Forecasting Competition

Emma Kaufman and Jaimie Wargo

2024-04-26

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

```
load <- read_excel(here("Data_TOPOST","load.xlsx"))
relative_humidity <- read_excel(here("Data_TOPOST","relative_humidity.xlsx"))
temperature <- read_excel(here("Data_TOPOST", "temperature.xlsx"))
```

WRANGLE/PROCESS THE DATASET

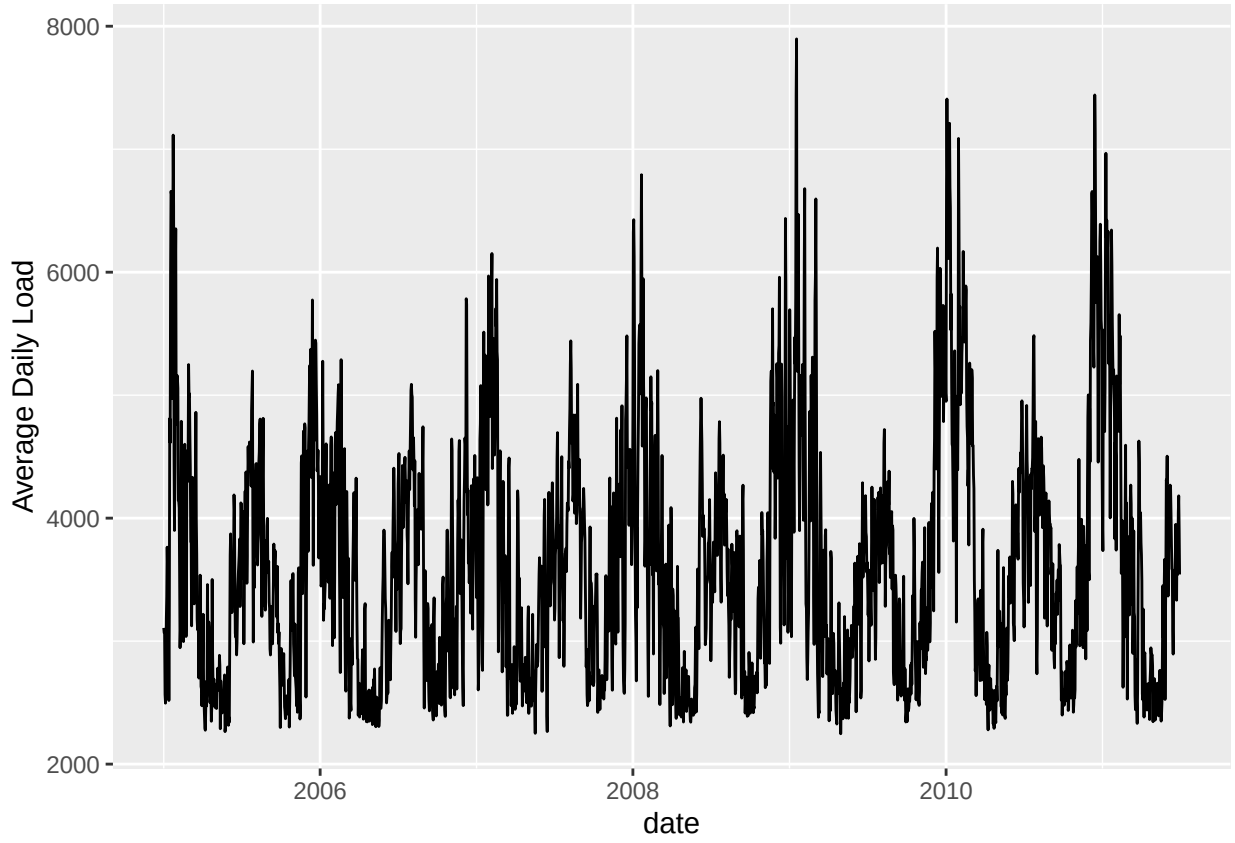
```
#start with wrangling load data
load_all <- load %>%
  pivot_longer(cols = h1:h24) %>%
  rename(Hour = name,
         Load= value) %>%
  mutate(Hour= as.numeric(gsub('h','', Hour))-1) %>%
  mutate(date= ymd(date)) %>%
  mutate(Day = day(date),
         Month= month(date),
         Year= year(date)) %>%
  select(meter_id, date, Year, Month, Day, Hour, Load)

#check for NAs
summary(load_all$Load)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##       0    2568    3352    3629    4520   10592       7
```

```r
#Creating a data frame with daily observations
load_daily <- load_all %>%
  filter( !is.na(Load) ) %>%
  group_by(meter_id, date, Year, Month, Day) %>% # here we left column with hour out to calculate daily
  summarise( Daily_mean_load = mean(Load)) #take the mean for the day
```

```
## `summarise()` has grouped output by 'meter_id', 'date', 'Year', 'Month'. You
## can override using the `.groups` argument.
```

```r
ggplot(load_daily, aes(x=date,y=Daily_mean_load)) +
  geom_line() +
  ylab("Average Daily Load")
```




```r
#check for NAs
summary(load_daily$Daily_mean_load)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2247    2798    3506    3629    4211    7897
```

```r
#temperature and relative humidity data
temperature_all <- temperature %>%
  mutate(date= ymd(date)) %>%
  mutate(Day= day(date),
```

```
          Month= month(date),
          Year= year(date)) %>%
  mutate(hr= hr-1) %>%
  rename(hour= hr)

relative_humidity_all <- relative_humidity %>%
   mutate(date= ymd(date)) %>%
  mutate(Day= day(date),
          Month= month(date),
          Year= year(date)) %>%
  mutate(hr= hr-1) %>%
  rename(hour= hr)

#check for NA
summary(temperature_all)
```

```
##       date                hour          t_ws1            t_ws2
##  Min.   :2005-01-01   Min.   : 0.0   Min.   :  9.00   Min.   :  9.00
##  1st Qu.:2006-08-17   1st Qu.: 6.0   1st Qu.: 47.00   1st Qu.: 48.00
##  Median :2008-04-01   Median :12.0   Median : 63.00   Median : 63.00
##  Mean   :2008-03-31   Mean   :11.5   Mean   : 60.86   Mean   : 60.87
##  3rd Qu.:2009-11-14   3rd Qu.:18.0   3rd Qu.: 74.00   3rd Qu.: 74.00
##  Max.   :2011-06-30   Max.   :23.0   Max.   :104.00   Max.   :104.00
##  NA's   :1            NA's   :1      NA's   :1        NA's   :1
##      t_ws3           t_ws4            t_ws5            t_ws6
##  Min.   : 4.00   Min.   : 11.00   Min.   : 17.00   Min.   : 15.00
##  1st Qu.:42.00   1st Qu.: 48.00   1st Qu.: 51.00   1st Qu.: 48.00
##  Median :57.00   Median : 63.00   Median : 64.00   Median : 63.00
##  Mean   :55.18   Mean   : 60.83   Mean   : 62.83   Mean   : 60.79
##  3rd Qu.:68.00   3rd Qu.: 74.00   3rd Qu.: 75.00   3rd Qu.: 74.00
##  Max.   :93.00   Max.   :103.00   Max.   :102.00   Max.   :100.00
##  NA's   :1       NA's   :1        NA's   :1        NA's   :1
##      t_ws7            t_ws8            t_ws9           t_ws10
##  Min.   : 15.00   Min.   : 12.00   Min.   : 0.00   Min.   : 12.00
##  1st Qu.: 49.00   1st Qu.: 50.00   1st Qu.:41.00   1st Qu.: 49.00
##  Median : 64.00   Median : 64.00   Median :55.00   Median : 64.00
##  Mean   : 61.83   Mean   : 62.77   Mean   :53.66   Mean   : 61.94
##  3rd Qu.: 75.00   3rd Qu.: 75.00   3rd Qu.:66.00   3rd Qu.: 75.00
##  Max.   :100.00   Max.   :104.00   Max.   :93.00   Max.   :104.00
##  NA's   :1        NA's   :1        NA's   :1       NA's   :1
##      t_ws11           t_ws12           t_ws13           t_ws14
##  Min.   : 11.00   Min.   : 12.00   Min.   :  9.00   Min.   :  9.00
##  1st Qu.: 46.00   1st Qu.: 48.00   1st Qu.: 46.00   1st Qu.: 46.00
##  Median : 61.00   Median : 63.00   Median : 63.00   Median : 61.00
##  Mean   : 59.56   Mean   : 61.08   Mean   : 60.57   Mean   : 58.93
##  3rd Qu.: 73.00   3rd Qu.: 74.00   3rd Qu.: 73.00   3rd Qu.: 72.00
##  Max.   :100.00   Max.   :105.00   Max.   :102.00   Max.   :103.00
##  NA's   :1        NA's   :1        NA's   :1        NA's   :1
##      t_ws15           t_ws16          t_ws17           t_ws18
##  Min.   :  5.00   Min.   :21.0   Min.   :  9.00   Min.   : 14.00
##  1st Qu.: 45.00   1st Qu.:51.0   1st Qu.: 46.00   1st Qu.: 51.00
##  Median : 61.00   Median :64.0   Median : 62.00   Median : 66.00
##  Mean   : 58.96   Mean   :62.7   Mean   : 59.91   Mean   : 63.28
```

```
##  3rd Qu.: 73.00   3rd Qu.:75.0   3rd Qu.: 73.00   3rd Qu.: 76.00
##  Max.   :104.00   Max.   :93.0   Max.   :103.00   Max.   :100.00
##  NA's   :1        NA's   :1      NA's   :1        NA's   :1
##      t_ws19           t_ws20          t_ws21           t_ws22
##  Min.   :  9.0   Min.   :16.00   Min.   : 5.00   Min.   :  7.00
##  1st Qu.: 46.0   1st Qu.:52.00   1st Qu.:45.00   1st Qu.: 45.00
##  Median : 61.0   Median :65.00   Median :59.00   Median : 59.00
##  Mean   : 59.5   Mean   :63.15   Mean   :57.87   Mean   : 58.17
##  3rd Qu.: 73.0   3rd Qu.:76.00   3rd Qu.:72.00   3rd Qu.: 72.00
##  Max.   :102.0   Max.   :98.00   Max.   :99.00   Max.   :100.00
##  NA's   :1       NA's   :1       NA's   :1       NA's   :1
##      t_ws23           t_ws24           t_ws25           t_ws26
##  Min.   :12.00   Min.   : 16.00   Min.   :  9.00   Min.   : 10.00
##  1st Qu.:50.00   1st Qu.: 47.00   1st Qu.: 48.00   1st Qu.: 48.00
##  Median :65.00   Median : 62.00   Median : 64.00   Median : 64.00
##  Mean   :62.28   Mean   : 60.49   Mean   : 61.66   Mean   : 61.73
##  3rd Qu.:75.00   3rd Qu.: 74.00   3rd Qu.: 75.00   3rd Qu.: 75.00
##  Max.   :99.00   Max.   :104.00   Max.   :102.00   Max.   :104.00
##  NA's   :1       NA's   :1
##      t_ws27           t_ws28            Day             Month
##  Min.   : 11.0   Min.   :  9.00   Min.   : 1.00   Min.   : 1.000
##  1st Qu.: 47.0   1st Qu.: 46.00   1st Qu.: 8.00   1st Qu.: 3.000
##  Median : 63.0   Median : 62.00   Median :16.00   Median : 6.000
##  Mean   : 60.9   Mean   : 59.98   Mean   :15.72   Mean   : 6.294
##  3rd Qu.: 74.0   3rd Qu.: 73.00   3rd Qu.:23.00   3rd Qu.: 9.000
##  Max.   :104.0   Max.   :104.00   Max.   :31.00   Max.   :12.000
##                                   NA's   :1       NA's   :1
##       Year
##  Min.   :2005
##  1st Qu.:2006
##  Median :2008
##  Mean   :2008
##  3rd Qu.:2009
##  Max.   :2011
##  NA's   :1
```

```
summary(relative_humidity_all)
```

```
##       date                 hour           rh_ws1           rh_ws2
##  Min.   :2005-01-01   Min.   : 0.0   Min.   :  4.00   Min.   :  1.00
##  1st Qu.:2006-08-17   1st Qu.: 6.0   1st Qu.: 51.00   1st Qu.: 53.00
##  Median :2008-04-01   Median :12.0   Median : 73.00   Median : 74.00
##  Mean   :2008-03-31   Mean   :11.5   Mean   : 69.22   Mean   : 67.84
##  3rd Qu.:2009-11-14   3rd Qu.:18.0   3rd Qu.: 89.00   3rd Qu.: 87.00
##  Max.   :2011-06-30   Max.   :23.0   Max.   :100.00   Max.   :100.00
##      rh_ws3           rh_ws4           rh_ws5           rh_ws6
##  Min.   :  8.00   Min.   :  6.00   Min.   :  6.00   Min.   : 11.00
##  1st Qu.: 54.00   1st Qu.: 48.00   1st Qu.: 54.00   1st Qu.: 58.00
##  Median : 74.00   Median : 65.00   Median : 76.00   Median : 73.00
##  Mean   : 69.89   Mean   : 64.51   Mean   : 70.87   Mean   : 70.61
##  3rd Qu.: 89.00   3rd Qu.: 84.00   3rd Qu.: 90.00   3rd Qu.: 86.00
##  Max.   :100.00   Max.   :100.00   Max.   :100.00   Max.   :100.00
##      rh_ws7           rh_ws8           rh_ws9           rh_ws10
##  Min.   : 11.00   Min.   :  3.00   Min.   :  1.00   Min.   :  5.00
```

```
##  1st Qu.: 58.00   1st Qu.: 52.00   1st Qu.: 52.00   1st Qu.: 53.00
##  Median : 77.00   Median : 73.00   Median : 69.00   Median : 74.00
##  Mean   : 72.87   Mean   : 69.12   Mean   : 66.82   Mean   : 70.78
##  3rd Qu.: 90.00   3rd Qu.: 88.00   3rd Qu.: 83.00   3rd Qu.: 92.00
##  Max.   :100.00   Max.   :100.00   Max.   :100.00   Max.   :100.00
##     rh_ws11          rh_ws12          rh_ws13          rh_ws14
##  Min.   :  8.00   Min.   :  4.00   Min.   :  1.00   Min.   :  8.00
##  1st Qu.: 46.00   1st Qu.: 46.00   1st Qu.: 49.00   1st Qu.: 48.00
##  Median : 63.00   Median : 64.00   Median : 69.00   Median : 66.00
##  Mean   : 63.04   Mean   : 63.26   Mean   : 66.26   Mean   : 64.97
##  3rd Qu.: 81.00   3rd Qu.: 82.00   3rd Qu.: 84.00   3rd Qu.: 84.00
##  Max.   :100.00   Max.   :100.00   Max.   :100.00   Max.   :100.00
##     rh_ws15          rh_ws16          rh_ws17          rh_ws18
##  Min.   :  7.00   Min.   : 18.00   Min.   :  9.00   Min.   : 10.00
##  1st Qu.: 50.00   1st Qu.: 65.00   1st Qu.: 48.00   1st Qu.: 58.00
##  Median : 69.00   Median : 77.00   Median : 67.00   Median : 75.00
##  Mean   : 68.13   Mean   : 75.27   Mean   : 65.35   Mean   : 71.01
##  3rd Qu.: 88.00   3rd Qu.: 87.00   3rd Qu.: 84.00   3rd Qu.: 87.00
##  Max.   :100.00   Max.   :100.00   Max.   :100.00   Max.   :100.00
##     rh_ws19          rh_ws20          rh_ws21          rh_ws22
##  Min.   :  0.00   Min.   : 11.00   Min.   :  1.00   Min.   :  1.00
##  1st Qu.: 49.00   1st Qu.: 65.00   1st Qu.: 49.00   1st Qu.: 47.00
##  Median : 74.00   Median : 77.00   Median : 69.00   Median : 66.00
##  Mean   : 68.68   Mean   : 73.96   Mean   : 66.97   Mean   : 63.75
##  3rd Qu.: 89.00   3rd Qu.: 87.00   3rd Qu.: 88.00   3rd Qu.: 83.00
##  Max.   :100.00   Max.   :100.00   Max.   :100.00   Max.   :100.00
##     rh_ws23          rh_ws24          rh_ws25         rh_ws26
##  Min.   :  9.00   Min.   :  9.00   Min.   :  8.0   Min.   :  9.00
##  1st Qu.: 60.00   1st Qu.: 54.00   1st Qu.: 50.0   1st Qu.: 53.00
##  Median : 78.00   Median : 70.00   Median : 70.0   Median : 75.00
##  Mean   : 73.88   Mean   : 68.91   Mean   : 67.7   Mean   : 71.28
##  3rd Qu.: 91.00   3rd Qu.: 85.00   3rd Qu.: 87.0   3rd Qu.: 92.00
##  Max.   :100.00   Max.   :100.00   Max.   :100.0   Max.   :100.00
##     rh_ws27          rh_ws28            Day             Month
##  Min.   :  8.00   Min.   :  8.00   Min.   : 1.00   Min.   : 1.000
##  1st Qu.: 48.00   1st Qu.: 52.00   1st Qu.: 8.00   1st Qu.: 3.000
##  Median : 67.00   Median : 74.00   Median :16.00   Median : 6.000
##  Mean   : 64.95   Mean   : 69.86   Mean   :15.72   Mean   : 6.294
##  3rd Qu.: 84.00   3rd Qu.: 89.00   3rd Qu.:23.00   3rd Qu.: 9.000
##  Max.   :100.00   Max.   :100.00   Max.   :31.00   Max.   :12.000
##       Year
##  Min.   :2005
##  1st Qu.:2006
##  Median :2008
##  Mean   :2008
##  3rd Qu.:2009
##  Max.   :2011
```

```
#summarize daily data
temp_daily <- temperature_all %>%
  group_by(date, Year, Month, Day) %>% # here we left column with hour out to calculate daily mean
  #daily mean for each station
  summarise_at(vars(matches("^t_ws[1-9]|^t_ws1[0-8]$")), list(Daily_temp = ~mean(., na.rm = TRUE)))
```

```r
rh_daily <- relative_humidity_all %>%
  group_by(date, Year, Month, Day) %>% # here we left column with hour out to calculate daily mean
  #daily mean for each station
  summarise_at(vars(matches("^rh_ws[1-9]|^rh_ws1[0-8]$")), list(Daily_temp = ~mean(., na.rm = TRUE)))
```

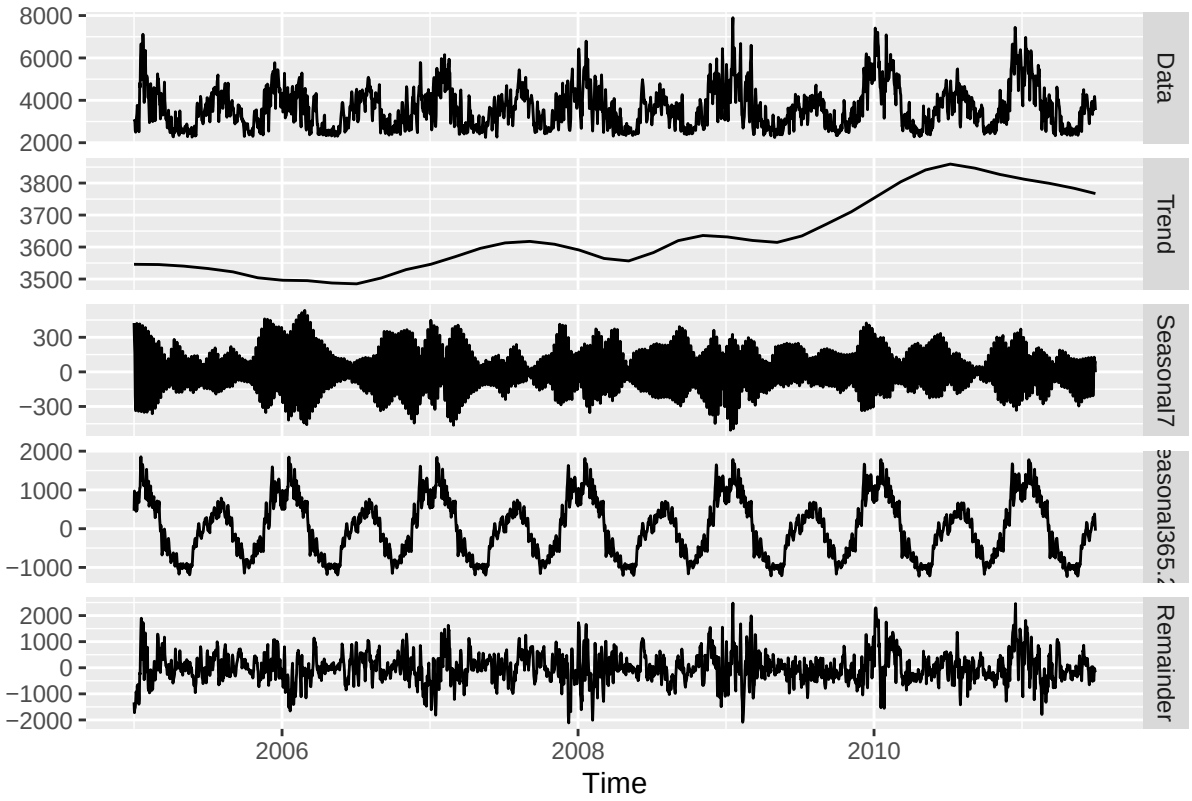
```r
#save datasets
write.csv(load_daily,file=here("Data","Processed","Daily_Load.csv"))
write.csv(temp_daily,file=here("Data","Processed","Daily_Temp.csv"))
write.csv(rh_daily,file=here("Data","Processed","Daily_Relative_Humidity.csv"))
```

```r
# load exogenous variables
raw_temp <- read.csv(here('Data','Processed','Daily_Temp.csv'))
raw_temp <- raw_temp %>%
  drop_na()

raw_humidity <- read.csv(here('Data','Processed','Daily_Relative_Humidity.csv'))

avg_temp <- raw_temp %>%
  mutate(avgTemp = rowMeans(select(., 6:33)))
```

CREATE A TIME SERIES OBJECT

```r
#load time series object
ts_load_daily <- msts(load_daily$Daily_mean_load, #daily data frame mean
                      seasonal.periods =c(7,365.25), #seasonal periods
                      start=c(2005,1,1))
```

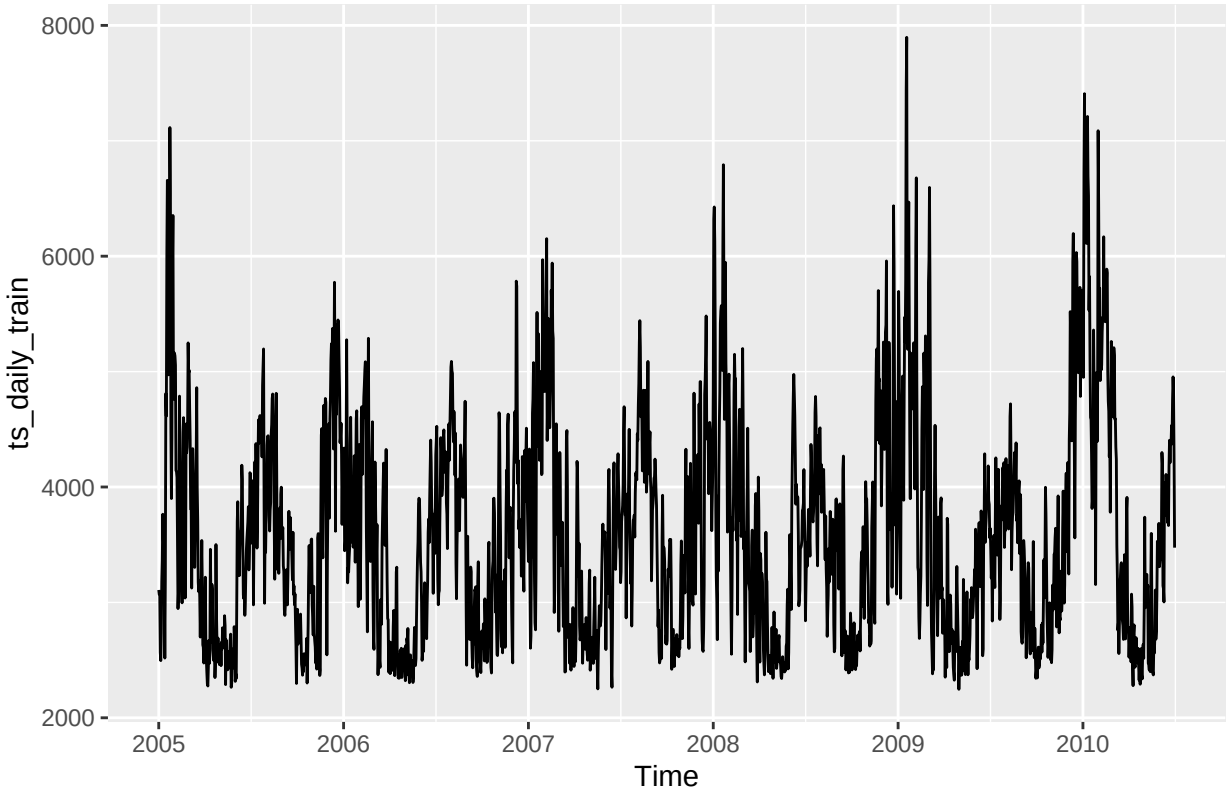
```r
#decomposition of daily data
ts_load_daily %>% mstl() %>%
  autoplot()
```



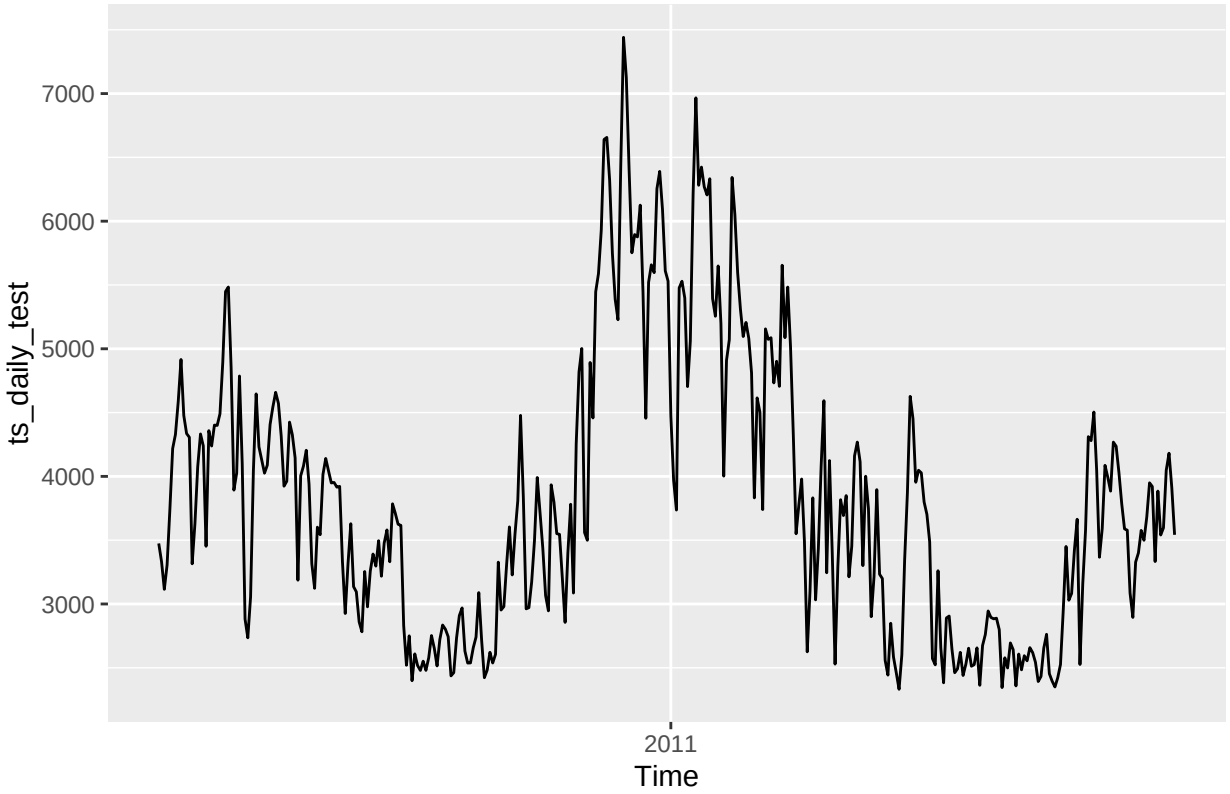
CREATING TEST AND TRAINING DATASETS (YEAR AND MONTH)

```
#training and test datasets

#create a subset for training purpose (try forecasting for one month!)
n_for = 365
ts_daily_train <- subset(ts_load_daily,
                                    end = length(ts_load_daily)-n_for)

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

#test is just a month
ts_daily_train_month <- subset(ts_load_daily,
                                    end = length(ts_load_daily)-30)

#create a subset for testing purpose
ts_daily_test_month <- subset(ts_load_daily,
                                    start = length(ts_load_daily)-30)
```

```
#plotting these test and train data
autoplot(ts_daily_train)
```




```
autoplot(ts_daily_test)
```



FITTING AND FORECASTING MODELS

Neural Networks

###K(2,4)

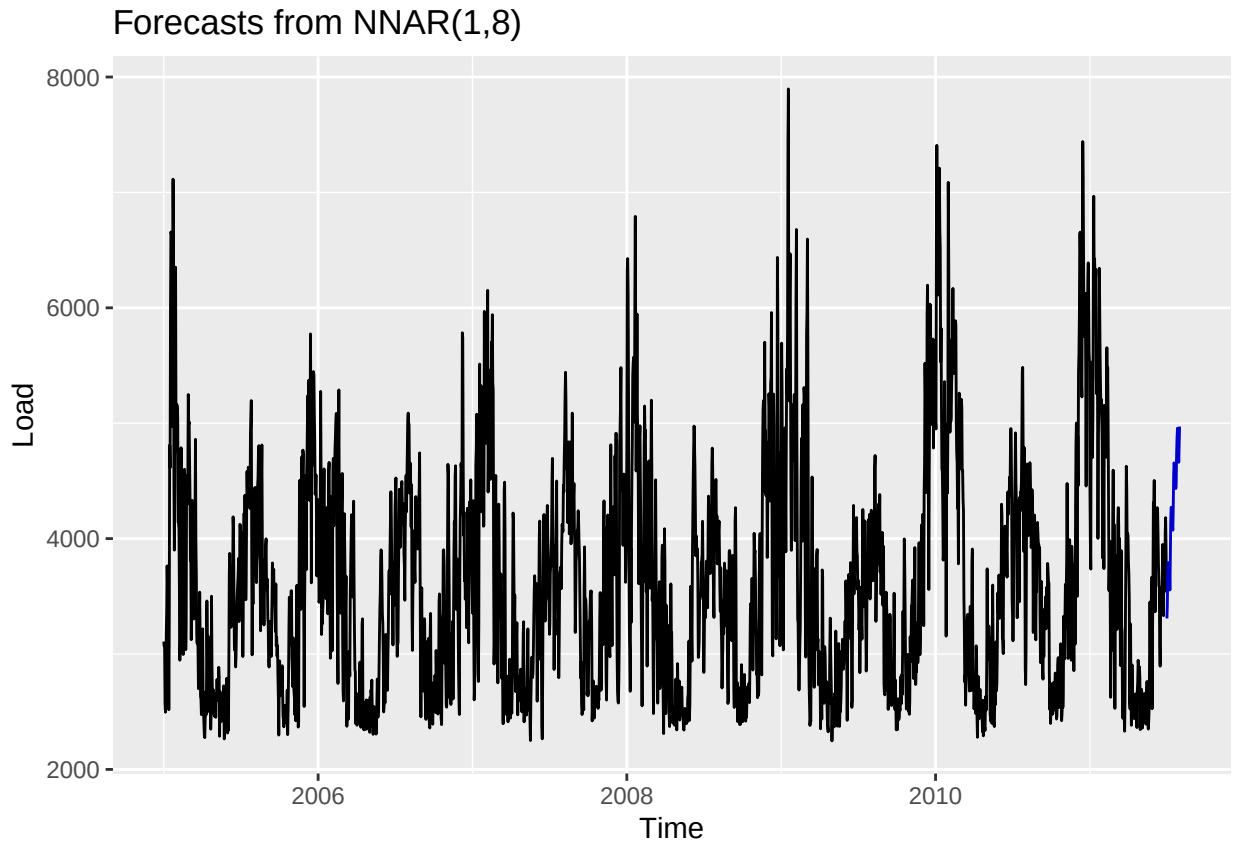
```
temp_daily <- drop_na(temp_daily)

ts_meter1 <- msts(temp_daily$t_ws1_Daily_temp,
                  start=c(2005,1,1),
                  seasonal.periods =c(7,365.25))

temp_meter1_for <- forecast(ts_meter1, h=31)
```

```
# Generate Fourier series components
fourier_components <- fourier(ts_load_daily, K=c(2,4))
fourier_for <- fourier(ts_load_daily, K=c(2,4), h=31)

# Combine Fourier components with temperature data
regressors <- cbind(as.matrix(data.frame(fourier_components)),
                    "temp" = ts_meter1)

regressors_for <- cbind(as.matrix(data.frame(fourier_for)),
                    "temp" = temp_meter1_for$mean)
```

```
#NN_fit <- nnetar(ts_act_power_daily_train,p=1,P=1)
NN_fit <- nnetar(ts_load_daily,p=1,P=0,
                 xreg=regressors)

#NN_for <- forecast(NN_fit, h=365)
NN_for <- forecast(NN_fit, h=31, xreg=regressors_for)
```

```
## Warning in forecast.nnetar(NN_fit, h = 31, xreg = regressors_for): xreg
## contains different column names from the xreg used in training. Please check
## that the regressors are in the same order.
```

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




```
# adding results to submission template
submission <- read_csv(here("output", "submission_template.csv"),
    col_types = cols(load = col_number()))

submission$load <- NN_for$mean

write.csv(submission, file=here("output","Submission2_EK_JW.csv"),
          row.names = F)
```

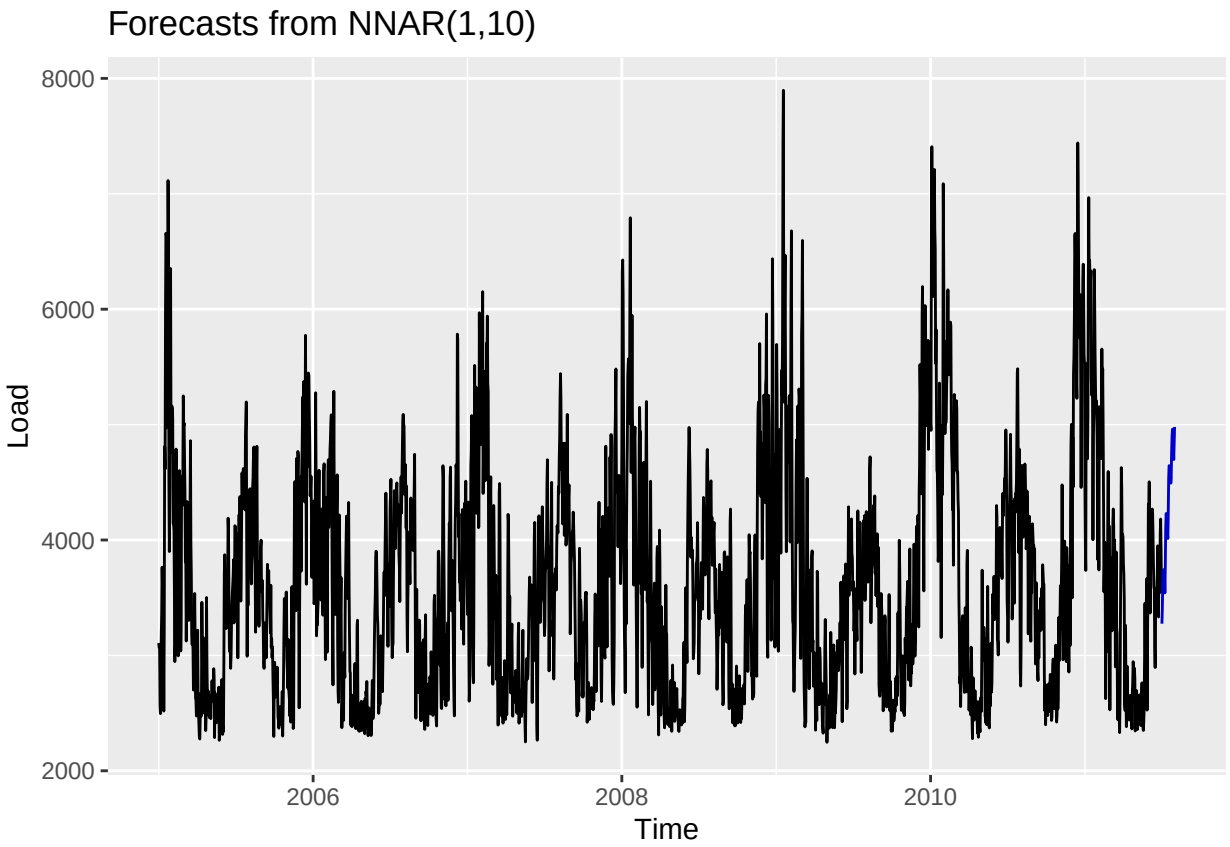
###K(2,6)

```
NN_testing <- function(x,X) {
  # Generate Fourier series components
  fourier_components <- fourier(ts_load_daily, K=c(x,X))
  fourier_for <- fourier(ts_load_daily, K=c(x,X), h=31)

  # Combine Fourier components with temperature data
  regressors <- cbind(as.matrix(data.frame(fourier_components)),
                      "temp" = ts_meter1)

  regressors_for <- cbind(as.matrix(data.frame(fourier_for)),
                      "temp" = temp_meter1_for$mean)

  #NN_fit <- nnetar(ts_act_power_daily_train,p=1,P=1)
  NN_fit <- nnetar(ts_load_daily,p=1,P=0,
                   xreg=regressors)

  #NN_for <- forecast(NN_fit, h=365)
  NN_for <- forecast(NN_fit, h=31, xreg=regressors_for)

  return(NN_for)

}
```

```
NN_26 <- NN_testing(2,6)
```

```
## Warning in forecast.nnetar(NN_fit, h = 31, xreg = regressors_for): xreg
## contains different column names from the xreg used in training. Please check
## that the regressors are in the same order.
```

```
autoplot(NN_26) +
  ylab("Load")
```



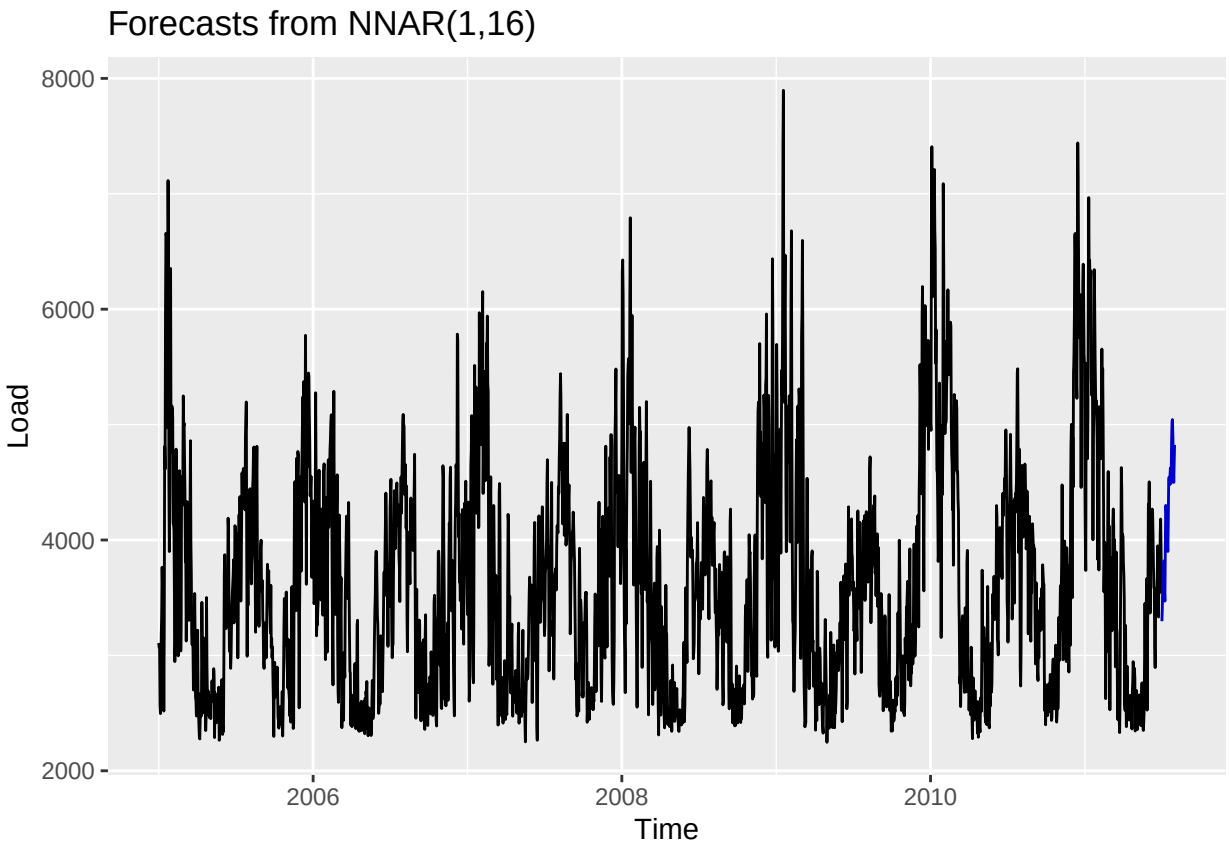

```
submission$load <- NN_26$mean

write.csv(submission, file=here("output","Submission3_EK_JW.csv"),
          row.names = F)
```

###K(2,12)

```
NN_212 <- NN_testing(2,12)
```

```
## Warning in forecast.nnetar(NN_fit, h = 31, xreg = regressors_for): xreg
## contains different column names from the xreg used in training. Please check
## that the regressors are in the same order.
```

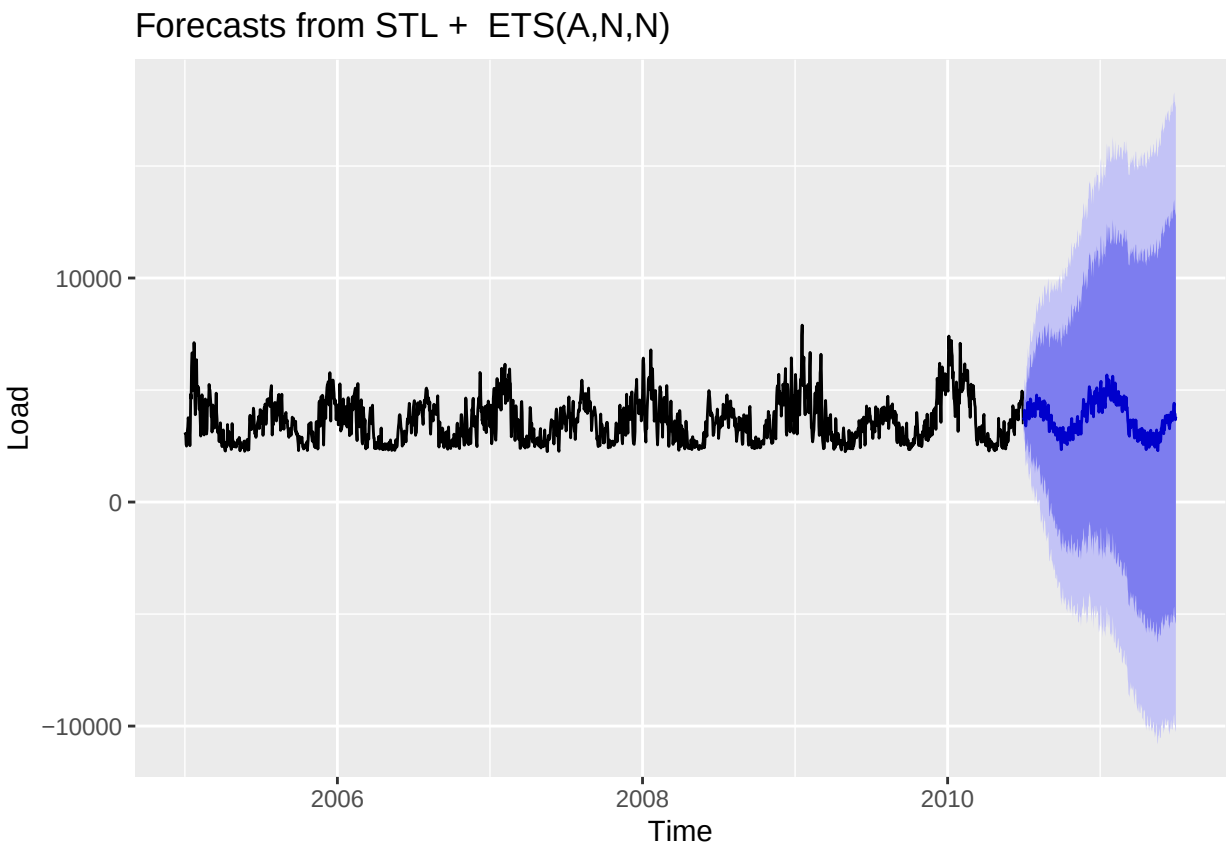
```
autoplot(NN_212) +
  ylab("Load")
```



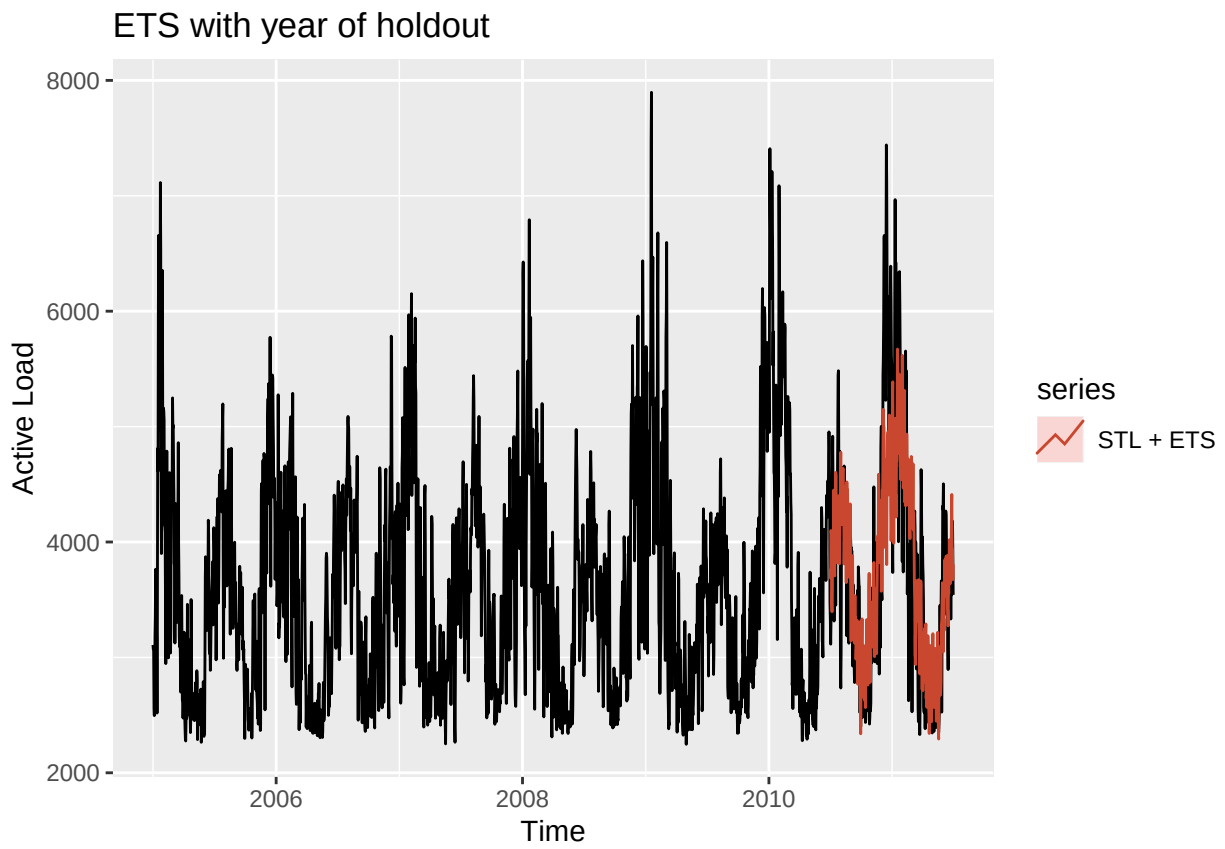

```
submission$load <- NN_212$mean

write.csv(submission, file=here("output","Submission4_EK_JW.csv"),
          row.names = F)
```

ETS

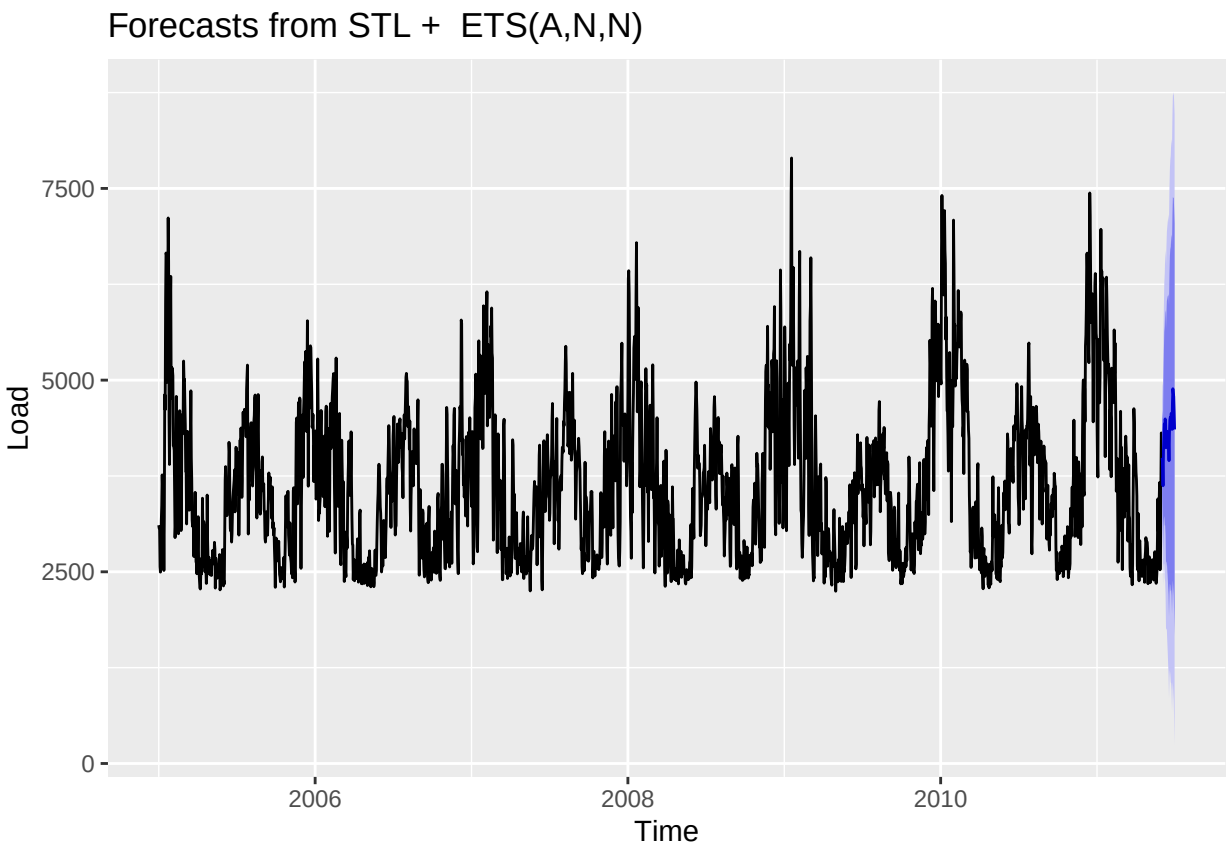
```
#Fit and forecast STL + ETS model to data with a year of holdout data
ETS_fit <-  stlf(ts_daily_train,h=365)

#Plot foresting results
autoplot(ETS_fit) + ylab("Load")
```



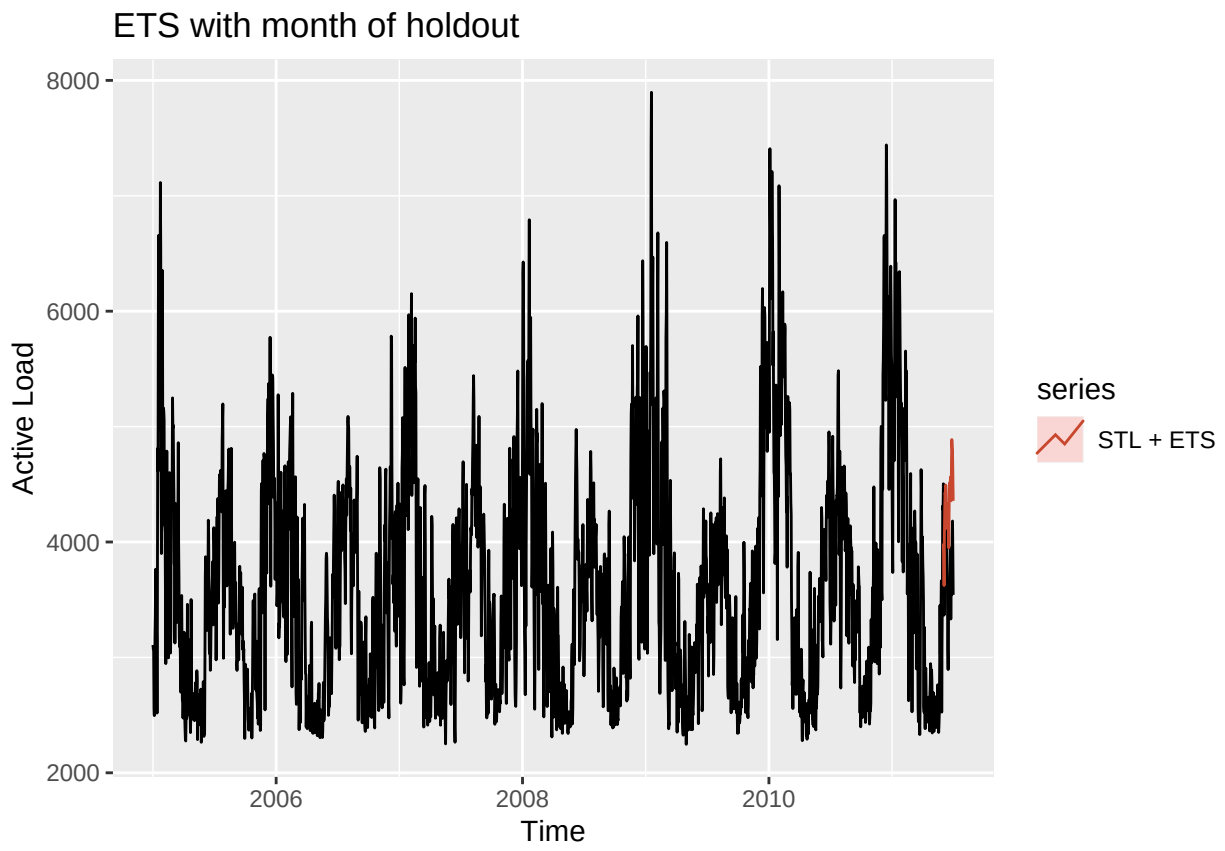

```
#Plot model + observed data
autoplot(ts_load_daily) +
  autolayer(ETS_fit, series="STL + ETS",PI=FALSE) +
  ylab("Active Load") +
  ggtitle("ETS with year of holdout")
```




```
#trying with just the month test set as opposed to year

ETS_fit_2 <-  stlf(ts_daily_train_month,h=30)

#Plot foresting results
autoplot(ETS_fit_2) + ylab("Load")
```




```
#Plot model + observed data
autoplot(ts_load_daily) +
  autolayer(ETS_fit_2, series="STL + ETS",PI=FALSE) +
  ylab("Active Load") +
  ggtitle("ETS with month of holdout")
```




```
ETS_scores <- accuracy(ETS_fit$mean,ts_daily_test)
ETS_scores
```

```
##                ME     RMSE      MAE       MPE     MAPE      ACF1 Theil's U
## Test set 71.213 769.6984 582.0748 -1.677709 14.97278 0.7371494  1.470813
```

```
ETS_scores_2 <- accuracy(ETS_fit_2$mean,ts_daily_test_month)
ETS_scores_2
```

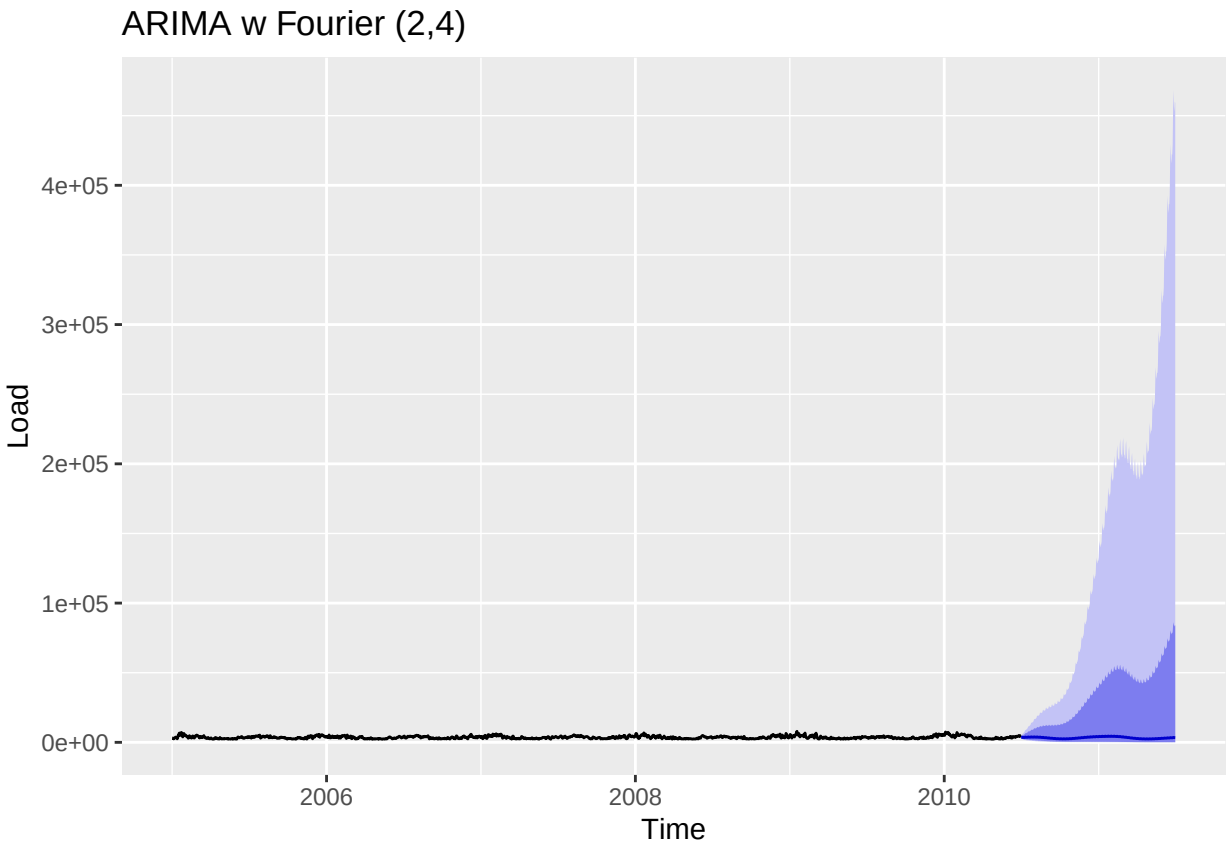
```
##                  ME     RMSE      MAE      MPE     MAPE     ACF1 Theil's U
## Test set -609.6557 736.2245 653.4374 -17.2509 18.24626 0.555879   2.35369
```

```
# adding results from ETS_fit to submission template (year holdout)
submission_ETS <- read_csv(here("output","submission_template.csv"),
    col_types = cols(load = col_number()))

submission_ETS$load <- ETS_fit$fitted[1977:2007]

#creating .csv output
write.csv(submission_ETS,file=here("output","Submission1_EK_JW.csv"),
          row.names = F)

# adding results from ETS_fit2 to submission template (month holdout)
submission_ETS2 <- read_csv(here("output","submission_template.csv"),
```

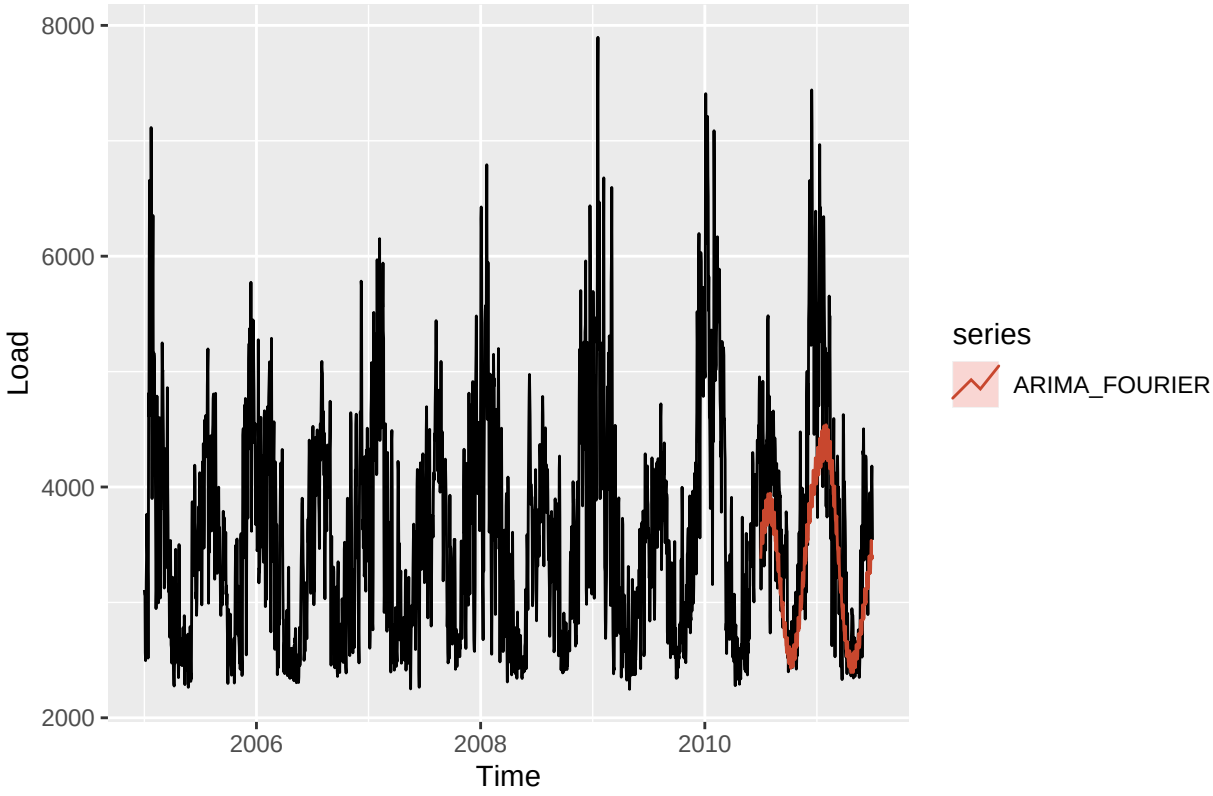
```
    col_types = cols(load = col_number()))

submission_ETS2$load <- ETS_fit_2$fitted[2312:2342]

#creating .csv output
write.csv(submission_ETS2,file=here("output","Submission5_EK_JW.csv"),
          row.names = F)
```

Arima and Fourier

```
ARIMA_Four_fit <- auto.arima(ts_daily_train,
                             seasonal=FALSE,
                             lambda=0,
                             xreg=fourier(ts_daily_train,
                                          K=c(2,4))
                             )

#Forecast with ARIMA fit
#also need to specify h for fourier terms
ARIMA_Four_for <- forecast(ARIMA_Four_fit,
                           xreg=fourier(ts_daily_train,
                                        K=c(2,4),
                                        h=365), h=365)
#Plot foresting results
autoplot(ARIMA_Four_for) +
  ylab("Load") +
  ggtitle("ARIMA w Fourier (2,4)")
```




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




```
#Model 2: ARIMA + Fourier
ARIMA_scores <- accuracy(ARIMA_Four_for$mean,ts_daily_test)
ARIMA_scores
```

```
##                 ME     RMSE      MAE      MPE     MAPE      ACF1 Theil's U
## Test set 506.1846 899.7746 651.9007 10.05863 15.02617 0.7929017  1.535785
```

```
# adding results from arima1 to submission template (month holdout)
submission_arima1 <- read_csv(here("output","submission_template.csv"),
    col_types = cols(load = col_number()))
submission_arima1$load <- ARIMA_Four_for$fitted[1977:2007]

#creating .csv output
write.csv(submission_arima1,file=here("output","Submission6_EK_JW.csv"),
          row.names = F)
```

ARIMA w (2,4) Fourier and Exogenous variables

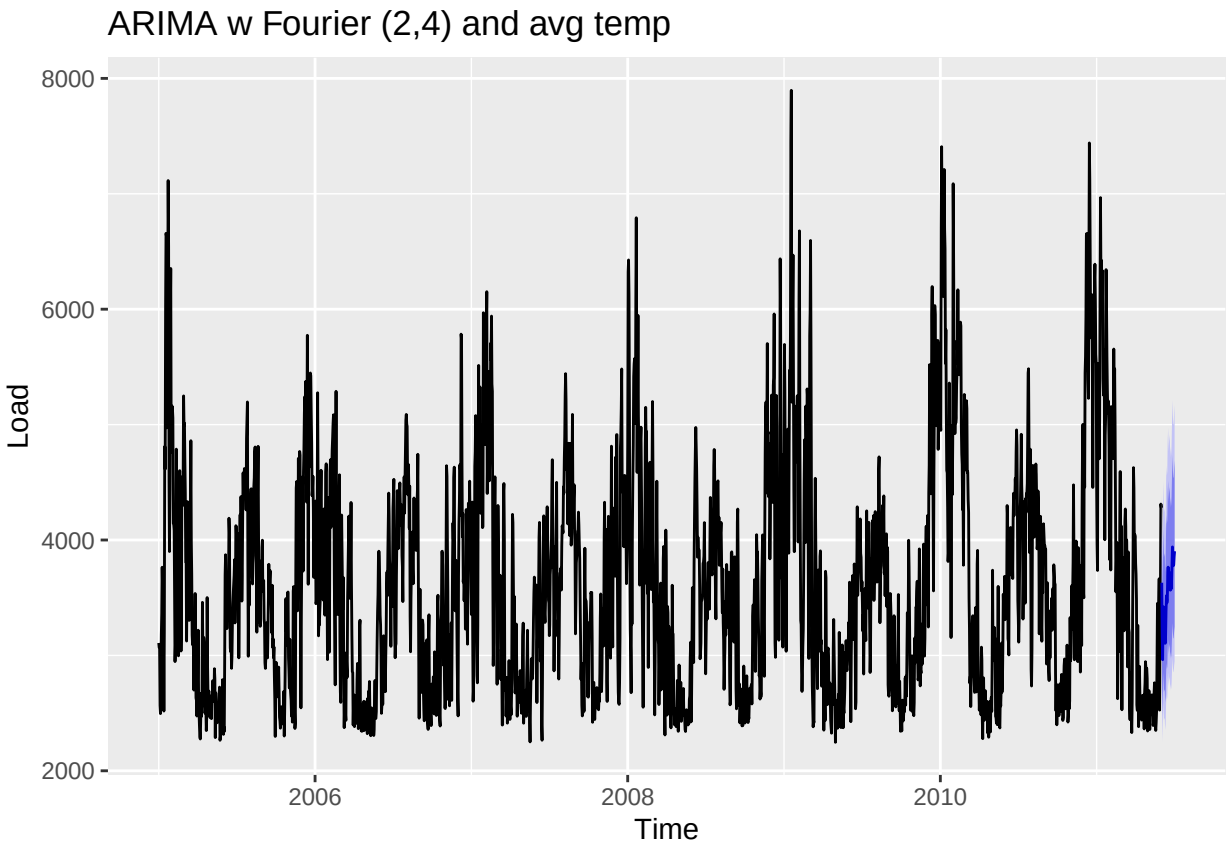
```
#ts object of exogenous variables
ts_avgtemp <- msts(avg_temp$avgTemp,
                   start=c(2005,1,1),
                   seasonal.periods =c(7,365.25))
```

```
#External regressors test and train
temp_avg_monthly_train <- subset(ts_avgtemp,
                                 end = length(ts_load_daily)-30)

temp_avg_monthly_test <- subset(ts_avgtemp,
                                 start = length(ts_load_daily)-30)

temp_avg_for <- forecast(ts_avgtemp, h=31)
```

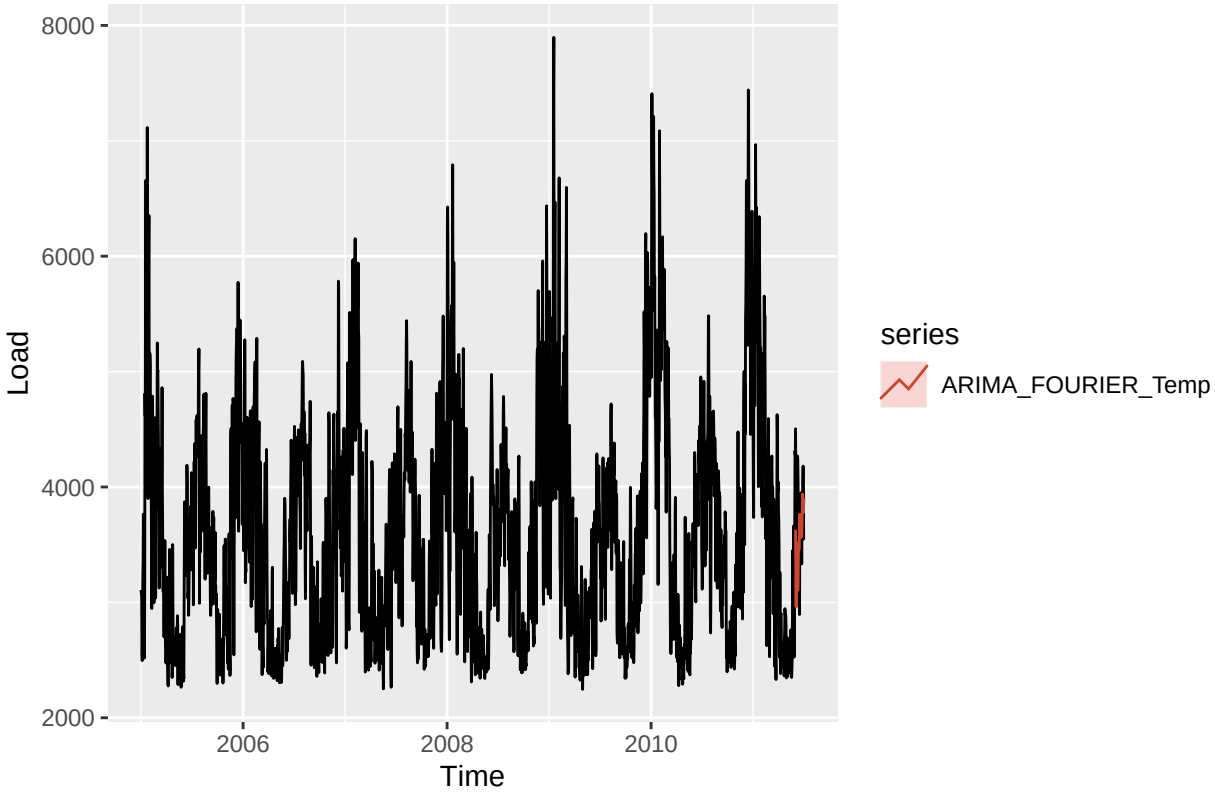
```
# Generate Fourier series components
fourier_components <- fourier(ts_daily_train_month, K=c(2,4))
fourier_for <- fourier(ts_daily_train_month, K=c(2,4), h=31)

# Combine Fourier components with temperature data
regressors <- cbind(as.matrix(data.frame(fourier_components)),
                    "temp" = temp_avg_monthly_train)

regressors_for <- cbind(as.matrix(data.frame(fourier_for)),
                    "temp" = temp_avg_monthly_test)

ARIMA_Four_fit_temp <- auto.arima(ts_daily_train_month,
                            seasonal=FALSE,
                            lambda=0,
                            xreg=regressors)

#Forecast with ARIMA fit
#also need to specify h for fourier terms
ARIMA_Four_for_temp <- forecast(ARIMA_Four_fit_temp,
                           xreg=regressors_for)
```

```
## Warning in forecast.forecast_ARIMA(ARIMA_Four_fit_temp, xreg = regressors_for):
## xreg contains different column names from the xreg used in training. Please
## check that the regressors are in the same order.
```

```
#Plot foresting results
autoplot(ARIMA_Four_for_temp) +
  ylab("Load") +
  ggtitle("ARIMA w Fourier (2,4) and avg temp")
```




```
#Plot model + observed data
autoplot(ts_load_daily) +
  autolayer(ARIMA_Four_for_temp, series="ARIMA_FOURIER_Temp",PI=FALSE) +
  ylab("Load")
```




```
#Model 2: ARIMA + Fourier
ARIMA_temp_scores <- accuracy(ARIMA_Four_for_temp$mean,ts_daily_test_month)
ARIMA_temp_scores
```

```
##                ME     RMSE      MAE      MPE     MAPE      ACF1 Theil's U
## Test set 217.2469 529.8995 437.8554 4.745606 11.50781 0.7280883  1.506544
```

```
# adding results from ARIMA_Four_for_temp to submission template (month holdout)
submission_arima2 <- read_csv(here("output","submission_template.csv"),
    col_types = cols(load = col_number()))
submission_arima2$load <- ARIMA_Four_for_temp$fitted[1977:2007]

#creating .csv output
write.csv(submission_arima2,file=here("output","Submission7_EK_JW.csv"),
          row.names = F)
```

Ensemble approach of averaging best performing NN model results with June Load data

```
#load submission 2
Best_performing_model <- read_csv(here("output","Submission2_EK_JW.csv"))
```

```
## Rows: 31 Columns: 2
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl  (1): load
## date (1): date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
Best_performing_model$june_data<- load_daily$Daily_mean_load[2342:2372]

EnsembleAvg <- Best_performing_model %>%
  mutate(Ensemble =rowMeans(select(.,2:3)))

submission_ensemble_1 <- read_csv(here("output","submission_template.csv"),
    col_types = cols(load = col_number()))

submission_ensemble_1$load <- EnsembleAvg$Ensemble

#creating .csv output
write.csv(submission_ensemble_1,file=here("output","Submission8_EK_JW.csv"),
          row.names = F)
```