ETM 510: Demand Forcasting

Efficacy of Support Vector Machines

Portland State University

Maseeh College of Engineering & Computer Science

Department of Electrical & Computer Engineering

Author:

Tylor Slay

March 21, 2017

Power Laboratory

Contents

1 Introduction

The purpose of this class was to develop an hourly demand forecast for 2017 based on on the previous demand and temperature data of the ISO NE region. There was no specified method to create the model, and the majority of the work was to be done outside of class. The class was supposed to coincide with the Global Energy Forecasting Competition (GefCOM) to drive innovations and competition. However, the class stated weeks into the competition leaving the students behind in both time and understanding.

The forecasting method chosen for this report was a Support Vector Machine (SVM) based on the mean hourly demand that factored in various time-series information such as: day-of-month, weekday, week-of-month, and month. The mean demand was chosen to reduce the error introduced by attempting to predict the weather as well as the demand. While the final results for the SVM model were not extraordinary by any means. It was still shown to be an affective tool for time-series forecasting.

This report does deviate from the majority of the class due to an error in the registration for the competition. Since this reports forecasts would not be seen by anyone other than in class. Nearly all of the competition deliverables were chosen to be either the previous years demand or it's mean hourly demand. It was evident that many of the models developed through the term were not good, therefore it seemed reasonable to simple use the previous years data as a place holder until a better model could be developed.

2 Literature Review

There were three sources of information that were found to be very helpful in our attempt to utilize the relatively new machine learning method know SVM's. "A Study on EUNITE Competition 2001" [1] was the inspiration for using SVM's in our demand forecasting. The team won EUNITE Competition using SVM's and

recommended temperature compensation and the addition of time-series concepts to improve demand forecasts. Their specific task was to predict max load for each day, which deviates from the goal of this study. Because of the difference in forcasts, temperature was considered.

"An Introduction to Statistical Learning with Applications in R" [2] was vital in understanding how SVM's worked and examples for R implementation. SVM's at their base definition are not complicated. They simply classify data using know characteristics. Usually the data has a specific classification for example, classifying species of plants. Each plant has specific colors, petals/stem dimensions, growth cycles/regions. The more data that can be define, the high the probability an SVM can make a correct prediction based on a new set of data. All machine learning tools do the same thing, but an SVM seemed to be particularly applicable to the forcasting we were attempting.

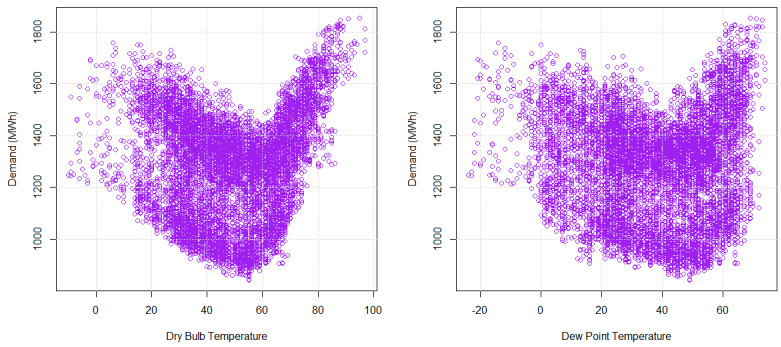
Finally the documentation for the package e1071 within R [3] was used to ensure proper implementation of the "libsvm" package. This package is just a time saving group of functions that are commonly used in support vector machine learning. We would not have had enough time to hard code each function into the demand forecast, while still learning about the very basics of demand forecasting.

3 Methodology

The initial data dive into the given ISO NE demand information revealed a few important aspects of the region specifically. The average demand for each year over the last 10 years has remained incredibly steady, with a slight decreasing trend. This is a very strange phenomenon as generally region demand increases with the number of customers, which causes growth in the commercial demand.

The average temperature over the last 10 years was also found to have a completely flat trend. Clearly this means that "Climate Change" is a hoax..., but on a more serious note, this could prove beneficial for the forecast as the weather is one of the hardest things to predicts.

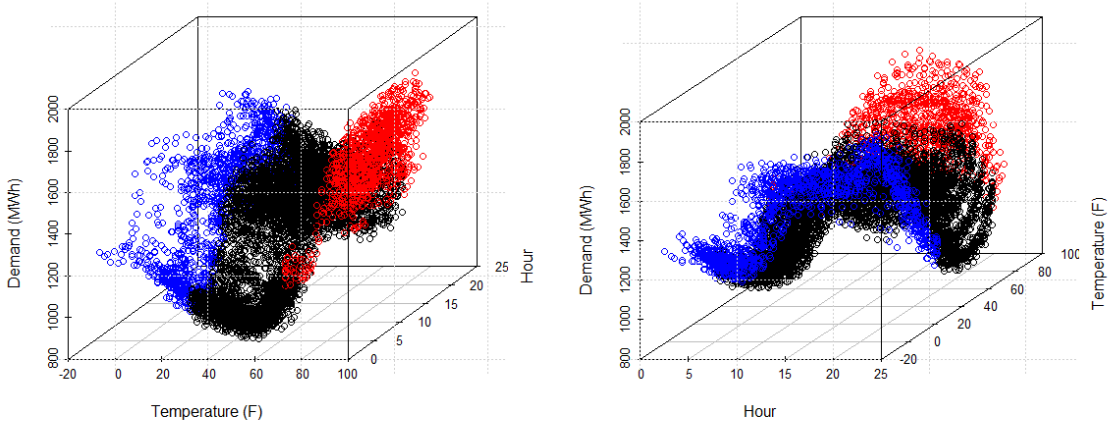
The data provides two temperature indicators, dry bulb and dew points. Dry bulb temperature is the free air temperature, and the dew point temperature factors in humidity. A comparison between the two seen in Figure 1 indicates a much stronger correlation between dry bulb temperature and demand. For this reason, the dew point temperature was excluded from the data set.



(a) (b)

Figure 1: Temperature correlation for 2016: (a) Demand vs Dry Bulb Temperature; (b) Demand vs Dew Point Temperature

Figure 2 demonstrates the relationship between the hour of the day and the temperature. The high temperatures have been colored red and lows blue to better visualize the change in daily profile as the temperature increases. The two plots clearly show a typical "duck" curve at low temperatures that transforms into a single peak for high temperatures.

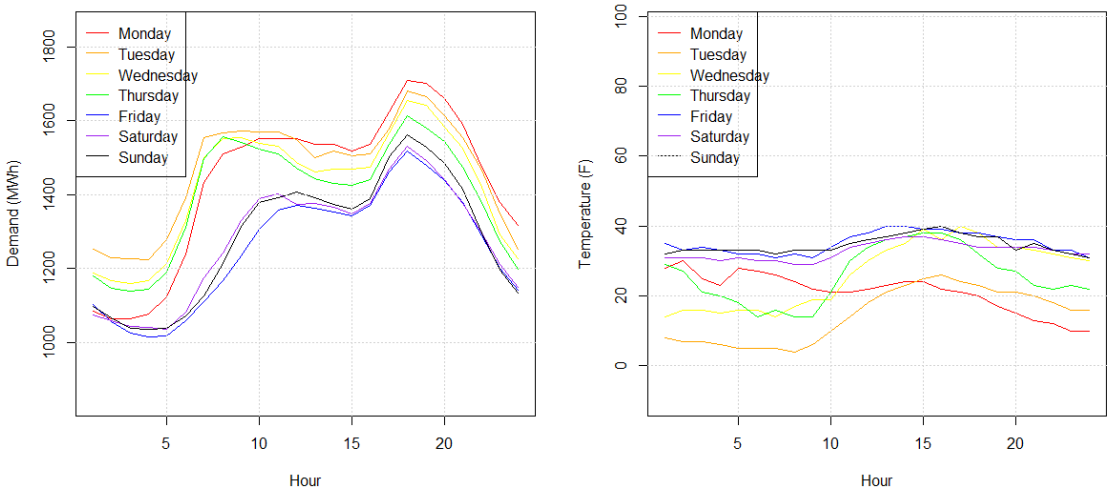


(a) (b)

Figure 2: 3D profile for 2016: (a) Demand vs Temperature vs Hour; (b) Demand vs Hour vs Temperature

It is evident that temperature is a strong indicator for demand forecasting, but the EUNITE article [1] concluded temperature may not be useful in mid-term load forecasting. This conclusion was due to the inability to accurately predict the weather for each day. Our study has an even more difficult task of predicting the hourly temperature for each day. With this realization, the mean demand for each hour was taken over the last 10 years. While this will exclude the specific affects of temperature, it should minimize the error the would be introduced by trying to predict the hourly temperature.

The other recommendation by the EUNITE team was to introduce time-series concepts. Figure 3 shows the the daily demand profile for each day of the week along with its corresponding daily temperature. While the peak demand for Friday, Saturday, and Sunday appears to be drastically reduced, it could be caused by the very mild temperature compared to the rest of the week.



(a) (b)

Figure 3: Daily profile for the first week of 2016: (a) Daily Demand; (b) Daily Temperature

One feature that appears to be unique is a delay in the morning demand on the weekend. Monday-Thursday seem to have a sharp increase at about 5 AM which peaks at around 8 AM. The demand for Friday-Sunday starts to increase at the same time, but the slope is decreased resulting in a morning peak later in the day at 10 AM. To capture the daily variance, we can aggregate the hourly demand for each weekday, each week of the month, and each month of the year.

Implementing the SVM for the data provided was not as straight forward as it would appear. The input matrix used the following variables: hour, day-of-month, weekday, week-of-month, month, and year. In the beginning, a great deal of time was spent trying to categorize the demand into specific classes as the output. The first method was trying to find demand days that were similar. The result was too hard to compare with the original data set due to the number of similar days.

The next trial was to classify days that were significantly different than the mean demand for the last 10 years. The goal was to highlight outliers in the data for

quick classification. However, it was found that the SVM models that were begin developed did not have a sufficiently large enough training set to create a model for a different month.

The documentation states that you should limit the training data, so we only used the data for the month in question from the previous year to predict the same month in the next year. This resulted in very poor forecasts that did not demonstrate the seasonality of each day/week of the month. When the training set was increased to the entire previous years data without trying to classify the demand, the forecast began to take shape. At this point, it did not seem necessary to proceed with trying to classify the hourly demand.

4 Results

Figure 4 shows the final demand forecast for the month of January 2017. This month was used as it was the most recent data available, giving us something to compare the SVM model to. The model follows the mean demand very closely for each hour. This was expected as the input data did not factor in the temperature or holidays specific to that day. The SVM model does appear to be closer to not only the daily profile for 2017, but also the demand magnitude.

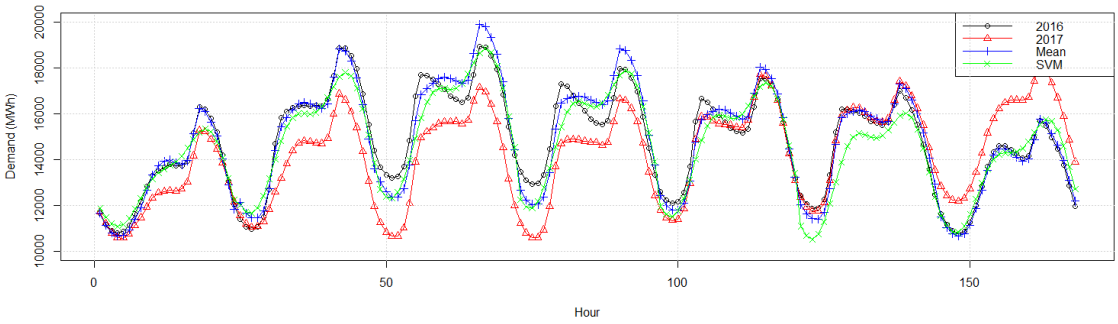


Figure 4: SVM forecast using mean demand data

To see just how well an SVM could perform using temperature as an input. We can create a new training set that uses the dry bulb temperature and real hourly demand. Figure 5 uses the actual temperature for January 2017 and uses it with the training set to create the model shown. As we see, the model is much closer to capturing the real hourly demand. It is extremely unlikely that we could model the actual hourly temperature perfectly, so other methods for improving the fit should be investigated.

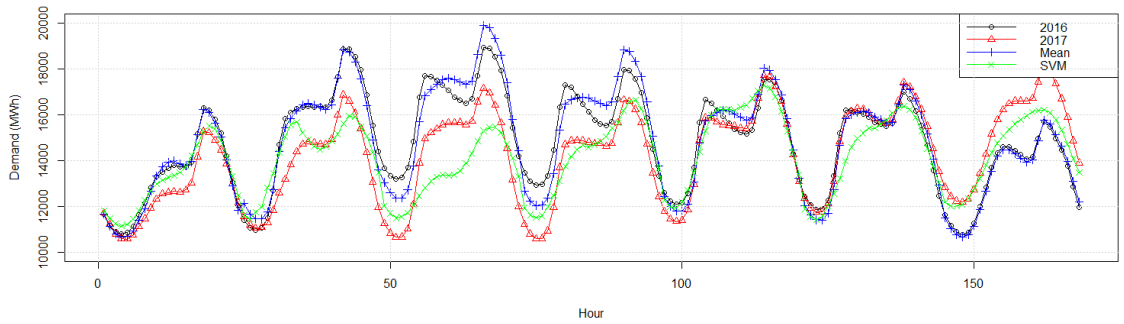


Figure 5: SVM forecast using real demand data with temperature

5 Conclusion

The goal for this study was to develop a model to forecast the hourly demand of the ISO NE region. We looked at at the time-series concepts to develop a mean demand model using the last 10 years of data that does just that. This model does not capture every day exactly, but it is a reasonable approximation for the unknown hourly demand of the future year as the results show.

The support vector machine model can still be improved. The is a tuning function, to hone the models predictive accuracy. There are various kernel methods that alter the data set to increase the classifying powers of the SVM's. These are all built into the libsvm package within R. Unfortunately there was just not enough of an understanding to implement into the study.

The data used in the training set can be expanded to include holidays and major events i.e. super bowl. Additionally with renewable energy generation on the rise, wind speed and solar penetration could be crucial for accurate demand forecasting. ISO NE models its day ahead demand at astonishing accuracy, but they utilize every technique and data set they have access to. They also do not try to create accurate models for more than a couple days ahead, as it is not realistic to presume any weather prediction will be accurate.

This class was an excellent learning experience, but it should be more of a traditional class structure if students with no background in forecasting continue to take the class. There is just too much information to cover in forecasting for a single person on their own to fully comprehend. The class also had a unique scenario were nearly every student chose a different forecasting method. This was great as it gave everyone insight into different ways to forecast, but it also made collaboration very difficult.

References

[1] M.-W. C. Bo-Juen Chen and C.-J. Lin, "Load forecasting using support vector machines: A study on eunite competition 2001," *IEEE*, vol. 19, no. 4, pp. 1821–1830, 2001.

[2] T. H. Gareth James, Daniela Witten and R. Tibshirani, *An Introduction to Statistical Learning with Applications in R.* Heidelberg Dordrecht London: Springer New York, 2013.

[3] D. Meyer, "Support vector machines: The interface to libsvm in package e1071."

A Excel Extract

```
cat("\014") #Clear consule

# Load libraries/packages
library(openxlsx)  # Read and write Excel files

setwd("C:\\Users\\Tylor\\Google Drive\\R\\ETM510_Forcast\\Data")

#assuming same naming convention for each file
file.years <- seq(2005,2015,1)
file.name <- sprintf("%d_smd_hourly.xlsx",file.years)
sheet.num <- "ISONE CA"
head.new <- c("Date", "Hr_End", "RT_Demand", "Dry_Bulb")

#assuming tab separated values with a header
datalist <- lapply(file.name, FUN=read.xlsx, sheet = sheet.num, detectDates = TRUE)

#assuming the same header/columns for all files
df1 <- do.call("rbind", datalist)

df1 <- data.frame(df1$Date, df1$Hour, df1$DEMAND, df1$DryBulb)

names(df1) <- head.new

#have to seperatly read 2016 and beyond as there was a formatting switch
file.years <- seq(2016,2017,1)
file.name <- sprintf("%d_smd_hourly.xlsx",file.years)
sheet.num <- "ISO NE CA"

datalist2 <- lapply(file.name, FUN=read.xlsx, sheet = sheet.num, detectDates = TRUE)
```

```
#assuming the same header/columns for all files
df2 <- do.call("rbind", datalist2)

df2 <- data.frame(df2$Date, df2$Hr_End, df2$RT_Demand, df2$Dry_Bulb)

names(df2) <- head.new

#for some reason the read.xlsx function won't read in the date so I must format
df2$Date <- as.Date(df2$Date, origin = "1899-12-30")
df2$Hr_End <- as.numeric(df2$Hr_End)

#combine all data into a single data frame
datafr <- rbind.data.frame(df1,df2)

#filter out any zero demand days
for(ii in 1:length(datafr$RT_Demand)){
  if (datafr$RT_Demand[ii] == 0){
    datafr$RT_Demand[ii] <- datafr$RT_Demand[ii-1]
  }
}

#build day of the year column
datafr$DOM <- as.numeric(format(as.Date(datafr$Date, format="%Y-%m-%d"), "%d"))

#build weekday column
datafr$weekday <- weekdays(as.Date(datafr$Date,"%Y-%m-%d"))

#build week of the year column
datafr$WOM <- ceiling(datafr$DOM/7)

#build month column
```

```
datafr$month <- months.Date(as.Date(datafr$Date,"%Y-%m-%d"))

#build year column
datafr$year <- format(as.Date(datafr$Date,"%Y-%m-%d"),'%Y')

#write to csv for faster reads during forcast
write.csv(datafr,"Demand_Summary.csv", row.names = FALSE)
```

B Data Dive

```
cat("\014") #Clear consule

# Load libraries/packages
library(scatterplot3d)
library(e1071)
library(rpart)
library(openxlsx)  # Read and write Excel files

setwd("C:\\Users\\Tylor\\Google Drive\\R\\ETM510_Forcast\\Data")

#assuming same naming convention for each file
file.years <- seq(2005,2015,1)
file.name <- sprintf("%d_smd_hourly.xlsx",file.years)

#assuming tab separated values with a header
datalist = lapply(file.name, FUN=read.xlsx, sheet = 3, detectDates = TRUE)

#assuming the same header/columns for all files
df1 = do.call("rbind", datalist)

head.new <- c("Date", "Hr_End", "DA_Demand", "RT_Demand", "DA_LMP",        "DA_EC",
```

```
              "DA_CC", "DA_MLC", "RT_LMP", "RT_EC", "RT_CC", "RT_MLC",
              "Dry_Bulb",        "Dew_Point")

names(df1) <- head.new

#have to seperatly read 2016 and beyond as there was a formatting switch
file.years <- seq(2016,2017,1)
file.name <- sprintf("%d_smd_hourly.xlsx",file.years)

datalist2 = lapply(file.name, FUN=read.xlsx, sheet = 3, detectDates = TRUE)

#assuming the same header/columns for all files
df2 = do.call("rbind", datalist2)

#for some reason the read.xlsx function won't read in the date so I must format
df2$Date <- as.Date(df2$Date, origin = "1899-12-30")

#combine all data into a single data frame
datafr <- rbind.data.frame(df1,df2)

#filter out any zero demand days
for(ii in 1:length(datafr$RT_Demand)){
  if (datafr$RT_Demand[ii] == 0){
    datafr$RT_Demand[ii] <- datafr$RT_Demand[ii-1]
  }
}

#build weekday column
datafr$weekday <- weekdays(as.Date(datafr$Date,"%Y-%m-%d"))

#build week of the year column
datafr$WOY <- as.numeric(format(as.Date(datafr$Date, format="%Y-%m-%d"), "%U"))
```

```
#build month column
datafr$month <- months.Date(as.Date(datafr$Date,"%Y-%m-%d"))

#build year column
datafr$year <- format(as.Date(datafr$Date,"%Y-%m-%d"),'%Y')

Drybulb.mean <- aggregate.data.frame(
  datafr$Dry_Bulb, list(datafr$Hr_End, datafr$weekday,
                        datafr$WOY, datafr$month, datafr$year),FUN=mean)

Demand.mean <- aggregate.data.frame(
  datafr$RT_Demand, list(datafr$Hr_End, datafr$weekday,
                         datafr$WOY, datafr$month, datafr$year),FUN=mean)

range <- Drybulb.mean$Group.2 == "Monday" & Drybulb.mean$Group.3 == 1 &
  Drybulb.mean$Group.4 == "January"
plot(Drybulb.mean$Group.1[range], Drybulb.mean$x[range])


#Determine the average yearly demand for inspection. This should show a yearly tren
Demand.max <- aggregate.data.frame(datafr$DEMAND, list(datafr$year),FUN=max)
Demand.avg <- aggregate.data.frame(datafr$DEMAND, list(datafr$year),FUN=mean)
Demand.min <- aggregate.data.frame(datafr$DEMAND, list(datafr$year),FUN=min)

plot(Demand.max$Group.1, Demand.max$x, xlab="Year", ylab="Demand (MWh)",
     ylim=c(min(Demand.min$x),max(Demand.max$x)+5000), col="red", type="l", lty=1)
lines(Demand.min$Group.1, Demand.min$x, col="blue",lty=1)
lines(Demand.avg$Group.1, Demand.avg$x, col="black",lty=1)
legend('topright',legend=c("Maximum", "Minimum","Average"),
       col=c('red', 'blue','black'), lty=1:1)
grid(nx = NULL, ny = NULL, col = "lightgray", lty = "dotted",
```

```
     lwd = par("lwd"), equilogs = TRUE)


#Determine the average yearly data for inspection
Drybulb.max <- aggregate.data.frame(datafr$DryBulb, list(datafr$year),FUN=max)
Drybulb.avg <- aggregate.data.frame(datafr$DryBulb, list(datafr$year),FUN=mean)
Drybulb.min <- aggregate.data.frame(datafr$DryBulb, list(datafr$year),FUN=min)

plot(Drybulb.max$Group.1, Drybulb.max$x, xlab="Year", ylab="Temperature (F)",
     ylim=c(min(Drybulb.min$x),max(Drybulb.max$x)+30), col="red", type="l", lty=1)
lines(Drybulb.min$Group.1,Drybulb.min$x,col="blue",lty=1)
lines(Drybulb.avg$Group.1,Drybulb.avg$x,col="black",lty=1)
legend('topright',legend=c("Maximum", "Minimum","Average"),
       col=c('red', 'blue','black'), lty=1:1)
grid(nx = NULL, ny = NULL, col = "lightgray", lty = "dotted",
     lwd = par("lwd"), equilogs = TRUE)


#Determine the average yearly data for inspection
Temp.daily <- aggregate.data.frame(datafr$DryBulb, list(datafr$Date),FUN=mean)
day <- unique(datafr$Date)

plot(Temp.daily$Group.1,Temp.daily$x, xlab="Day", ylab="Temperature (F)",
     col="purple", pch=1)
grid(nx = NULL, ny = NULL, col = "lightgray", lty = "dotted",
     lwd = par("lwd"), equilogs = TRUE)


Demand.daily <- aggregate.data.frame(datafr$DEMAND[datafr$year == "2015"],
                                     list(datafr$Date[datafr$year == "2015"]),FUN=mea

plot(Demand.daily$x, xlab="Day", ylab="Demand (MWh)", col="purple", type="l")
```

```
grid(nx = NULL, ny = NULL, col = "lightgray", lty = "dotted",
     lwd = par("lwd"), equilogs = TRUE)


year <- "2016"
idx.1 <- 1
idx.2 <- (idx.1 + 23)

day.name <- c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Su
color <- c('red', 'orange', 'yellow', 'green', 'blue', 'purple', 'black')

Hr_End.mon <- datafr$Hr_End[datafr$year == year & datafr$weekday == "Monday"]
RT_Demand.mon <- datafr$RT_Demand[datafr$year == year & datafr$weekday == "Monday"]

Hr_End.tue <- datafr$Hr_End[datafr$year == year & datafr$weekday == "Tuesday"]
RT_Demand.tue <- datafr$RT_Demand[datafr$year == year & datafr$weekday == "Tuesday"]

Hr_End.wed <- datafr$Hr_End[datafr$year == year & datafr$weekday == "Wednesday"]
RT_Demand.wed <- datafr$RT_Demand[datafr$year == year & datafr$weekday == "Wednesday"

Hr_End.thr <- datafr$Hr_End[datafr$year == year & datafr$weekday == "Thursday"]
RT_Demand.thr <- datafr$RT_Demand[datafr$year == year & datafr$weekday == "Thursday"]

Hr_End.fri <- datafr$Hr_End[datafr$year == year & datafr$weekday == "Friday"]
RT_Demand.fri <- datafr$RT_Demand[datafr$year == year & datafr$weekday == "Friday"]

Hr_End.sat <- datafr$Hr_End[datafr$year == year & datafr$weekday == "Saturday"]
RT_Demand.sat <- datafr$RT_Demand[datafr$year == year & datafr$weekday == "Saturday"]

Hr_End.sun <- datafr$Hr_End[datafr$year == year & datafr$weekday == "Sunday"]
RT_Demand.sun <- datafr$RT_Demand[datafr$year == year & datafr$weekday == "Sunday"]
```

```
plot(Hr_End.mon[idx.1:idx.2],RT_Demand.mon[idx.1:idx.2], xlab="Hour",
     ylim = c(min(datafr$RT_Demand[datafr$year == year]),
              max(datafr$RT_Demand[datafr$year == year])),
     ylab="Demand (MWh)", col="red", type = "l", lty=1)
lines(Hr_End.tue[idx.1:idx.2], RT_Demand.tue[idx.1:idx.2], col="orange")
lines(Hr_End.wed[idx.1:idx.2], RT_Demand.wed[idx.1:idx.2], col="yellow")
lines(Hr_End.thr[idx.1:idx.2], RT_Demand.thr[idx.1:idx.2], col="green")
lines(Hr_End.fri[idx.1:idx.2], RT_Demand.fri[idx.1:idx.2], col="blue")
lines(Hr_End.sat[idx.1:idx.2], RT_Demand.sat[idx.1:idx.2], col="purple")
lines(Hr_End.sun[idx.1:idx.2], RT_Demand.sun[idx.1:idx.2], col="black")
legend('topleft',
       legend=day.name, col= color, lty = 1:1)
grid(nx = NULL, ny = NULL, col = "lightgray", lty = "dotted",
     lwd = par("lwd"), equilogs = TRUE)


year <- "2016"
idx.1 <- 1
idx.2 <- (idx.1 + 23)

day.name <- c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Su
color <- c('red', 'orange', 'yellow', 'green', 'blue', 'purple', 'black')

Hr_End.mon <- datafr$Hr_End[datafr$year == year & datafr$weekday == "Monday"]
temp.mon <- datafr$Dry_Bulb[datafr$year == year & datafr$weekday == "Monday"]

Hr_End.tue <- datafr$Hr_End[datafr$year == year & datafr$weekday == "Tuesday"]
temp.tue <- datafr$Dry_Bulb[datafr$year == year & datafr$weekday == "Tuesday"]

Hr_End.wed <- datafr$Hr_End[datafr$year == year & datafr$weekday == "Wednesday"]
temp.wed <- datafr$Dry_Bulb[datafr$year == year & datafr$weekday == "Wednesday"]
```

```
Hr_End.thr <- datafr$Hr_End[datafr$year == year & datafr$weekday == "Thursday"]
temp.thr <- datafr$Dry_Bulb[datafr$year == year & datafr$weekday == "Thursday"]

Hr_End.fri <- datafr$Hr_End[datafr$year == year & datafr$weekday == "Friday"]
temp.fri <- datafr$Dry_Bulb[datafr$year == year & datafr$weekday == "Friday"]

Hr_End.sat <- datafr$Hr_End[datafr$year == year & datafr$weekday == "Saturday"]
temp.sat <- datafr$Dry_Bulb[datafr$year == year & datafr$weekday == "Saturday"]

Hr_End.sun <- datafr$Hr_End[datafr$year == year & datafr$weekday == "Sunday"]
temp.sun <- datafr$Dry_Bulb[datafr$year == year & datafr$weekday == "Sunday"]

plot(Hr_End.mon[idx.1:idx.2],temp.mon[idx.1:idx.2], xlab="Hour",
     ylim = c(min(datafr$Dry_Bulb[datafr$year == year]),
              max(datafr$Dry_Bulb[datafr$year == year])),
     ylab="Temperature (F)", col="red", type="l", lty = 1)
lines(Hr_End.tue[idx.1:idx.2], temp.tue[idx.1:idx.2], col="orange")
lines(Hr_End.wed[idx.1:idx.2], temp.wed[idx.1:idx.2], col="yellow")
lines(Hr_End.thr[idx.1:idx.2], temp.thr[idx.1:idx.2], col="green")
lines(Hr_End.fri[idx.1:idx.2], temp.fri[idx.1:idx.2], col="blue")
lines(Hr_End.sat[idx.1:idx.2], temp.sat[idx.1:idx.2], col="purple")
lines(Hr_End.sun[idx.1:idx.2], temp.sun[idx.1:idx.2], col="black")
legend('topleft',
       legend=day.name, col= color, lty = 1:1)
grid(nx = NULL, ny = NULL, col = "lightgray", lty = "dotted",
     lwd = par("lwd"), equilogs = TRUE)


Demand <- datafr$DEMAND[datafr$year == "2015"]
Temper <- datafr$DewPnt[datafr$year == "2015"]
plot(Temper,Demand, xlab="Dew Point Temperature", ylab="Demand (MWh)",
     col="purple")
```

```
grid(nx = NULL, ny = NULL, col = "lightgray", lty = "dotted",
     lwd = par("lwd"), equilogs = TRUE)


Demand <- datafr$DEMAND[datafr$year == "2015"]
Temper <- datafr$DryBulb[datafr$year == "2015"]

Demand.Mon <- datafr$DEMAND[datafr$year == "2015" & datafr$weekday == "Monday"]
Drybulb.Mon <- datafr$DryBulb[datafr$year == "2015" & datafr$weekday == "Monday"]

plot(Temper,Demand, xlab="Dry Bulb Temperature", ylab="Demand (MWh)",
     col=ifelse(Temper > 70, 'red', ifelse(Temper < 30, 'blue', 'black')), pch=1)
points(Drybulb.Mon[seq(1,length(Drybulb.Mon),24)],
       Demand.Mon[seq(1,length(Drybulb.Mon),24)], col="yellow", pch=16)
points(Drybulb.Mon[seq(15,length(Drybulb.Mon),24)],
       Demand.Mon[seq(15,length(Drybulb.Mon),24)], col="green", pch=16)
legend('topleft',legend=c("Demand", "Morning", "Afternoon"),
       col=c('black','yellow', 'green'), pch = c(1,16,16))
grid(nx = NULL, ny = NULL, col = "lightgray", lty = "dotted",
     lwd = par("lwd"), equilogs = TRUE)


Demand <- datafr$RT_Demand[datafr$year == "2016"]
Hour <- datafr$Hr_End[datafr$year == "2016"]
Temper <- datafr$Dry_Bulb[datafr$year == "2016"]

scatterplot3d(Temper,Hour,Demand, xlab="Temperature (F)",
              ylab="Hour", zlab="Demand (MWh)",
              color=ifelse(Temper > 70, 'red', ifelse(Temper < 30, 'blue', 'black')),
              pch=1)
grid(nx = NULL, ny = NULL, col = "lightgray", lty = "dotted",
     lwd = par("lwd"), equilogs = TRUE)
```

```
scatterplot3d(Hour,Temper,Demand, xlab="Hour",
              ylab="Temperature (F)", zlab="Demand (MWh)",
              color=ifelse(Temper > 70, 'red', ifelse(Temper < 30, 'blue', 'black')),
              pch=1)
grid(nx = NULL, ny = NULL, col = "lightgray", lty = "dotted",
     lwd = par("lwd"), equilogs = TRUE)


Demand <- datafr$RT_Demand[datafr$year == "2016" & datafr$Hr_End == 1]
Hour <- datafr$Hr_End[datafr$year == "2016" & datafr$Hr_End == 1]
Temper <- datafr$Dry_Bulb[datafr$year == "2016" & datafr$Hr_End == 1]

plot(Temper,Demand, xlab="Temperature (F)", ylab="Demand (MWh)",
     col=ifelse(Temper > 60, 'red', ifelse(Temper < 40, 'blue', 'black')), pch=1)
grid(nx = NULL, ny = NULL, col = "lightgray", lty = "dotted",
     lwd = par("lwd"), equilogs = TRUE)

Monday <- datafr$DEMAND[datafr$year == "2015" & datafr$weekday == "Monday"]
Hour <- datafr$Hour[datafr$year == "2015" & datafr$weekday == "Monday"]

MT.Monday <- datafr$DEMAND[datafr$year == "2015" & datafr$weekday == "Monday"
                           & datafr$DryBulb > 50 & datafr$DryBulb < 60]
MT.Hour <- datafr$Hour[datafr$year == "2015" & datafr$weekday == "Monday"
                       & datafr$DryBulb > 50 & datafr$DryBulb < 60]

HT.Monday <- datafr$DEMAND[datafr$year == "2015" & datafr$weekday == "Monday"
                           & datafr$DryBulb > 80]
HT.Hour <- datafr$Hour[datafr$year == "2015" & datafr$weekday == "Monday"
                       & datafr$DryBulb > 80]
```

```
LT.Monday <- datafr$DEMAND[datafr$year == "2015" & datafr$weekday == "Monday"
                           & datafr$DryBulb < 20]
LT.Hour <- datafr$Hour[datafr$year == "2015" & datafr$weekday == "Monday"
                       & datafr$DryBulb < 20]

plot(Hour, Monday, xlab="Hour", ylab="Demand (MWh)", col="purple", pch=1)
points(MT.Hour, MT.Monday, col="black", pch=16)
points(HT.Hour, HT.Monday, col="red", pch=16)
points(LT.Hour, LT.Monday, col="blue", pch=16)
legend('topleft',legend=c("Demand", "50F < T < 80F", "T > 80F", "T < 20F"),
       col=c('purple', 'black', 'red', 'blue'), pch = c(1,16,16,16))
grid(nx = NULL, ny = NULL, col = "lightgray", lty = "dotted",
     lwd = par("lwd"), equilogs = TRUE)

Drybulb.mean <- aggregate.data.frame(datafr$DryBulb,
                                     list(datafr$Hour, datafr$day, datafr$WOY,
                                          datafr$month, datafr$year),FUN=mean)
Demand.mean <- aggregate.data.frame(datafr$DEMAND,
                                    list(datafr$Hour, datafr$day, datafr$WOY,
                                         datafr$month, datafr$year),FUN=mean)

year <- "2015"
month <- "January"
day <- c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"
hour <- 1:24

class.1 <- 0
class.2 <- 0
for (ii in 1:length(day)){
  for (jj in hour){
    time <- datafr$Hour[datafr$year == year
                        & datafr$month == month
```

```
                    & datafr$weekday == day[ii]
                    & datafr$Hour == jj]

date <- datafr$Date[datafr$year == year
                    & datafr$month == month
                    & datafr$weekday == day[ii]
                    & datafr$Hour == jj]

demand <- datafr$DEMAND[datafr$year == year
                        & datafr$month == month
                        & datafr$weekday == day[ii]
                        & datafr$Hour == jj]

temp <- datafr$DryBulb[datafr$year == year
                       & datafr$month == month
                       & datafr$weekday == day[ii]
                       & datafr$Hour == jj]

demand.High <- max(demand)
demand.Low <- min(demand)
demand.Mid <- (demand.High + demand.Low)/2

temp.High <- max(temp)
temp.Low <- min(temp)
temp.Mid <- (temp.High + temp.Low)/2

for (kk in 1:length(temp)){
  if (demand[kk] <= demand.Mid){
    datafr$class[datafr$year == year
                 & datafr$month == month
                 & datafr$weekday == day[ii]
                 & datafr$Hour == jj
```

```
                    & datafr$Date == date[kk]] <- "normal"
      } else{
        datafr$class[datafr$year == year
                    & datafr$month == month
                    & datafr$weekday == day[ii]
                    & datafr$Hour == jj
                    & datafr$Date == date[kk]] <- "high"
      }
    }
  }
}
```

C SVM Model

```
cat("\014") #Clear consule

## Load libraries/packages
library(e1071)
library(rpart)
library(openxlsx)  # Read and write Excel files

## Read data from csv summary created by (xlsx2csv.R)
setwd("C:\\Users\\Tylor\\Google Drive\\R\\ETM510_Forcast\\Data")
file.name <- "Demand_Summary.csv"
datafr <- read.csv(file.name, header = TRUE, sep = ",", quote = "\"",
         dec = ".", fill = TRUE, comment.char = "")
# find the mean hourly temperature from the last 10 years of data
Drybulb.mean <- aggregate.data.frame(datafr$Dry_Bulb,
                                     list(datafr$Hr_End, datafr$weekday,
                                          datafr$WOM, datafr$month),FUN=mean)
# find the mean hourly demand from the last 10 years of data
```

```
Demand.mean <- aggregate.data.frame(datafr$RT_Demand,
                                    list(datafr$Hr_End, datafr$weekday,
                                         datafr$WOM, datafr$month),FUN=mean)

## Create training set for the data. I used the most recent year of data to scale
## the mean demand using MAPE comparison
year <- 2016
date <- unique(datafr$Date[datafr$year == year])
for (ii in 1:length(date)){
  DOM <- as.numeric(format(as.Date(date[ii], format="%Y-%m-%d"), "%d"))
  wkcnt <- ceiling(DOM/7)
  wkday <- weekdays(as.Date(date[ii],"%Y-%m-%d"))
  month <- months.Date(as.Date(date[ii],"%Y-%m-%d"))
  Demand.model <- Demand.mean$x[Demand.mean$Group.2 == wkday &
                                Demand.mean$Group.3 == wkcnt &
                                Demand.mean$Group.4 == month]
  Demand.actual <- datafr$RT_Demand[datafr$Date == date[ii]]
  scale <- seq(0.75,1.25,0.01)
  MAPE <- 0
  for (jj in 1:NROW(scale)){
    MAPE[jj] <- abs(mean((Demand.actual-scale[jj]*Demand.model)/Demand.actual))*100
  }
  x <- scale[which.min(MAPE)]
  datafr$Model[datafr$Date == date[ii]] <- x*Demand.model
  datafr$Error[datafr$Date == date[ii]] <- mean((Demand.actual-x*Demand.model)/Demand
}


## Start the test year on the same weekday to ensure accurate comparison
yr.past <- 2016
yr.predct <- 2017
mn.predct <- "January"
```

```
wkday.start <- datafr$weekday[datafr$year == yr.predct &
                              datafr$month == mn.predct &
                              datafr$DOM == 1]
day.start <- datafr$DOM[datafr$year == yr.past &
                        datafr$month == mn.predct & datafr$WOM == 1 &
                        datafr$weekday == wkday.start]

range <- datafr$year == yr.past & datafr$month == mn.predct &
         datafr$DOM >= day.start[1]
range2 <- datafr$year == yr.predct & datafr$month == mn.predct
Demand.previous <- datafr$RT_Demand[range]
Demand.model <- datafr$Model[range]
Demand.actual <- datafr$RT_Demand[range2]




## SVM Model
idx <- datafr$year == 2016
x.train <- data.frame(datafr$Dry_Bulb[idx], datafr$Hr_End[idx], datafr$DOM[idx], data
                      datafr$weekday[idx], datafr$month[idx], datafr$RT_Demand[idx])
names(x.train) <- c("Temp", "Hr_End", "DOM", "WOM", "weekday", "month", "Model")

idx <- datafr$year == 2017
x.test<- data.frame(datafr$Dry_Bulb[idx], datafr$Hr_End[idx], datafr$DOM[idx], datafr
                    datafr$weekday[idx], datafr$month[idx])
names(x.test) <- c("Temp", "Hr_End", "DOM", "WOM", "weekday", "month")

svm.model <- svm(Model~., data=x.train, gamma=1)
svm.pred <- predict(svm.model, newdata=x.test)

plot(Demand.previous[1:(24*7)], type = "o", lty=1, pch=1, col="black",
```

```
     ylim = c(10000, 20000), xlab="Hour", ylab="Demand (MWh)")
lines(Demand.actual[1:(24*7)], type = "o", lty=1, pch=2, col="red")
lines(Demand.model[1:(24*7)], type = "o", lty=1, pch=3, col="blue")
lines(svm.pred[1:(24*7)], type = "o", lty=1, pch=4, col="green")
legend('topright',legend=c("2016", "2017","Mean","SVM"),
       col=c('black', 'red','blue', 'green'), lty=1:1, pch=c(1,2,3,4))
grid(nx = NULL, ny = NULL, col = "lightgray", lty = "dotted",
     lwd = par("lwd"), equilogs = TRUE)

## SVM Model without temperature
idx <- datafr$year == 2016
x.train <- data.frame(datafr$Hr_End[idx], datafr$DOM[idx], datafr$WOM[idx],
                      datafr$weekday[idx], datafr$month[idx], datafr$Model[idx])
names(x.train) <- c("Hr_End", "DOM", "WOM", "weekday", "month", "Model")

idx <- datafr$year == 2017
x.test<- data.frame(datafr$Hr_End[idx], datafr$DOM[idx], datafr$WOM[idx],
                    datafr$weekday[idx], datafr$month[idx])
names(x.test) <- c("Hr_End", "DOM", "WOM", "weekday", "month")

svm.model <- svm(Model~., data=x.train, gamma=1)
svm.pred <- predict(svm.model, newdata=x.test)

plot(Demand.previous[1:(24*7)], type = "o", lty=1, pch=1, col="black",
     ylim = c(10000, 20000), xlab="Hour", ylab="Demand (MWh)")
lines(Demand.actual[1:(24*7)], type = "o", lty=1, pch=2, col="red")
lines(Demand.model[1:(24*7)], type = "o", lty=1, pch=3, col="blue")
lines(svm.pred[1:(24*7)], type = "o", lty=1, pch=4, col="green")
legend('topright',legend=c("2016", "2017","Mean","SVM"),
       col=c('black', 'red','blue', 'green'), lty=1:1, pch=c(1,2,3,4))
grid(nx = NULL, ny = NULL, col = "lightgray", lty = "dotted",
     lwd = par("lwd"), equilogs = TRUE)
```