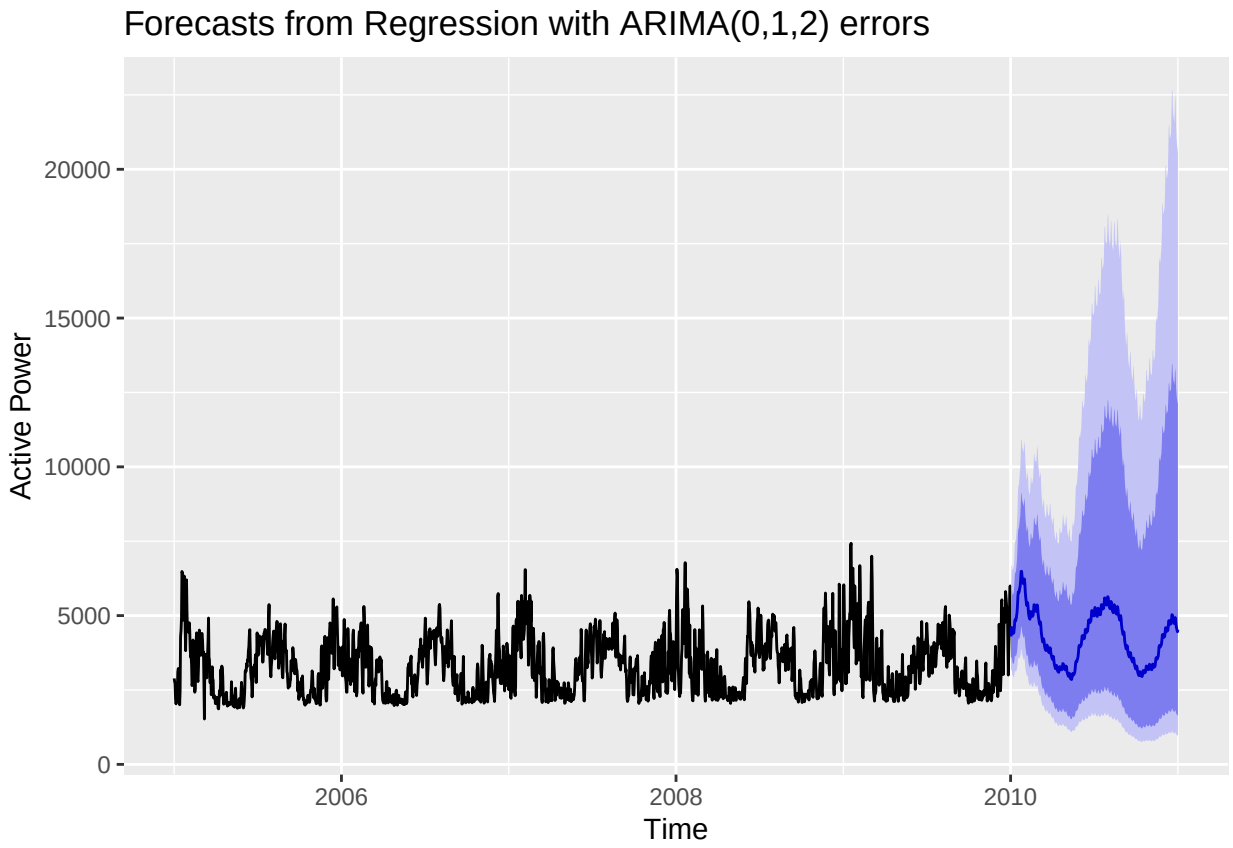



```
#Plot model + observed data
autoplot(ts_daily_demand) +
  autolayer(ETS_fit, series="STL + ETS",PI=FALSE)+
  ylab("Demand")
```

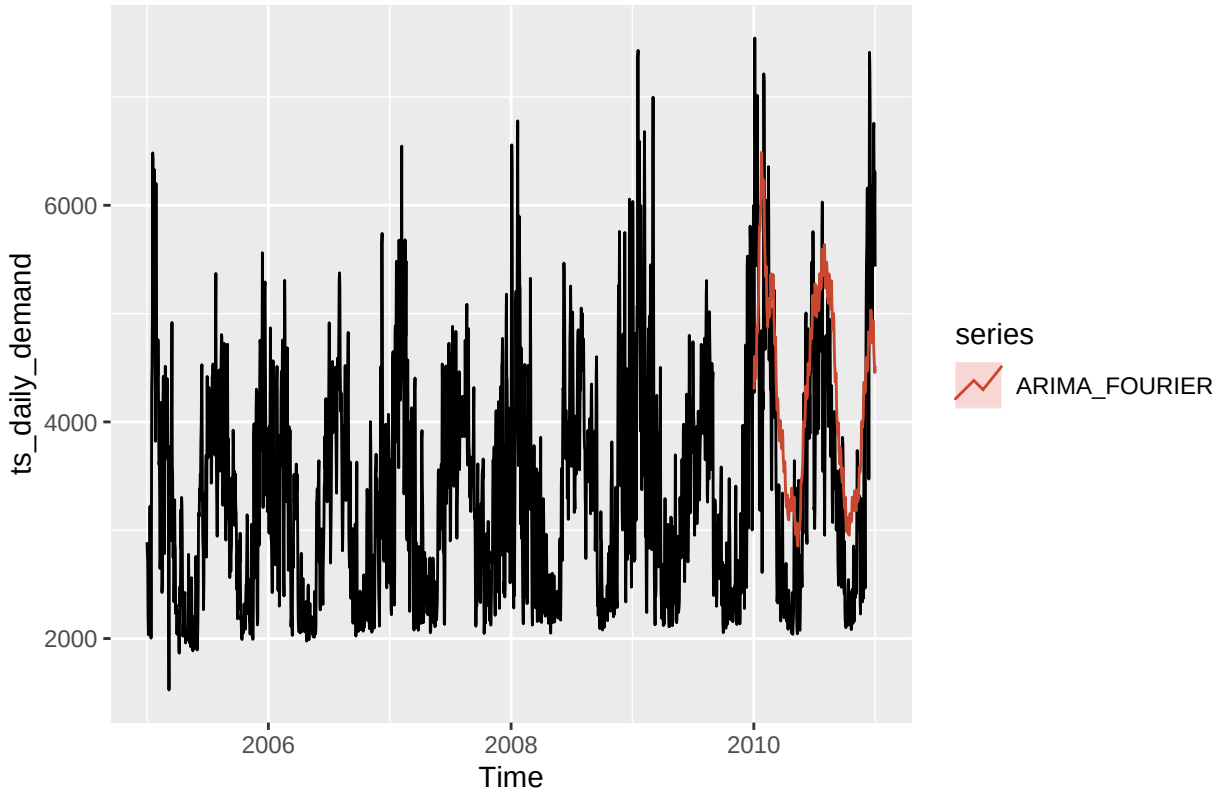


Model 2: ARIMA + FOURIER terms

```
#Fit arima model with fourier terms as exogenous regressors
ARIMA_Four_fit <- auto.arima(ts_daily_demand_train,
                             seasonal=FALSE,
                             lambda=0,
                             xreg=fourier(ts_daily_demand_train,
                                          K=c(2,12))
                             )

#Forecast with ARIMA fit
#also need to specify h for fourier terms
ARIMA_Four_for <- forecast(ARIMA_Four_fit,
                           xreg=fourier(ts_daily_demand_train,
                                        K=c(2,12),
                                        h=365),
                           h=365
                           )

#Plot foresting results
autoplot(ARIMA_Four_for) + ylab("Active Power")
```



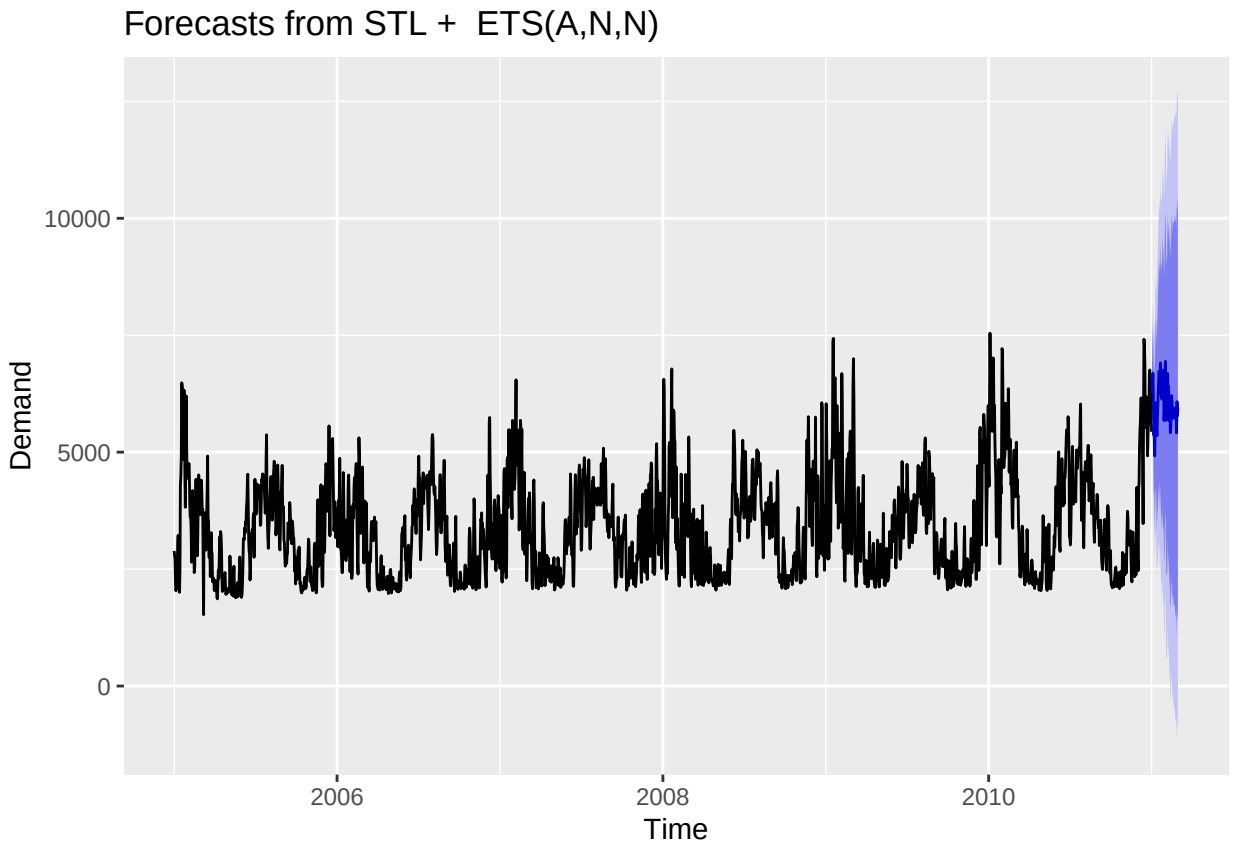

```
#Plot model + observed data
autoplot(ts_daily_demand) +
  autolayer(ARIMA_Four_for, series="ARIMA_FOURIER",PI=FALSE)
```



FORECAST DAILY DEMAND FOR 2011

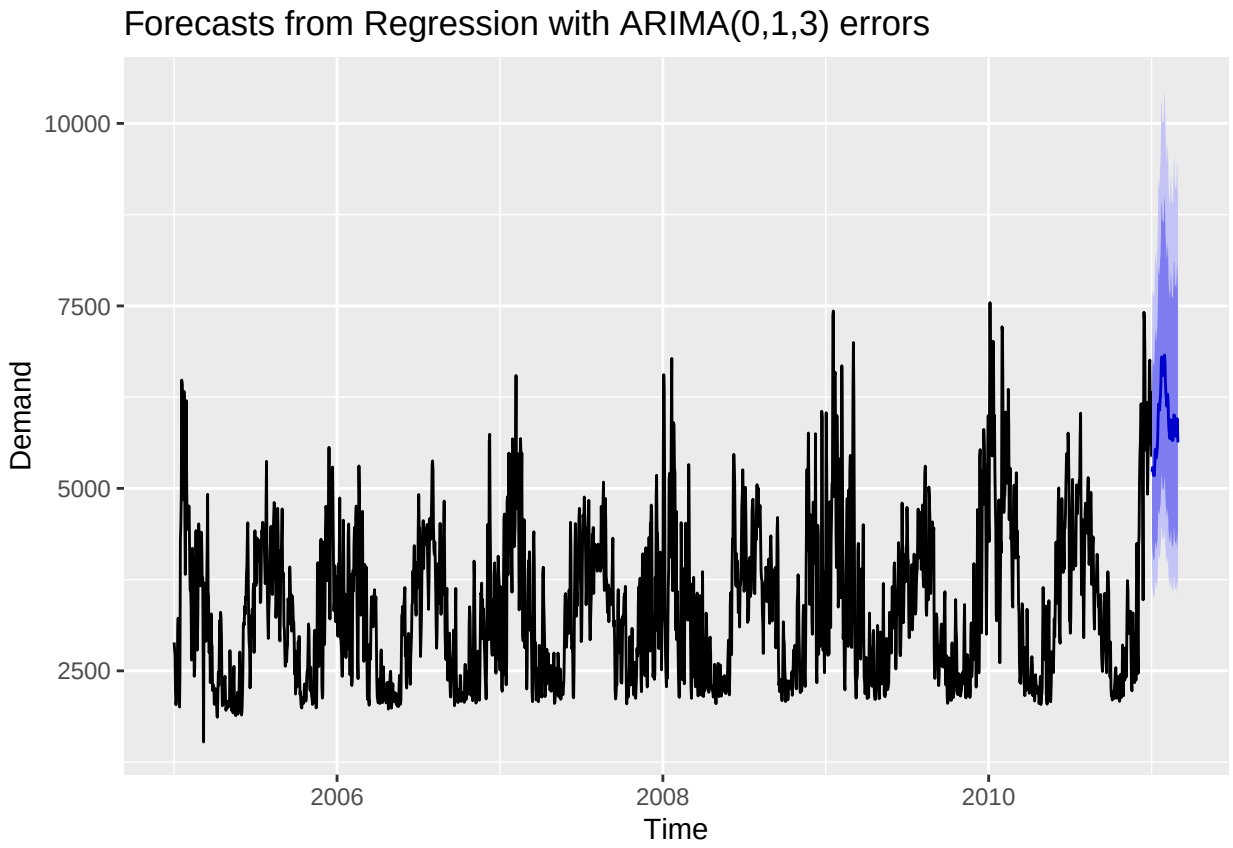
Just for the good model(s) you will **re-run** the model but now using the entire dataset (2005-2010) for model fitting and forecast Jan 1st 2011 to Feb 28 2011. ###ETS

```
#Fit and forecast STL + ETS model to data
ETS_fit2 <-  stlf(ts_daily_demand,h=59)

#Plot foresting results
autoplot(ETS_fit2) + ylab("Demand")
```

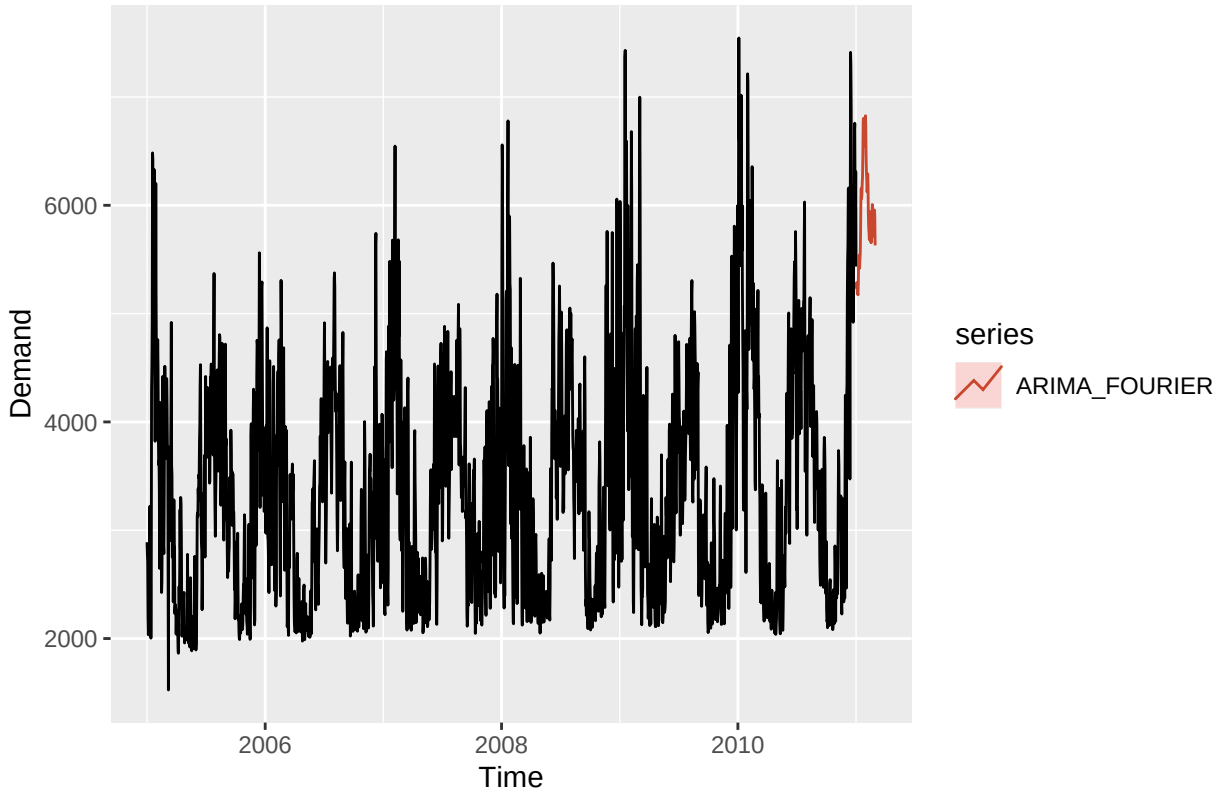


###Arima with fourier

```
#Forecast with ARIMA fit
ARIMA_Four_fit2 <- auto.arima(ts_daily_demand,
                              seasonal=FALSE,
                              lambda=0,
                              xreg=fourier(ts_daily_demand,
                                           K=c(2,12))
                              )
#also need to specify h for fourier terms
ARIMA_Four_for_jantofeb2 <- forecast(ARIMA_Four_fit2,
                             xreg=fourier(ts_daily_demand,
                                          K=c(2,12),
                                          h=59),
                             h=59
                             )


#Plot foresting results
autoplot(ARIMA_Four_for_jantofeb2) + ylab("Demand")
```

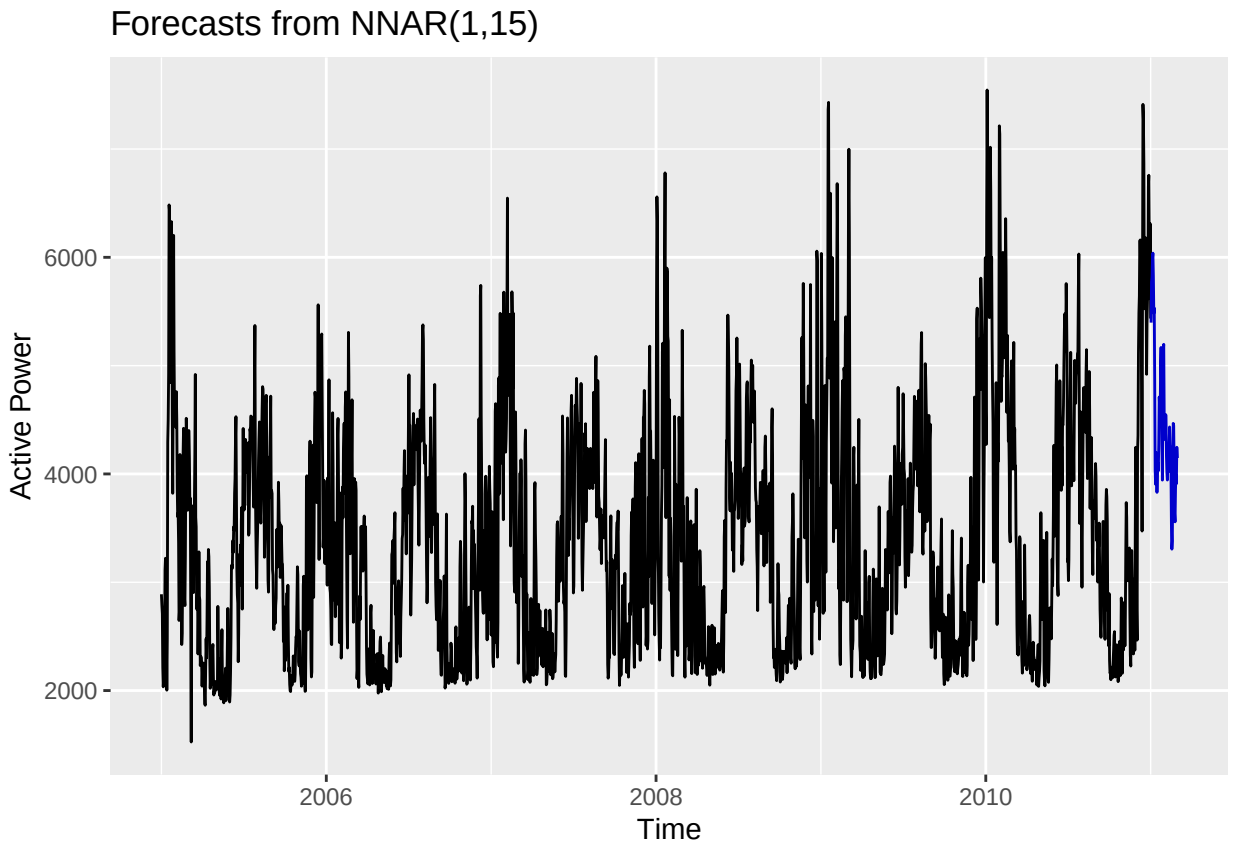



```
#Plot model + observed data
autoplot(ts_daily_demand) +
  autolayer(ARIMA_Four_for_jantofeb2, series="ARIMA_FOURIER",PI=FALSE) +
  ylab("Demand")
```



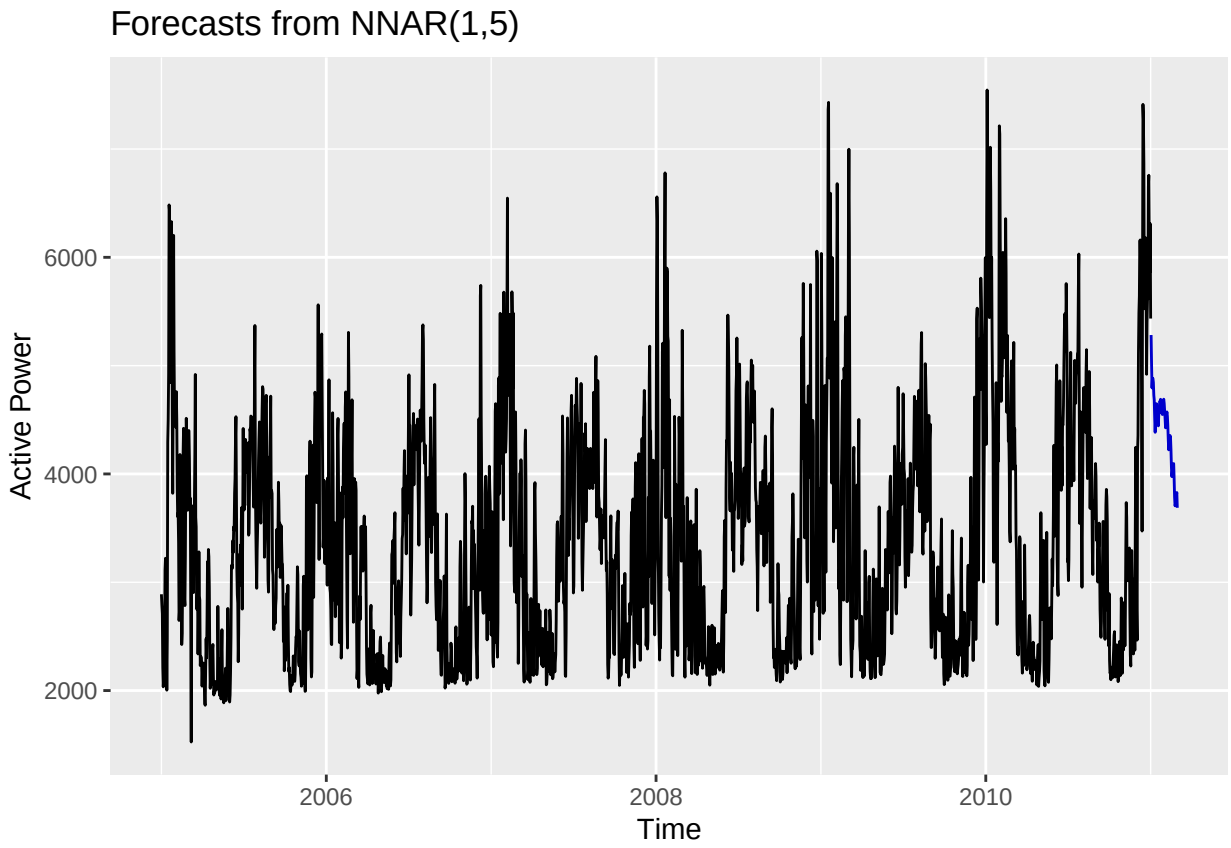
###Neural Network 1 ##for the Neural Network, we are changing the K values.

```
NN_fit <- nnetar(ts_daily_demand,p=1,P=0,xreg=fourier(ts_daily_demand, K=c(2,12)))

#NN_for <- forecast(NN_fit, h=365)
NN_for <- forecast(NN_fit, h=59,xreg=fourier(ts_daily_demand,
                                             K=c(2,12),h=59))

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



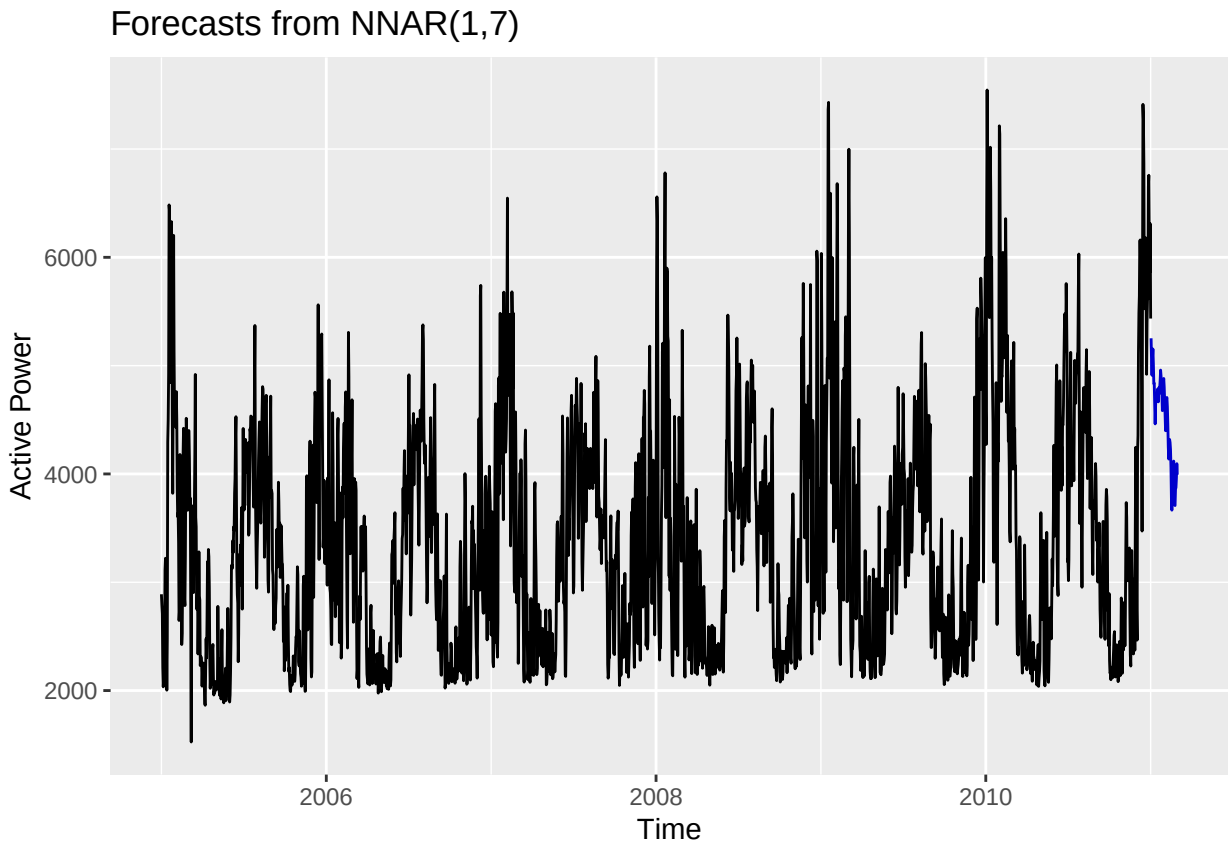
###Neural Network 2

```
NN_fit2 <- nnetar(ts_daily_demand,p=1,P=0,xreg=fourier(ts_daily_demand, K=c(2,2)))

#NN_for <- forecast(NN_fit, h=365)
NN_for2 <- forecast(NN_fit2, h=59,xreg=fourier(ts_daily_demand,
                                               K=c(2,2),h=59))

#Plot foresting results
autoplot(NN_for2) +
  ylab("Active Power")
```



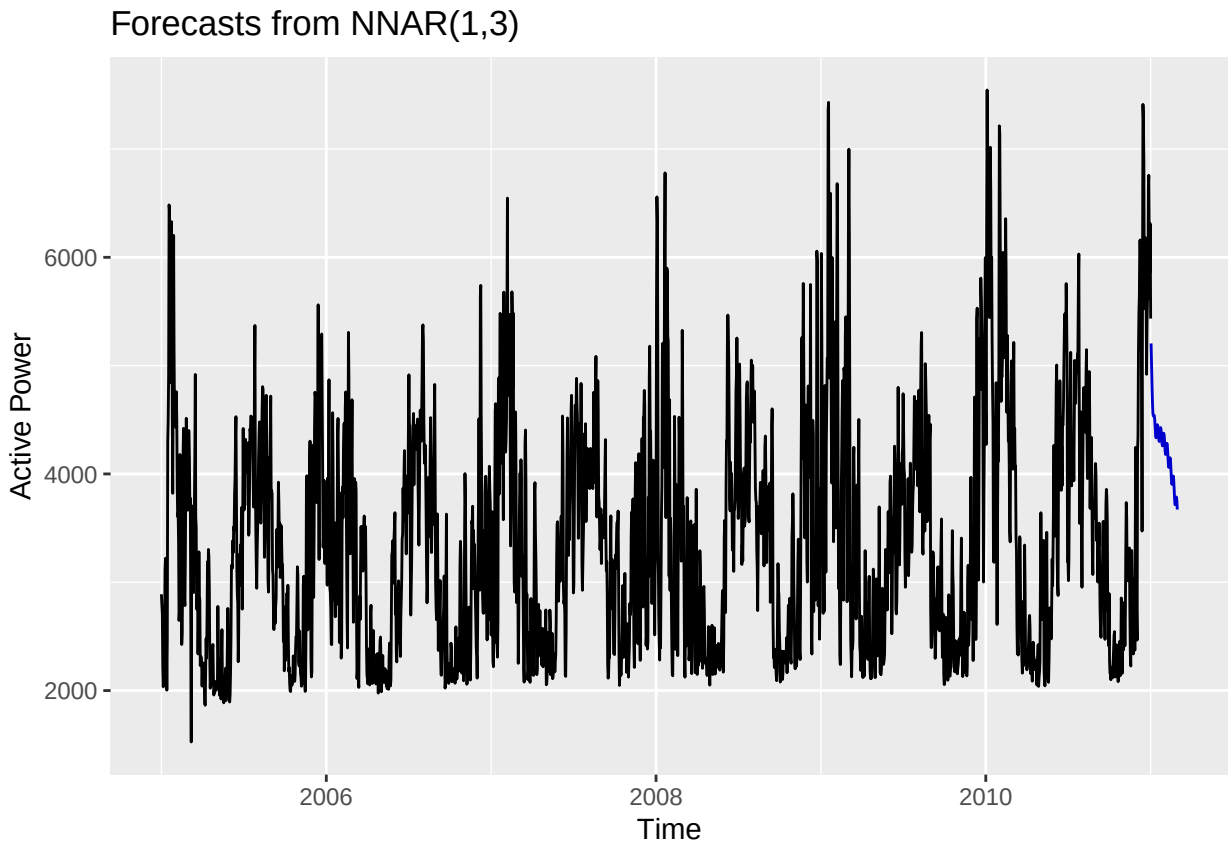
###Neural Network 3

```
NN_fit3 <- nnetar(ts_daily_demand,p=1,P=0,xreg=fourier(ts_daily_demand, K=c(2,4)))

#NN_for <- forecast(NN_fit, h=365)
NN_for3 <- forecast(NN_fit3, h=59,xreg=fourier(ts_daily_demand,
                                               K=c(2,4),h=59))

#Plot foresting results
autoplot(NN_for3) +
  ylab("Active Power")
```



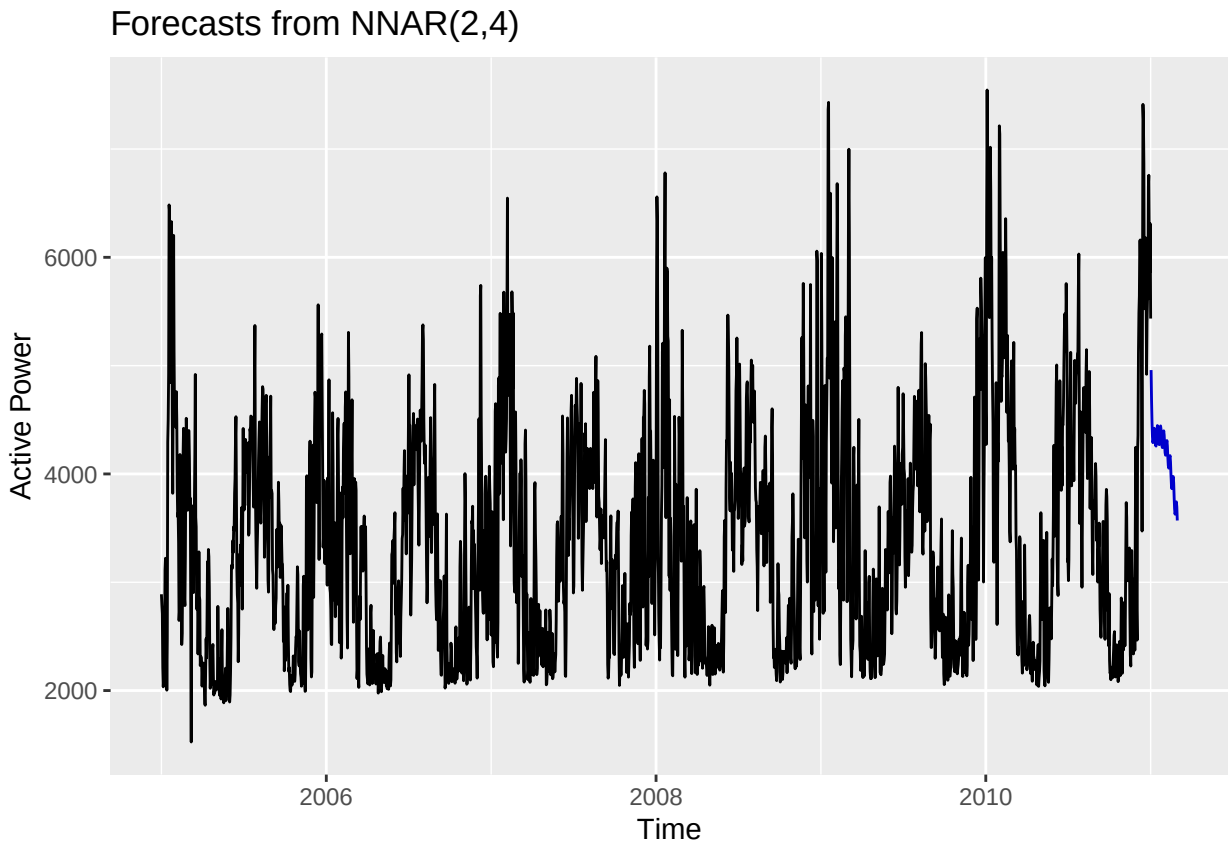
###Neural Network 4 (2nd best model)

```
NN_fit4 <- nnetar(ts_daily_demand,p=1,P=0,xreg=fourier(ts_daily_demand, K=c(1,1)))

#NN_for <- forecast(NN_fit, h=365)
NN_for4 <- forecast(NN_fit4, h=59,xreg=fourier(ts_daily_demand,
                                          K=c(1,1),h=59))

#Plot foresting results
autoplot(NN_for4) +
  ylab("Active Power")
```



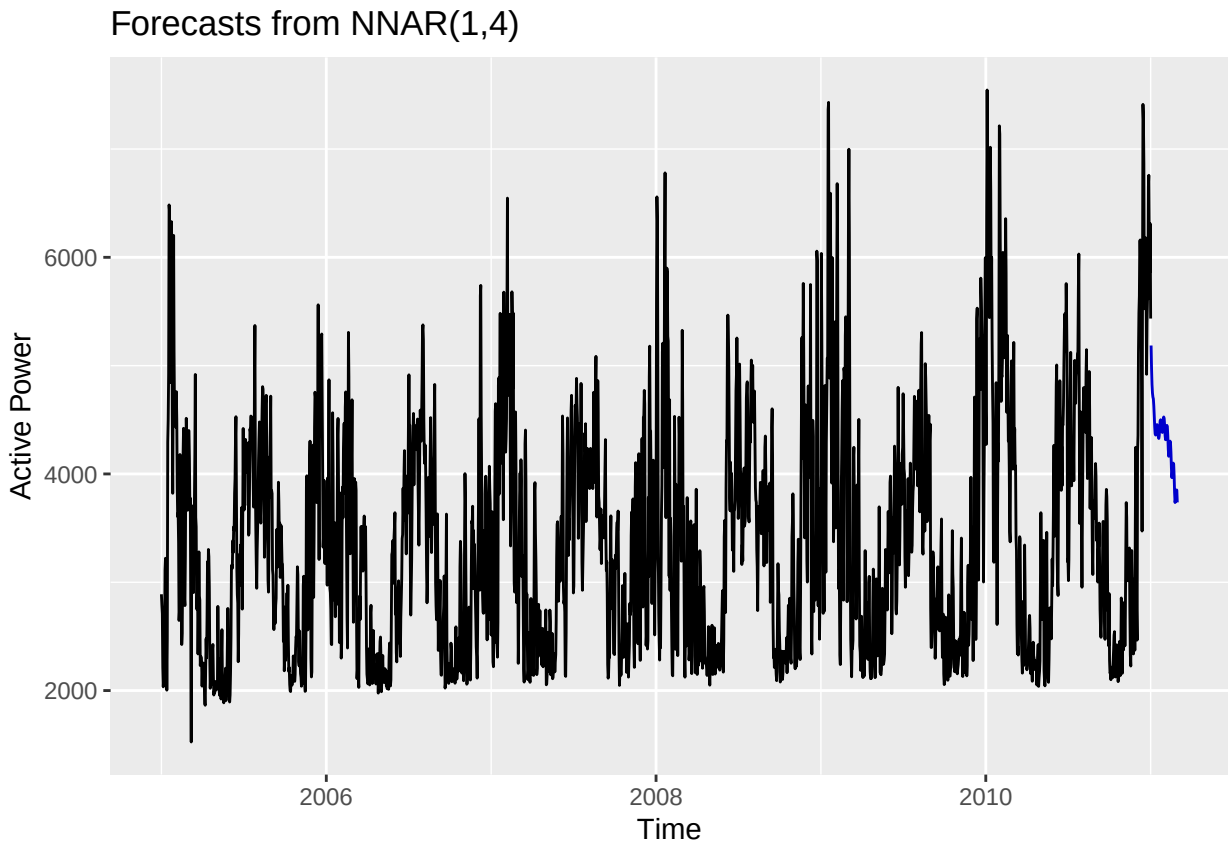
###Neural Network 7 changed p value

```
NN_fit7 <- nnetar(ts_daily_demand,p=2,P=0,xreg=fourier(ts_daily_demand, K=c(1,1)))

#NN_for <- forecast(NN_fit, h=365)
NN_for7 <- forecast(NN_fit7, h=59,xreg=fourier(ts_daily_demand,
                                          K=c(1,1),h=59))

#Plot foresting results
autoplot(NN_for7) +
  ylab("Active Power")
```



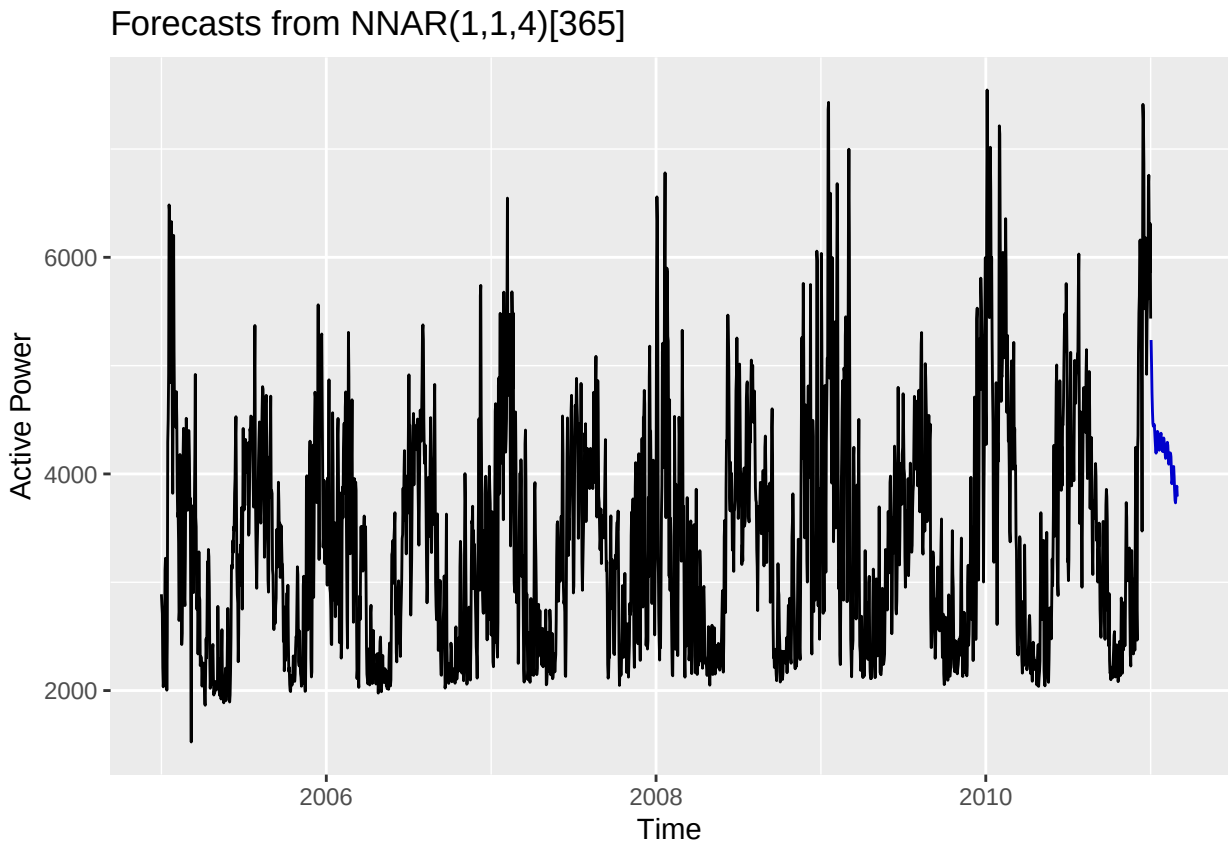
###Neural Network 8 with different K value

```
NN_fit9 <- nnetar(ts_daily_demand,p=1,P=0,xreg=fourier(ts_daily_demand, K=c(1,2)))

#NN_for <- forecast(NN_fit, h=365)
NN_for9 <- forecast(NN_fit9, h=59,xreg=fourier(ts_daily_demand,
                                          K=c(1,2),h=59))

#Plot foresting results
autoplot(NN_for9) +
  ylab("Active Power")
```



###Neural Network 9 with different P value

```
#You can play with the different values for p and P, you can also use xreg with Fourier term to model t

#NN_fit <- nnetar(ts_act_power_daily_train,p=1,P=1)
NN_fit10 <- nnetar(ts_daily_demand,p=1,P=1,xreg=fourier(ts_daily_demand, K=c(1,1)))

#NN_for <- forecast(NN_fit, h=365)
NN_for10 <- forecast(NN_fit10, h=59,xreg=fourier(ts_daily_demand,
                                          K=c(1,1),h=59))

#Plot foresting results
autoplot(NN_for10) +
  ylab("Active Power")
```

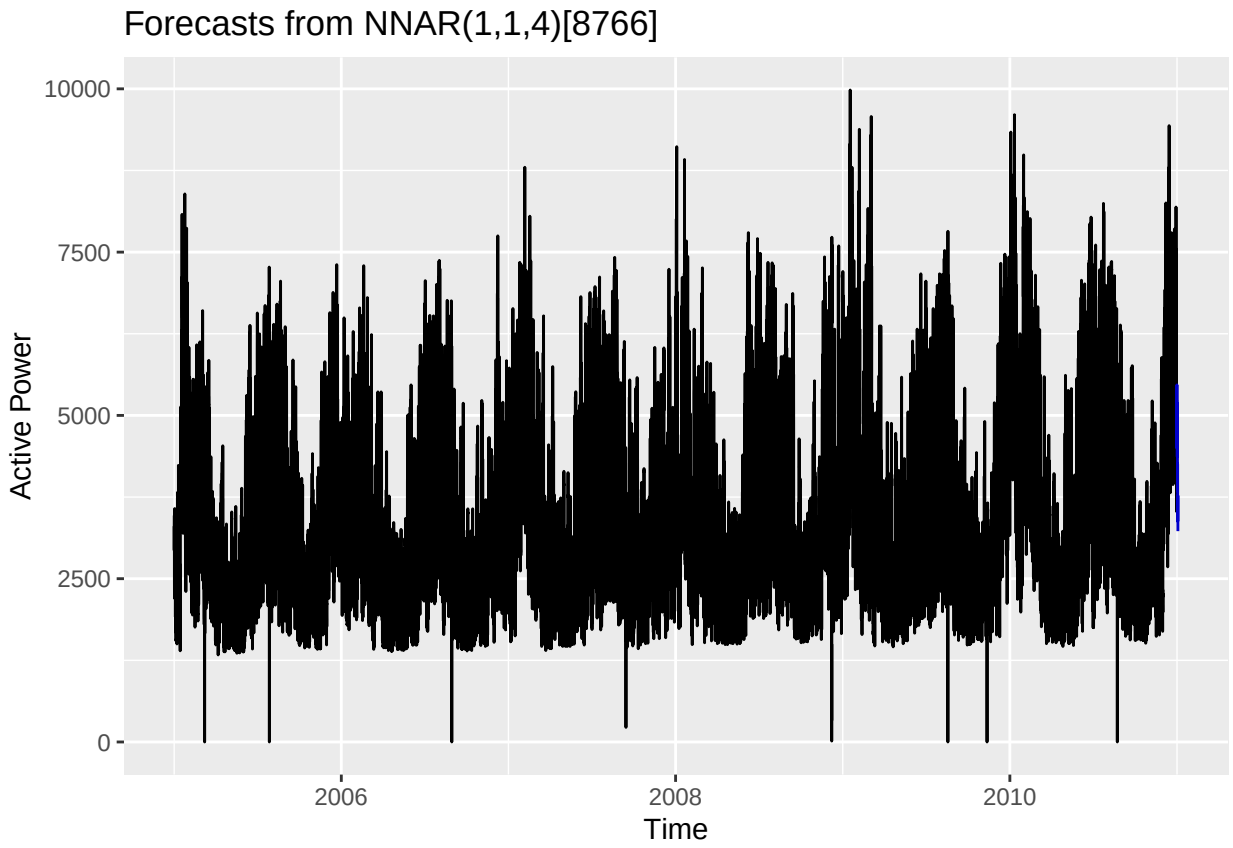


##Neural Network 10 with hourly data! ##winner winner chicken dinner

```
#You can play with the different values for p and P, you can also use xreg with Fourier term to model t

#NN_fit <- nnetar(ts_act_power_daily_train,p=1,P=1)
NN_fit11 <- nnetar(ts_hourly_demand,p=1,P=1,xreg=fourier(ts_hourly_demand, K=c(1,1,1)))
```

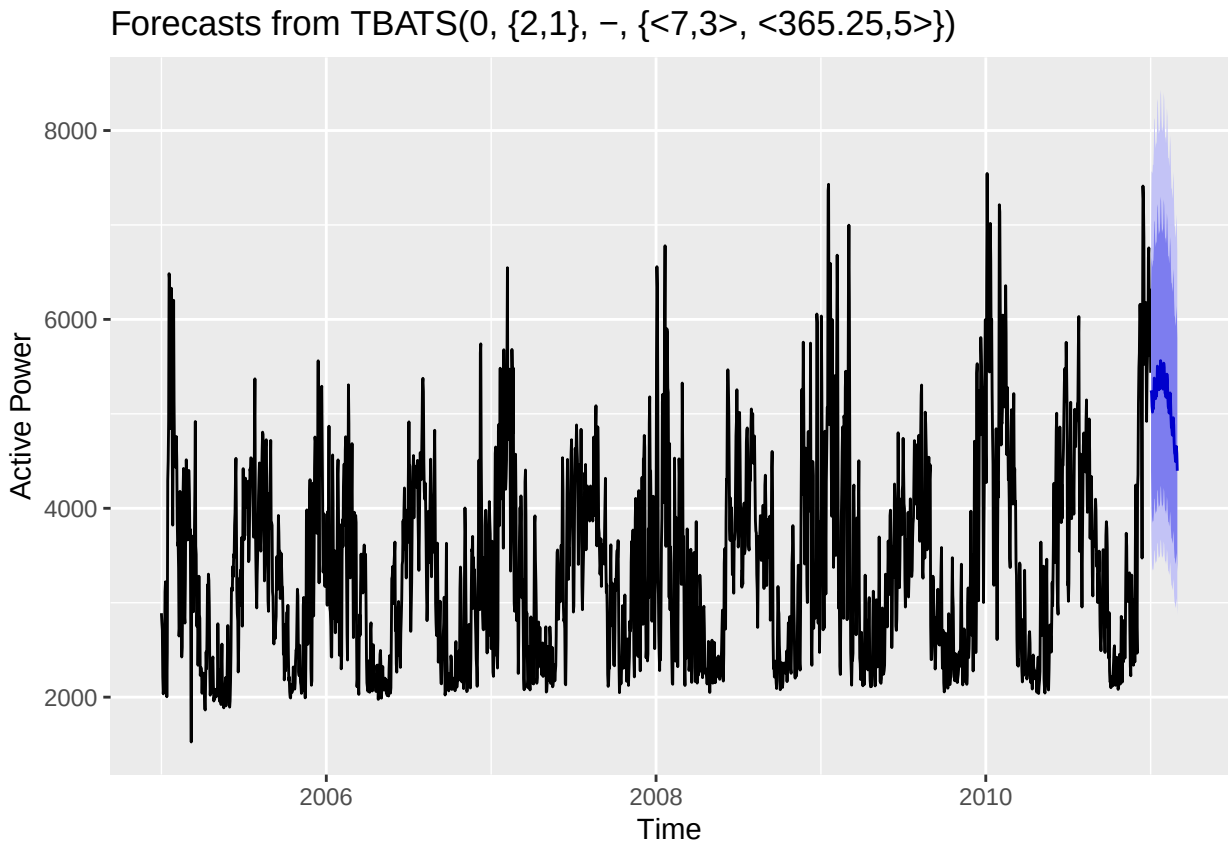
```
## Warning in nnetar(ts_hourly_demand, p = 1, P = 1, xreg =
## fourier(ts_hourly_demand, : Missing values in x, omitting rows
```

```
#NN_for <- forecast(NN_fit, h=365)
NN_for11 <- forecast(NN_fit11, h=59,xreg=fourier(ts_hourly_demand,
                                          K=c(1,1,1),h=59))

#Plot foresting results
autoplot(NN_for11) +
  ylab("Active Power")
```



###TBATS

```
# TBATS can take time to fit
TBATS_fit <- tbats(ts_daily_demand)

TBATS_for <- forecast(TBATS_fit, h=59)

#Plot foresting results
autoplot(TBATS_for) +
  ylab("Active Power")
```



CREATE AN EXCEL FILE WITH FORECAST

Look at the excel file in your Output folder name "submission_template.csv". You will need to create your own output file with forecast for January 2011. Your file needs to be in the format of the submission template. If your forecast is a probability distribution function, consider the mean to be the point forecast.

##model 1

```
janfeb<- read_excel("../Competition/Output/submission_template.xlsx")
janfeb<- janfeb%>%
  mutate(load=as.matrix(ETS_fit2$mean))%>%
  select(date, load)

write.csv(janfeb, file = "../Competition/Output/model1.csv", row.names = FALSE)
```

##model 2

```
janfeb2<- read_excel("../Competition/Output/submission_template.xlsx")
janfeb2<- janfeb%>%
  mutate(load=as.matrix(ARIMA_Four_for_jantofeb2$mean))%>%
  select(date, load)

write.csv(janfeb2, file = "../Competition/Output/model2.csv", row.names = FALSE)
```

##model 3

```
janfeb3<- janfeb%>%
  mutate(load=as.matrix(NN_for$mean))%>%
  select(date, load)
```

```
write.csv(janfeb3, file = "../Competition/Output/model3.csv", row.names = FALSE)
```

##model 4

```
janfeb4<- janfeb%>%
  mutate(load=as.matrix(TBATS_for$mean))%>%
  select(date, load)

write.csv(janfeb4, file = "../Competition/Output/model4.csv", row.names = FALSE)
```

##model 5

```
model5<- cbind(janfeb4, janfeb3)

model5<- model5[,2:4]

model5<- model5%>%
  rename(TBATS = "load", date= "date", NN="load.1")

model5<-model5%>%
  select(date, TBATS, NN)%>%
  mutate(load=rowMeans(across(TBATS:NN)))%>%
  select(date, load)

write.csv(model5, file = "../Competition/Output/model5.csv", row.names = FALSE)
```

##model 6

```
janfeb5<- janfeb%>%
  mutate(load=as.matrix(NN_for2$mean))%>%
  select(date, load)

write.csv(janfeb5, file = "../Competition/Output/model6.csv", row.names = FALSE)
```

##model 7

```
janfeb6<- janfeb%>%
  mutate(load=as.matrix(NN_for3$mean))%>%
  select(date, load)

write.csv(janfeb6, file = "../Competition/Output/model7.csv", row.names = FALSE)
```

##model 9

```
janfeb7<- janfeb%>%
  mutate(load=as.matrix(NN_for4$mean))%>%
  select(date, load)

write.csv(janfeb7, file = "../Competition/Output/model9.csv", row.names = FALSE)
```

##model 10

```
#janfeb8<- janfeb%>%
  #mutate(load=NN_For)%>%
  #select(date, load)

#write.csv(janfeb8, file = "./Competition/Output/model10.csv", row.names = FALSE)
```

##model 11

```
janfeb9<- janfeb%>%
  mutate(load=as.matrix(NN_for7$mean))%>%
  select(date, load)

write.csv(janfeb9, file = "../Competition/Output/model11.csv", row.names = FALSE)
```

##model 12

```
#janfeb10<- janfeb%>%
  #mutate(load=as.matrix(NN_for8$mean))%>%
  #select(date, load)

#write.csv(janfeb10, file = "../Competition/Output/model12.csv", row.names = FALSE)
```

##model 13

```
janfeb11<- janfeb%>%
  mutate(load=as.matrix(NN_for9$mean))%>%
  select(date, load)

write.csv(janfeb11, file = "../Competition/Output/model13.csv", row.names = FALSE)
```

##model 14

```
janfeb12<- janfeb%>%
  mutate(load=as.matrix(NN_for10$mean))%>%
  select(date, load)

write.csv(janfeb12, file = "../Competition/Output/model14.csv", row.names = FALSE)
```

##model 15

```
model15<- cbind(janfeb9, janfeb7)

model15<- model15[,2:4]

model15<- model15%>%
  rename(NN1 = "load", date= "date", NN2="load.1")

model15<-model15%>%
  select(date, NN1, NN2)%>%
  mutate(load=rowMeans(across(NN1:NN2)))%>%
  select(date, load)

write.csv(model15, file = "../Competition/Output/model15.csv", row.names = FALSE)
```

##model 16

```
janfeb13<- janfeb%>%
  mutate(load=as.matrix(NN_for11$mean))%>%
  select(date, load)

write.csv(janfeb13, file = "../Competition/Output/model17.csv", row.names = FALSE)
```

LOAD TEMPLATE IN KAGGLE

I created a kaggle competition for this assignment. You will need to enter the competition using this invitation.

Once you enter the competition you should be to visualize and submit your group's solution using this [link.][https://www.kaggle.com/competitions/tsa-s23-competition/]

COMPLETE YOUR PROJECT REPORT

For the project report you only need to organize your current Rmd file. Make sure you follow the guidelines and you provide a link to you Github repository.

1. Write in scientific style, not narrative style
2. Global options for R chunks should be set so that only relevant output is displayed. Turn on/off messages and warnings when applicable to avoid unnecessary outputs on the pdf.
3. Make sure your final knitted PDF looks professional. Format tables, size figures, chapters, etc.
4. Make sure the PDF file has the file name "Lastname1Lastname2_ENV790_A09_Competition.pdf" and submit it to Sakai under A09. You will only submit your PDF file.

GRADING RUBRIC

You will be graded based on how much time and effort you put into the competition and your ability to fit a model to the data set. More specifically I will look into:

1. number of commitments to Github repo, this item will show how the team interacted and how much you worked on the project;
2. number of submissions to Kaggle platform, this will show how many models you tried and it's also an indication of how much effort the team put into the project;
3. ability to beat the vanilla/benchmark model, this will show your forecasting skills.

The team that is leading the board when the competition ends will get extra points, but they still need to get good scores on 1 and 2.