Final Report for Kaggle

Chloe Young and Zuocheng Zhang

2025-04-25

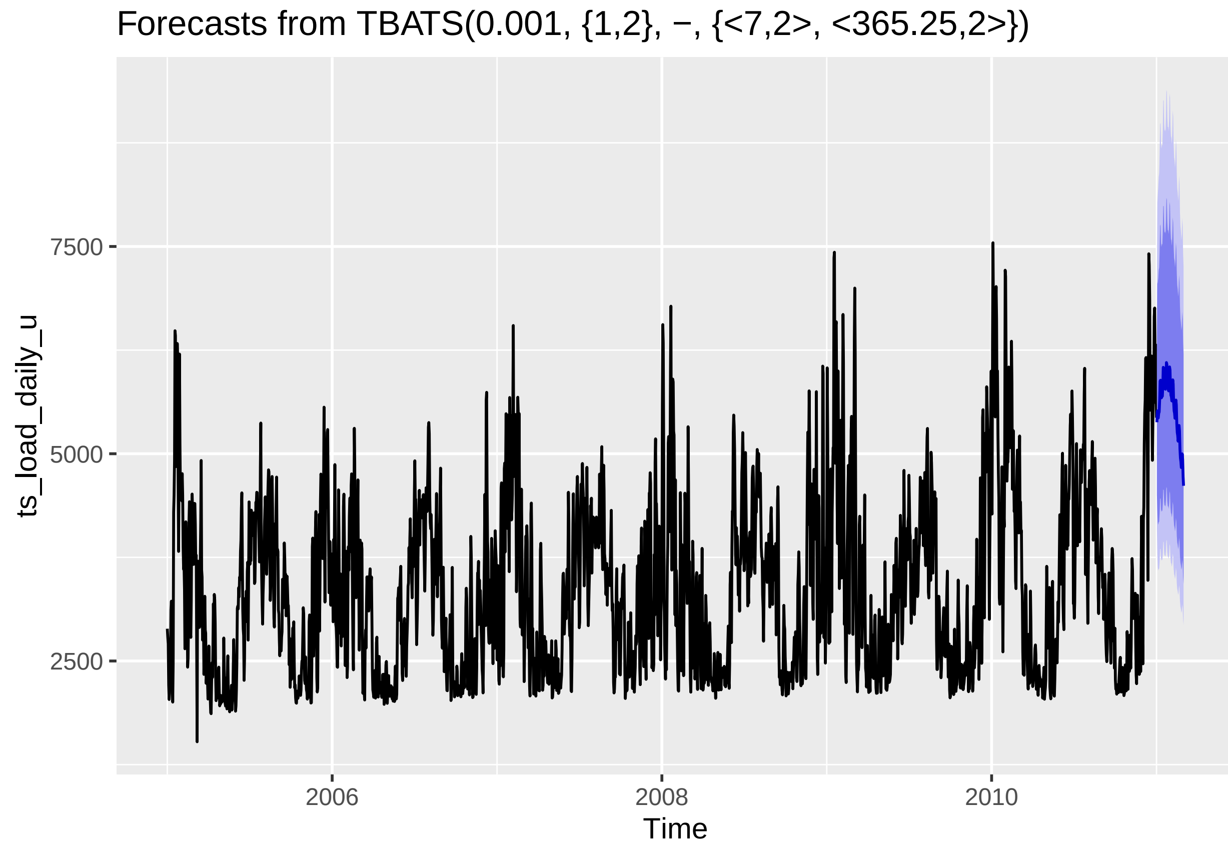
Introduction

For the Kaggle competition, we first tried several several models learnt from class, and then we stick to one model - Neural Network - and tried different features originate from data itself.

Model learnt from class

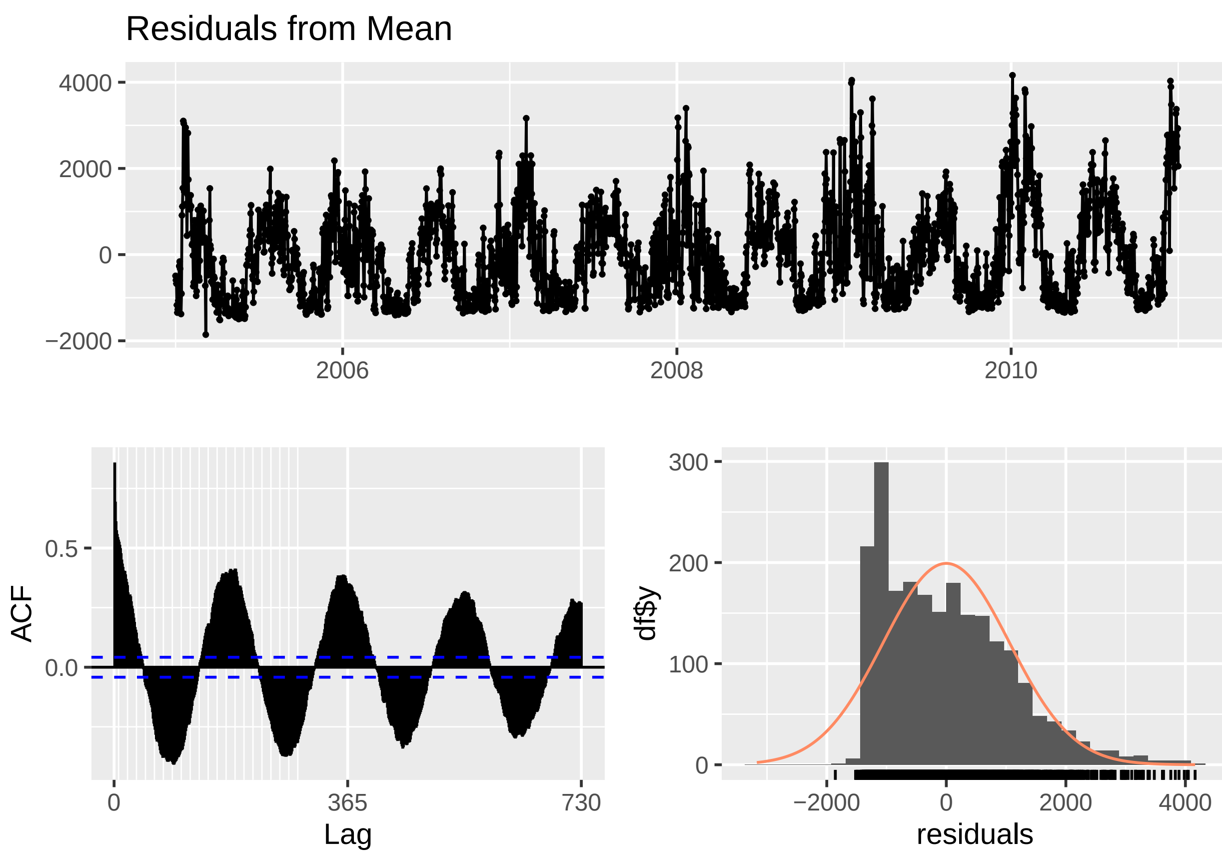
Model 1: TBATS

```
tbats_fit <- tbats(ts_load_daily_u)
tbats_forecast <- forecast(tbats_fit, h = 60)
autoplot(tbats_forecast)
```



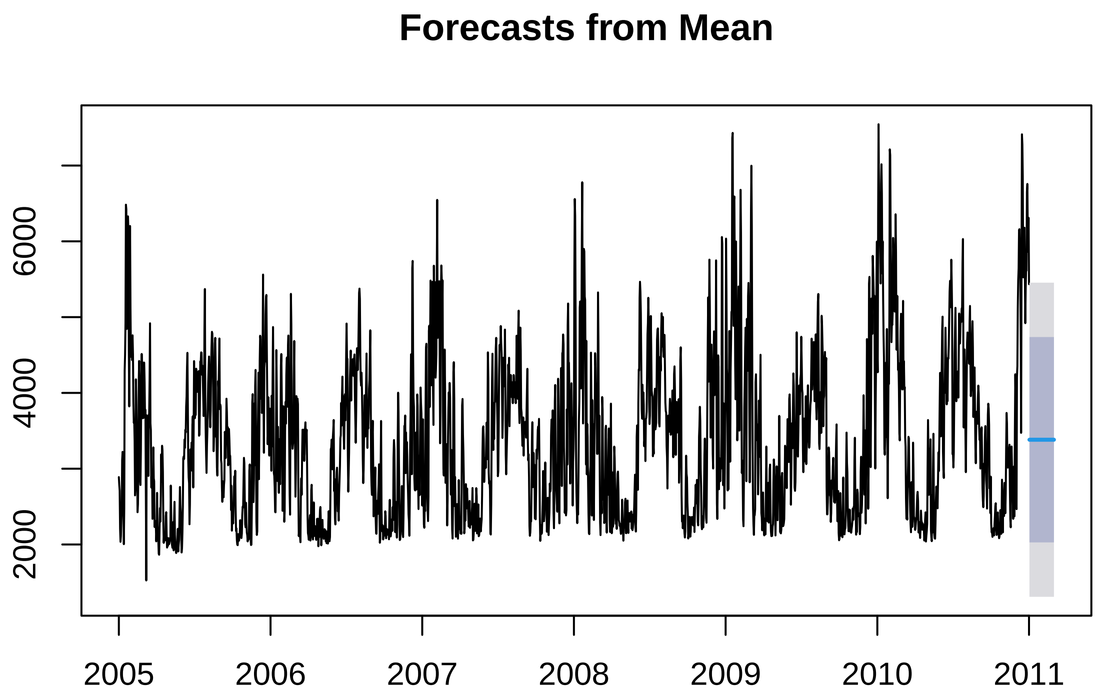
Model 2: Arithmetic mean

```
MEAN_seas <- meanf(y = ts_load_daily_u, h = 60)
checkresiduals(MEAN_seas)
```



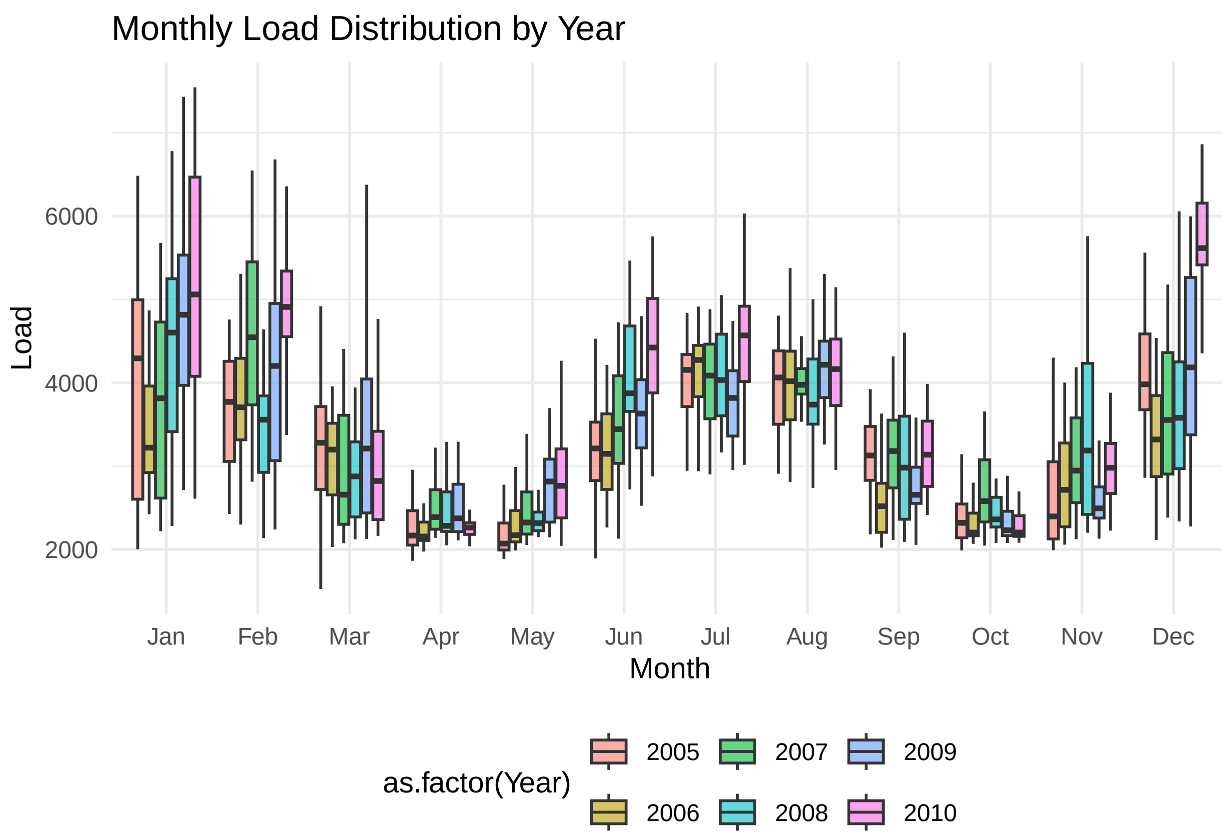

```
##
##  Ljung-Box test
##
## data:  Residuals from Mean
## Q* = 83192, df = 438, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 438
```

```
plot(MEAN_seas)
```

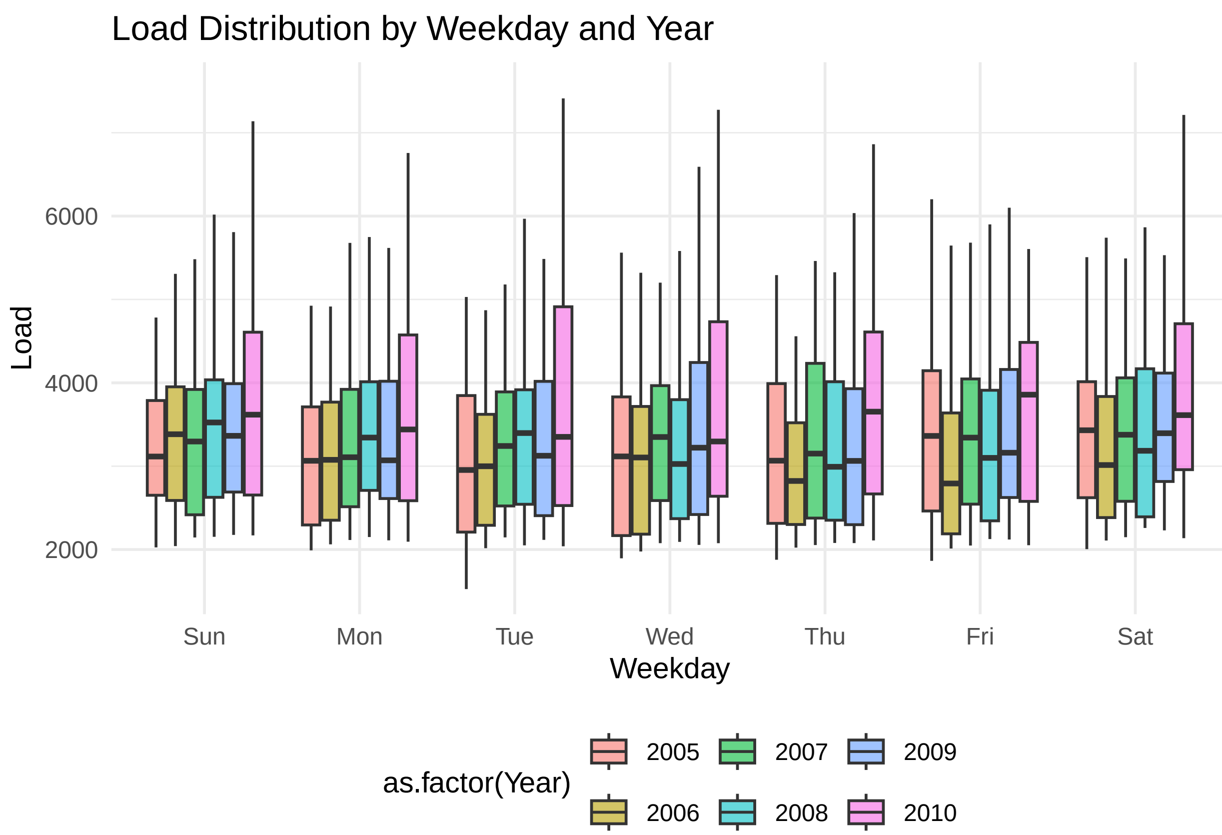


Neural Network w/data features

The reason we use data features is that some patterns of data could be an external factor of prediction. For example:




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



We see that the load varies a lot in different months, but not so different within days. We also considers other variables that could be interesting, such as season, weekends, holidays, as the load could be different in these days.

Also, by doing AFC and PACF plots, we find the autocorrelation, which could mean that we could add lagged features as external variables.

Finally, we also considered the fluctuation of data, as it could reflect more uncertainty patterns.

Weather variables are not helpful to predict daily data, we considered the lagged values of temperatures, but only the current temperature could do a little help. In the end, we didn't consider them for our best model.

Here are some procedures of extracting features

Get features

Day features

```
daily_data$date <- as.Date(daily_data$date)
us_years <- as.numeric(format(daily_data$date, "%Y"))
us_holidays <- holidayNYSE(unique(us_years))


daily_data <- daily_data %>%
  mutate(is_weekend = as.integer(weekdays(date) %in% c("Saturday", "Sunday"))) %>%
  mutate(quarter = quarter(date),is_us_holiday = as.integer(date %in% as.Date(us_holidays)))
```

```r
## Weekday

daily_data <- daily_data %>%
  mutate(weekday = weekdays(date)) %>%
  mutate(weekday = factor(weekday,
                          levels = c("Monday", "Tuesday", "Wednesday",
                                     "Thursday", "Friday", "Saturday", "Sunday")))

everyday <- model.matrix(~ weekday - 1, data = daily_data)
daily_data <- cbind(daily_data, everyday)


## Season

daily_data <- daily_data %>%
  mutate(month = month(date)) %>%
  mutate(season = case_when(
    month %in% c(12, 1, 2)  ~ "Winter",
    month %in% c(3, 4, 5)   ~ "Spring",
    month %in% c(6, 7, 8)   ~ "Summer",
    month %in% c(9, 10, 11) ~ "Fall"
  ))

each_season <- model.matrix(~ season - 1, data = daily_data)
daily_data <- cbind(daily_data, each_season)


## Month
daily_data <- daily_data %>%
  mutate(month = month(date, label = TRUE, abbr = FALSE))

each_month <- model.matrix(~ month - 1, data = daily_data)
daily_data <- cbind(daily_data, each_month)
```

Lagged values of load

```r
daily_data <- daily_data %>%
              mutate(lag1 = lag(daily_data$load_daily,1),
                     lag2 = lag(daily_data$load_daily,2),
                     lag3 = lag(daily_data$load_daily,3),
                     lag4 = lag(daily_data$load_daily,4),
                     lag5 = lag(daily_data$load_daily,5),
                     lag6 = lag(daily_data$load_daily,6),
                     lag7 = lag(daily_data$load_daily,7),
                     )
```

Lagged values of weather

```r
daily_data <- daily_data %>%
  mutate(
    temp_lag_0 = temperature_daily_ts,
    temp_lag_1 = lag(temperature_daily_ts, 1),
    temp_lag_2 = lag(temperature_daily_ts, 2),
    temp_lag_3 = lag(temperature_daily_ts, 3)
  )



daily_data <- daily_data %>%
  mutate(
    humi_lag_0 = humidity_daily_ts,
    humi_lag_1 = lag(humidity_daily_ts, 1),
    humi_lag_1 = lag(humidity_daily_ts, 2)
  )
```

Moving average

```r
daily_data <- daily_data %>%
  mutate(
    rolling_mean_2 = rollapply(load_daily, width = 7, FUN = mean, align = "right", fill = NA),
    rolling_mean_3 = rollapply(load_daily, width = 7, FUN = mean, align = "right", fill = NA),
    rolling_mean_7 = rollapply(load_daily, width = 7, FUN = mean, align = "right", fill = NA),
  )
```

Drop NA

```r
daily_data <- daily_data %>% drop_na()


## Delete temp vars
daily_data <- daily_data %>% select(-c("season","weekday","month","quarter","humidity_daily_ts","tempera

#daily_data <- daily_data %>% select(-c("humidity_daily_ts","temperature_daily_ts"))
```

Fluctuation

```r
daily_data <- daily_data %>%
mutate(volatility_2d = rollapply(daily_data$load_daily, width = 3, FUN = sd, align = "right", fill = NA
       volatility_3d = rollapply(daily_data$load_daily, width = 7, FUN = sd, align = "right", fill = NA
       volatility_7d = rollapply(daily_data$load_daily, width = 7, FUN = sd, align = "right", fill = NA
       volatility_14d = rollapply(daily_data$load_daily, width = 7, FUN = sd, align = "right", fill = N
```

Transform to ts

```r
msts_load_daily <- msts(daily_data,
                           seasonal.periods =c(7,365.25),
                           start=c(2005,01,01))

total_length <- length(msts_load_daily[,1])


test_length <- 365


train_set <- window(msts_load_daily, end = c(2005 + (total_length - test_length - 1)/365.25))
test_set <- window(msts_load_daily, start = c(2005 + (total_length - test_length)/365.25))
```

Model 3

This model actually got the best prediction. We use the test sample as the future series, it works very well.

```r
variable <- c("lag1","lag2","lag3","volatility_3d","volatility_7d","is_weekend")

xreg_train <- cbind(fourier(train_set[,"load_daily"],K=c(2,6)), train_set[,variable])
xreg_test  <- cbind(fourier(test_set[,"load_daily"] ,K=c(2,6) , h=length(test_set[,"load_daily"])), tes

NN_fit <- nnetar(train_set[,"load_daily"], xreg = xreg_train, size=7, repeats = 40)
```

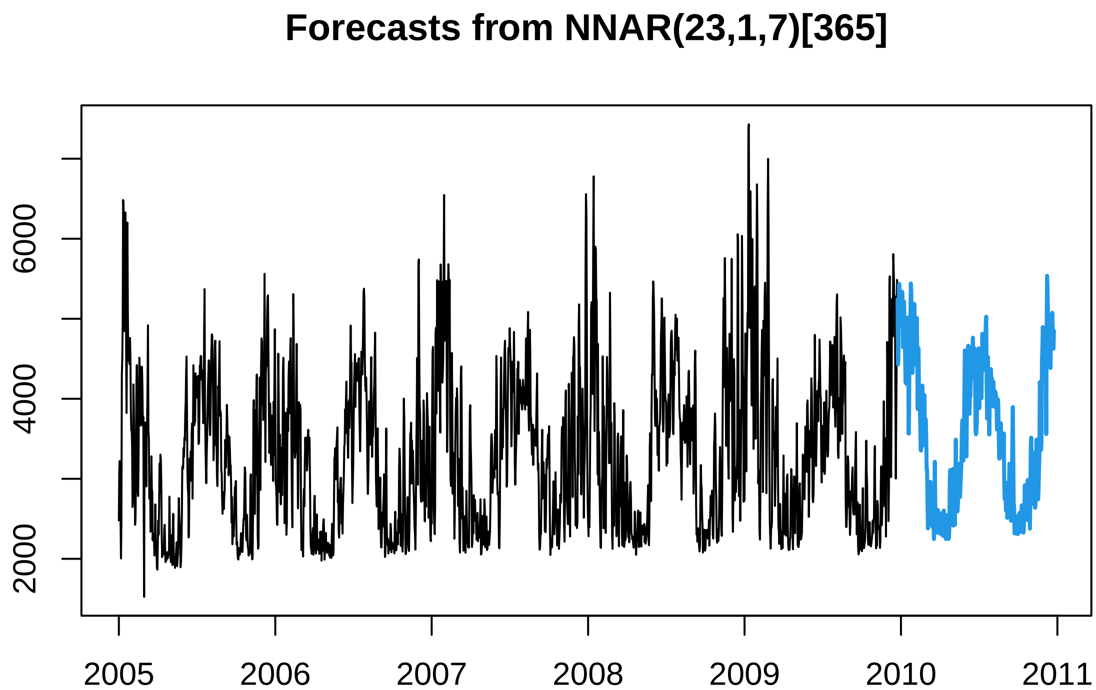
```
## Warning in nnetar(train_set[, "load_daily"], xreg = xreg_train, size = 7, :
## Missing values in xreg, omitting rows
```

```r
NN_for <- forecast(NN_fit, xreg = xreg_test, h=length(test_set[,1]))
```

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

```r
forecast_NN_accuracy <- accuracy(NN_for$mean, test_set[,"load_daily"])


plot(NN_for)
```



Model 4

Added month variebles

```
variable <- c("lag1","lag2","volatility_2d","monthApril","monthMay","monthOctober","monthJuly")
xreg_train <- cbind(fourier(train_set[,"load_daily"],K=c(2,6)), train_set[,variable])
xreg_test  <- cbind(fourier(test_set[,"load_daily"] ,K=c(2,6) , h=length(test_set[,"load_daily"])), tes

NN_fit <- nnetar(train_set[,"load_daily"], xreg = xreg_train, size=7, repeats = 40)
```

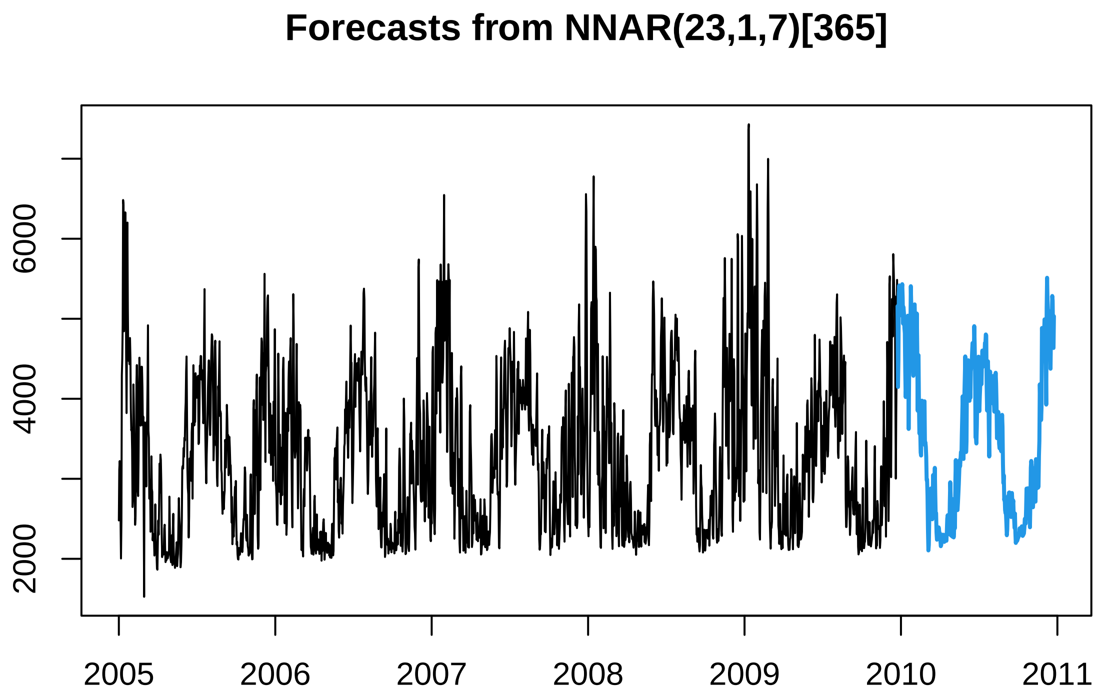
```
## Warning in nnetar(train_set[, "load_daily"], xreg = xreg_train, size = 7, :
## Missing values in xreg, omitting rows
```

```
NN_for <- forecast(NN_fit, xreg = xreg_test, h=length(test_set[,1]))
```

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

```
forecast_NN_accuracy <- accuracy(NN_for$mean, test_set[,"load_daily"])


plot(NN_for)
```



Model 5

Add everything (but not so good)

```
variable <- c(-2)
xreg_train <- cbind(fourier(train_set[,"load_daily"],K=c(2,6)), train_set[,variable])
xreg_test  <- cbind(fourier(test_set[,"load_daily"] ,K=c(2,6) , h=length(test_set[,"load_daily"])), tes

NN_fit <- nnetar(train_set[,"load_daily"], xreg = xreg_train, size=7, repeats = 40)
```

```
## Warning in nnetar(train_set[, "load_daily"], xreg = xreg_train, size = 7, :
## Missing values in xreg, omitting rows
```

```
NN_for <- forecast(NN_fit, xreg = xreg_test, h=length(test_set[,1]))
```

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

```
forecast_NN_accuracy <- accuracy(NN_for$mean, test_set[,"load_daily"])


plot(NN_for)
```

Forecasts from NNAR(23,1,7)[365]