ADS506 - Group Project - ERCOT Load Forecasting

Please see code blocks below which have been used to load, prep, transform and plot our data.

This will serve as our initial pre-processing, which we can further analyze for our modelling phase and our paper.

Further analysis will also be performed as we dive deeper into our paper and final code.

```
load17.df <- read_excel('Native_Load_2017.xlsx')
load18.df <- read_excel('Native_Load_2018.xlsx')
load19.df <- read_excel('Native_Load_2019.xlsx')
load20.df <- read_excel('Native_Load_2020.xlsx')
load21.df <- read_excel('Native_Load_2021.xlsx')
```

```
## The load dataset for 2017 had a space in the Timestamp column, preventing the binding of the four yea
colnames(load17.df)[1] <- "HourEnding"

## Using bind_rows from tidyverse, the data from 2017-2020 was merged.
load_4_year <- bind_rows(load17.df,load18.df,load19.df,load20.df)
sum(is.na(load_4_year))
```

```
## [1] 0
```

```
## Using strptime, the time variable data type was set.
load_4_year$HourEnding <- as.POSIXct(strptime(load_4_year$HourEnding, format="%m/%d/%Y %H:%M"))

#Remove NA's which arise due to issues with daylight savings (1 record in each year)
load_4_year <- na.omit(load_4_year)

## Converting to a time series
ts_load <- xts(load_4_year, load_4_year$HourEnding)
ts_load <- ts_load[,-1]
head(ts_load)
```

```
##                     COAST       EAST        FWEST      NORTH       NCENT
## 2017-01-01 01:00:00 " 8791.790" " 896.7463" "1997.718" " 683.6220" " 9239.153"
## 2017-01-01 02:00:00 " 8569.708" " 865.9306" "1997.781" " 677.9694" " 9104.997"
## 2017-01-01 03:00:00 " 8326.426" " 839.0512" "1993.699" " 671.9989" " 8988.035"
## 2017-01-01 04:00:00 " 8137.497" " 822.8293" "1995.541" " 675.2680" " 8979.148"
## 2017-01-01 05:00:00 " 8011.870" " 814.0162" "1995.254" " 663.6199" " 9033.548"
## 2017-01-01 06:00:00 " 7978.100" " 823.4194" "1981.352" " 669.7480" " 9195.158"
##                     SOUTH      SCENT       WEST        ERCOT
## 2017-01-01 01:00:00 "2366.633" " 4490.781" " 954.1929" "29420.64"
## 2017-01-01 02:00:00 "2332.745" " 4370.657" " 951.0252" "28870.81"
## 2017-01-01 03:00:00 "2237.506" " 4210.650" " 944.3577" "28211.72"
## 2017-01-01 04:00:00 "2178.102" " 4088.713" " 943.1887" "27820.29"
## 2017-01-01 05:00:00 "2133.954" " 4021.757" " 954.9379" "27628.96"
## 2017-01-01 06:00:00 "2122.017" " 4045.300" " 972.5624" "27787.66"
```

```r
## A function has been created to plot data once appropriate transformations have been made.

plot_data = function(df, title){

  plot_melt = melt(df, na.rm = FALSE, value.name = 'Load', id = 'HourEnding')
  names(plot_melt)[2] <- "Region"

  plt = ggplot(data = plot_melt, mapping = aes(x=HourEnding, y=Load, colour = Region)) +
  geom_line(size = 0.1, alpha=0.4) +
  ggdark::dark_mode() +
  theme(legend.position = 'bottom', ) +
  ggtitle(paste('ERCOT Actual Power Load by Region - ' , title)) +
  theme(plot.title = element_text(hjust = 0.5)) +
  ylab("Electrical Load (MW)") +
  xlab("Hour Ending") +
  theme(panel.grid.major = element_blank()) +
  guides(shape = guide_legend(override.aes = list(size = 1)),
              color = guide_legend(override.aes = list(size = 1))) +
  theme(legend.title = element_text(size = 8),
             legend.text  = element_text(size = 8),
             legend.key.size = unit(1, "lines"))

  return(plt)}

plot_data(load_4_year, 'Raw Data')
```

```r
std = load_4_year %>%
  mutate_if(is.numeric, scale)

plot_data(std, 'Standardized')
```

```r
## Applying differencing and detrending to remove seasonality and make data stationary

dload <- data.frame(lapply(load_4_year[,-1], diff))
dload["HourEnding"] <- load_4_year$HourEnding[2:35060]

plot_data(dload, 'Differenced')
```

```
## Building out the min/max scaler to normalize the data as a function
normalize <- function(x) {
 (x-min(x))/(max(x)-min(x))
}

## Removing the POSIXlt column to normalize
loadnorm <- (load_4_year[,-1])

## Running the new DF through the normalization function
loadnorm <- normalize(loadnorm)

## Adding the POSIXlt column back on
loadnorm["HourEnding"] <- load_4_year$HourEnding

## Plotting the data
plot_data(loadnorm, "Normalized (Min/Max)")
```

It is clear that a min max normalization scalar is not a usable option as it does not account for the mean. The graph is still visually the same.