Steel Industry Energy Consumption Analysis

Adhithya

10/31/2022

#Description

The dataset contains feature vectors from 35040 different instances. The data was acquired from Gwangyang, South Korea's DAEWOO Steel Co. Ltd. This company manufactures iron plates, steel sheets, and coils. The dataset's "Usage kWh" column contains data on the energy needed to produce different kinds of coils and plates, as well as the usage of energy consumed to produce different kinds of loads, including light, medium, and maximum loads.

Additionally, the equipment's manufacturing date and time, as well as the specific day of the week and whether it was a workday or a weekend, are all included in this dataset. For the manufacturing sectors, where a variety of machinery, including generators, motors, lathes, synchronous machines, transformers, electrostatic machines, and so forth are used, the power factor is a major element.The machines are successfully using electricity and operating effectively when they have a unity power factor. The load current has a "leading power factor" if it is capacitive, as opposed to a "lagging power factor" if it is inductive.

The values for both leading and lagging power factors may be found in the dataset. In this situation, a leading power factor can be rectified by adding inductive loads, and a lagging power factor can be solved by adding capacitive loads. The data on power use is stored on a cloud-based platform. The Korea Electric Power Corporation's website has data on the energy use of the sector.

#Objective

We want to know how much energy is typically used for each type of load on a daily and annual basis, as well as which month produces the most CO2 emissions. We are attempting to determine which load—the inductive load, which produces lagging reactive power, or the capacitive load, which produces leading reactive power—is responsible for the greatest amount of reactive power loss because it is a big industrial problem. Additionally, we're attempting to illustrate the many load types that use the most energy as well as the industry's breakdown of every day vs. weekend energy consumption.

```
sessionInfo()
```

```
## R version 4.2.1 (2022-06-23 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 26100)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
```

```
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## loaded via a namespace (and not attached):
##  [1] compiler_4.2.1    fastmap_1.1.1     cli_3.6.3         tools_4.2.1
##  [5] htmltools_0.5.6.1 rstudioapi_0.15.0 yaml_2.3.7        rmarkdown_2.25
##  [9] knitr_1.45        xfun_0.40         digest_0.6.29     rlang_1.1.1
## [13] evaluate_0.22
```

```
library("ggplot2")
```

```
## Warning: package 'ggplot2' was built under R version 4.2.3
```

```
library("readxl")
```

```
## Warning: package 'readxl' was built under R version 4.2.3
```

```
library("dplyr")
```

```
## Warning: package 'dplyr' was built under R version 4.2.3

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library("gridExtra")
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library("grid")
library("tidyr")
```

```
## Warning: package 'tidyr' was built under R version 4.2.3
```

```
library("readr")
```

```
## Warning: package 'readr' was built under R version 4.2.3
```

```r
library("treemapify")
```

```
## Warning: package 'treemapify' was built under R version 4.2.3
```

```r
library("lubridate")
```

```
## Warning: package 'lubridate' was built under R version 4.2.3
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```r
library("tidyverse")
```

```
## Warning: package 'tidyverse' was built under R version 4.2.3
```

```
## Warning: package 'tibble' was built under R version 4.2.3
```

```
## Warning: package 'purrr' was built under R version 4.2.3
```

```
## Warning: package 'stringr' was built under R version 4.2.3
```

```
## Warning: package 'forcats' was built under R version 4.2.3
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats 1.0.0     v stringr 1.5.0
## v purrr   1.0.2     v tibble  3.2.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x gridExtra::combine() masks dplyr::combine()
## x dplyr::filter()      masks stats::filter()
## x dplyr::lag()         masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library("treemap")
```

```
## Warning: package 'treemap' was built under R version 4.2.3
```

```r
library("histogram")
```

```r
#The below Code is use to Read the CSV file into the R Studio
steel_data <- read_csv('Steel_industry_data.csv', show_col_types = FALSE)
```

#Data Cleaning

The first step is to clean the data after reading the dataset. Column "NSM" that seems to be unnecessary will be removed. After that, we removed the irrelevant data from the dataset in order to maintain a clean working directory.

```r
library(dplyr)

# Step 1: Replace empty strings with NA in character columns
steel_data <- steel_data %>%
  mutate(across(where(is.character), ~ na_if(., "")))

# Step 2: Handle the 'Usage_kWh' column
steel_data <- steel_data %>%
  mutate(Usage_kWh = as.numeric(Usage_kWh)) %>%   # Ensure Usage_kWh is numeric
  mutate(Usage_kWh = na_if(Usage_kWh, NA))        # Replace any 'NA' or inappropriate values with NA

# Optional: Check the structure of the updated data
str(steel_data)
```

```
## tibble [35,040 x 11] (S3: tbl_df/tbl/data.frame)
##  $ date                                : chr [1:35040] "01-01-2018 00:15" "01-01-2018 00:30" "01-01-2
##  $ Usage_kWh                           : num [1:35040] 3.17 4 3.24 3.31 3.82 3.28 3.6 3.6 3.28 3.78
##  $ Lagging_Current_Reactive.Power_kVarh: num [1:35040] 2.95 4.46 3.28 3.56 4.5 3.56 4.14 4.28 3.64 4
##  $ Leading_Current_Reactive_Power_kVarh: num [1:35040] 0 0 0 0 0 0 0 0 0 0 ...
##  $ CO2.tCO2.                           : num [1:35040] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Lagging_Current_Power_Factor        : num [1:35040] 73.2 66.8 70.3 68.1 64.7 ...
##  $ Leading_Current_Power_Factor        : num [1:35040] 100 100 100 100 100 100 100 100 100 100 ...
##  $ NSM                                 : num [1:35040] 900 1800 2700 3600 4500 5400 6300 7200 8100 9
##  $ WeekStatus                          : chr [1:35040] "Weekday" "Weekday" "Weekday" "Weekday" ...
##  $ Day_of_week                         : chr [1:35040] "Monday" "Monday" "Monday" "Monday" ...
##  $ Load_Type                           : chr [1:35040] "Light_Load" "Light_Load" "Light_Load" "Light_
```

#Data Exploration

To understand the dataset's contents and examine its many features, we are employing data exploration analysis.

```r
 #Printing dataset characteristics
head(steel_data)
```

```
## # A tibble: 6 x 11
##   date         Usage_kWh Lagging_Current_Reac~1 Leading_Current_Reac~2 CO2.tCO2.
##   <chr>            <dbl>                  <dbl>                  <dbl>     <dbl>
## 1 01-01-2018 ~      3.17                   2.95                      0         0
## 2 01-01-2018 ~      4                      4.46                      0         0
## 3 01-01-2018 ~      3.24                   3.28                      0         0
## 4 01-01-2018 ~      3.31                   3.56                      0         0
## 5 01-01-2018 ~      3.82                   4.5                       0         0
## 6 01-01-2018 ~      3.28                   3.56                      0         0
## # i abbreviated names: 1: Lagging_Current_Reactive.Power_kVarh,
## #   2: Leading_Current_Reactive_Power_kVarh
## # i 6 more variables: Lagging_Current_Power_Factor <dbl>,
## #   Leading_Current_Power_Factor <dbl>, NSM <dbl>, WeekStatus <chr>,
## #   Day_of_week <chr>, Load_Type <chr>
```

```r
summary(steel_data)
```

```
##      date             Usage_kWh      Lagging_Current_Reactive.Power_kVarh
##  Length:35040       Min.   :  0.00   Min.   : 0.00
##  Class :character   1st Qu.:  3.20   1st Qu.: 2.30
##  Mode  :character   Median :  4.57   Median : 5.00
##                     Mean   : 27.39   Mean   :13.04
##                     3rd Qu.: 51.24   3rd Qu.:22.64
##                     Max.   :157.18   Max.   :96.91
##  Leading_Current_Reactive_Power_kVarh   CO2.tCO2.
##  Min.   : 0.000                       Min.   :0.00000
##  1st Qu.: 0.000                       1st Qu.:0.00000
##  Median : 0.000                       Median :0.00000
##  Mean   : 3.871                       Mean   :0.01152
##  3rd Qu.: 2.090                       3rd Qu.:0.02000
##  Max.   :27.760                       Max.   :0.07000
##  Lagging_Current_Power_Factor Leading_Current_Power_Factor      NSM
##  Min.   :  0.00               Min.   :  0.00               Min.   :    0
##  1st Qu.: 63.32               1st Qu.: 99.70               1st Qu.:21375
##  Median : 87.96               Median :100.00               Median :42750
##  Mean   : 80.58               Mean   : 84.37               Mean   :42750
##  3rd Qu.: 99.02               3rd Qu.:100.00               3rd Qu.:64125
##  Max.   :100.00               Max.   :100.00               Max.   :85500
##   WeekStatus        Day_of_week         Load_Type
##  Length:35040       Length:35040       Length:35040
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
```

```
sapply(steel_data, class)
```

```
##                                date                            Usage_kWh
##                         "character"                            "numeric"
## Lagging_Current_Reactive.Power_kVarh Leading_Current_Reactive_Power_kVarh
##                           "numeric"                            "numeric"
##                           CO2.tCO2.         Lagging_Current_Power_Factor
##                           "numeric"                            "numeric"
##         Leading_Current_Power_Factor                                 NSM
##                           "numeric"                            "numeric"
##                          WeekStatus                          Day_of_week
##                         "character"                          "character"
##                           Load_Type
##                         "character"
```

#Data Visualization

In order to comprehend the trends and patterns in the data, we are utilizing data visualization to give a graphical depiction. In order to see the different trends of the data for the year 2018, we apply data visualization techniques such as histograms, bar graphs, tree maps, scatter plots, stacked bar graphs, and line graphs.

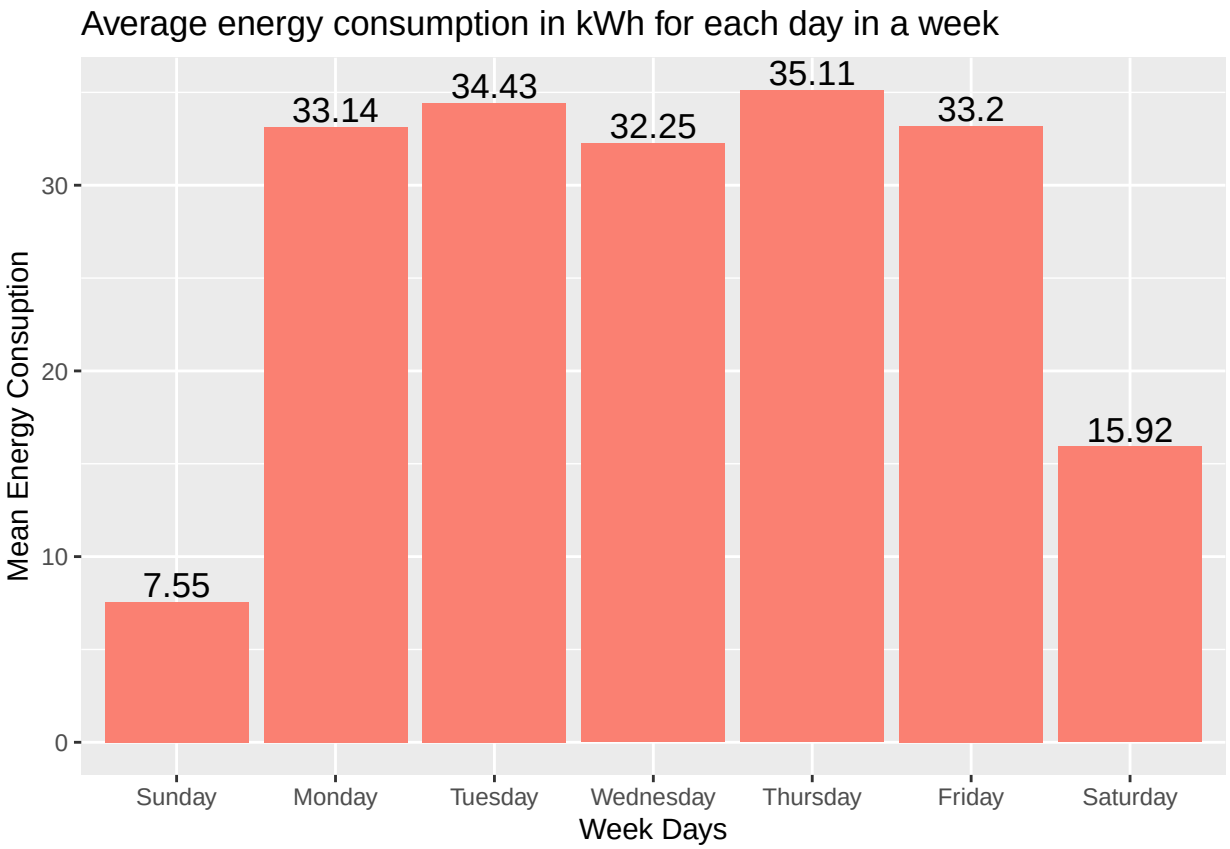
These techniques include looking at the average amount of energy used for each type of load on a daily and annual basis, Co2 emission throughout the year, reactive power loss, different types of loads that consume the most energy, and the industry's weekdays versus weekend energy use ratio.

1] Average energy consumption in kWh for each day in a week

```
#Steel1 is the Subset of the data in which the Data to be plot is Stored
#Grouping the Data by the Days of Week and Load Type
steel_data1<-steel_data%>%
  group_by(Day_of_week)%>%
  summarize(AvgUsage_kWh = mean(Usage_kWh))

steel_data1$AvgUsage_kWh_roff <- round(steel_data1$AvgUsage_kWh, digits = 2)

steel_data1$Day_of_week <- factor(steel_data1$Day_of_week, levels= c("Sunday", "Monday",
                      "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"))
steel_data1[order(steel_data1$Day_of_week), ]
```

```
## # A tibble: 7 x 3
##   Day_of_week AvgUsage_kWh AvgUsage_kWh_roff
##   <fct>              <dbl>             <dbl>
## 1 Sunday              7.55              7.55
## 2 Monday             33.1              33.1
## 3 Tuesday            34.4              34.4
## 4 Wednesday          32.3              32.2
## 5 Thursday           35.1              35.1
## 6 Friday             33.2              33.2
## 7 Saturday           15.9              15.9
```

```
graph_1<-ggplot(data=steel_data1, aes(x=Day_of_week, y=AvgUsage_kWh_roff)) +
  geom_bar(stat="identity",fill="salmon",position = "dodge")+
  geom_text(aes(label = AvgUsage_kWh_roff), vjust = -0.2, size = 4.5,
            position = position_dodge(0.9))+
  labs(title="Average energy consumption in kWh for each day in a week",x="Week Days", y= "Mean Energy
graph_1
```



We can see from this bar graph, which shows the weekly average energy consumption in kWh, that weekdays use more energy than weekends do since there are fewer people working on Saturdays and Sundays and less energy is used for production.

As a result, The days of the week that need the greatest energy are Tuesday, Thursday, and then Monday. The days with the closest equivalence in energy use are Wednesday and Friday.

2] Lagging current reactive power vs leading current reactive power throughout the year

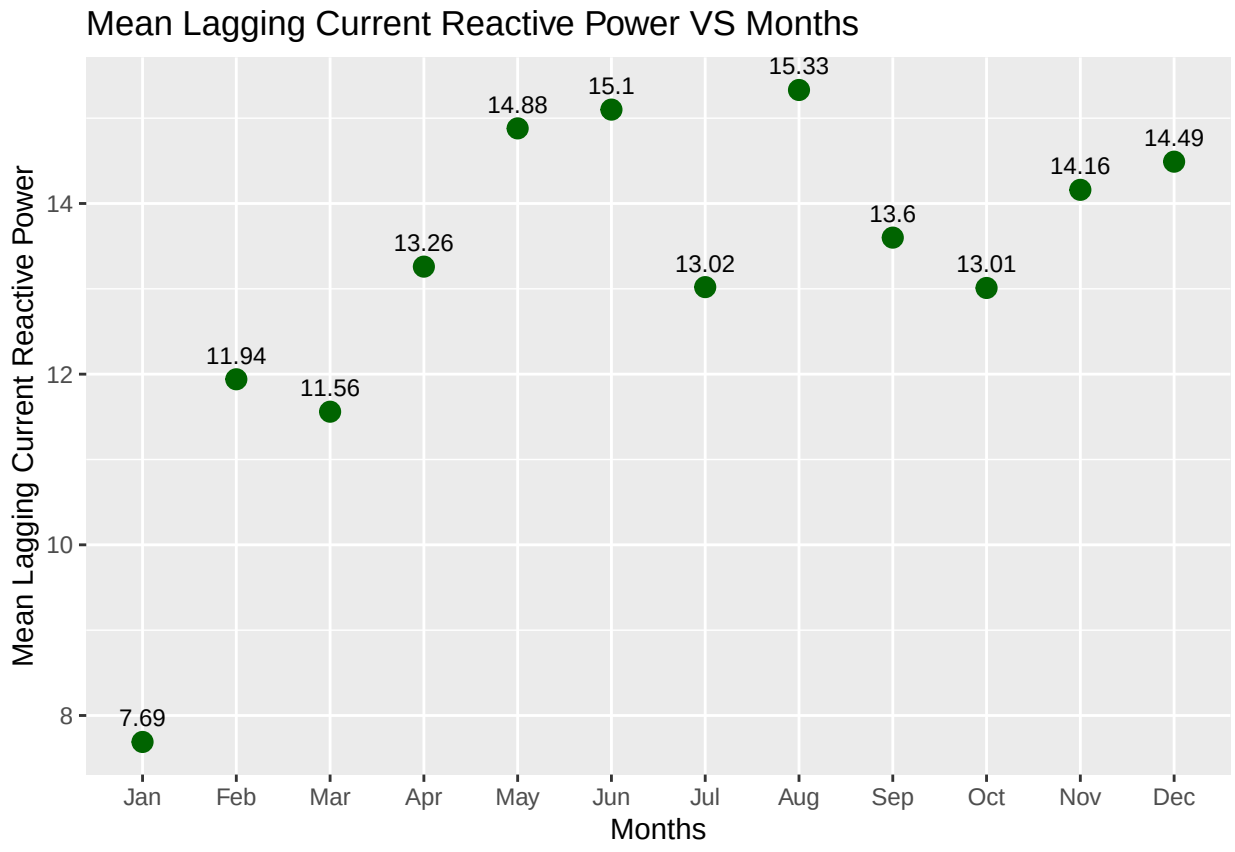
#Lagging current reactive power

```
# Converting the datetime on date format
Date_format <- as.Date(mdy_hms(steel_data$date))
```

```
## Warning: 21216 failed to parse.
```

```
# extracting the months from the date
month_x2 <- format(Date_format, "%m")
steel_data$months <- month_x2

#The Below code is use to Show Mean Mean Lagging Current Reactive Power VS Months.
steel_data2 <- steel_data%>%
  select(Lagging_Current_Reactive.Power_kVarh, months) %>%
```

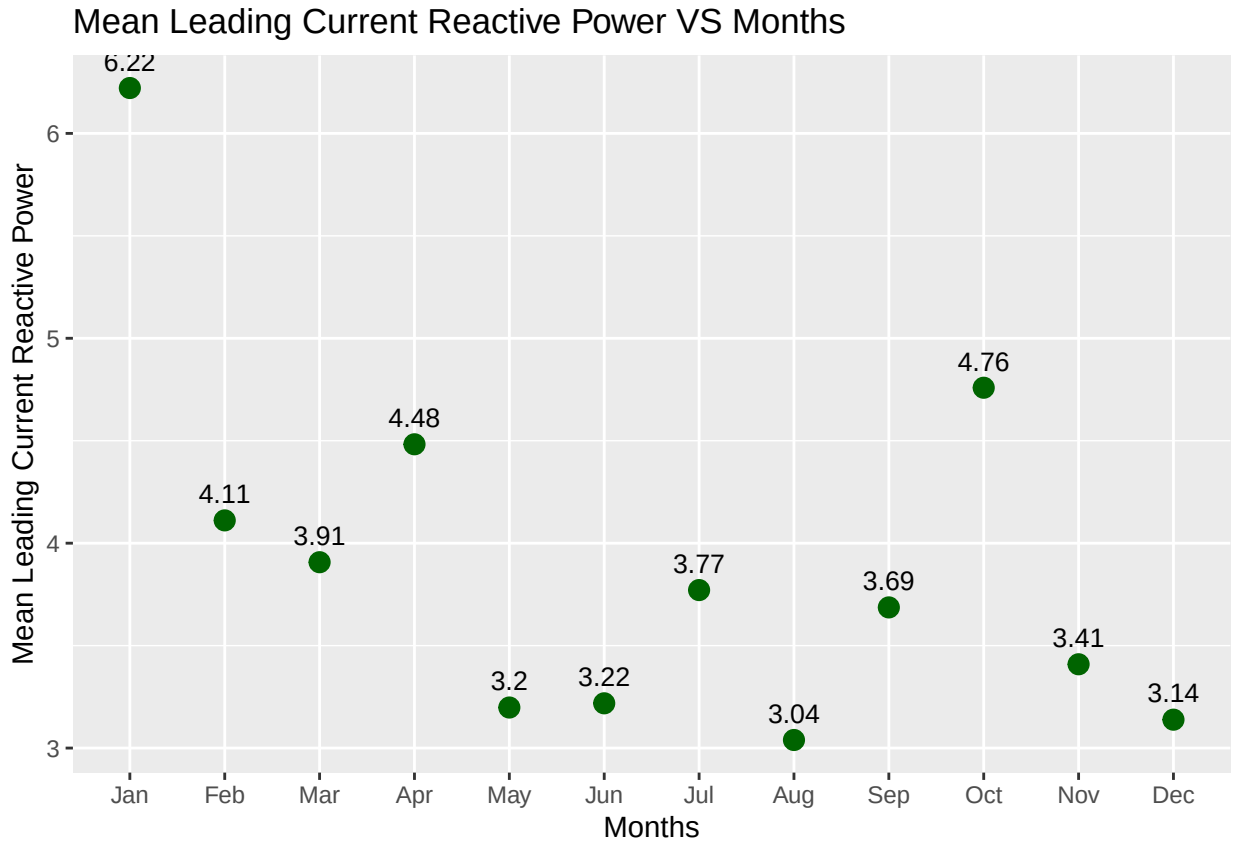
```
  group_by(months)%>%
  summarize(Avglagging = mean(Lagging_Current_Reactive.Power_kVarh))%>%
  na.omit

month_xInt2 <- as.integer(steel_data2$months)
steel_data2$months<- month.abb[month_xInt2]

steel_data2$Avglagging_roff <- round(steel_data2$Avglagging, digits = 2)

graph_2<- ggplot(steel_data2, aes(x=fct_inorder(months), y=Avglagging_roff)) +
  geom_point(shape=20, color="dark green",size=5)+
      geom_text(aes(label = Avglagging_roff), vjust = -0.9, size = 3,
            position = position_dodge(0.9))+
  labs(title="Mean Lagging Current Reactive Power VS Months",x="Months", y= "Mean Lagging Current Reacti
graph_2
```



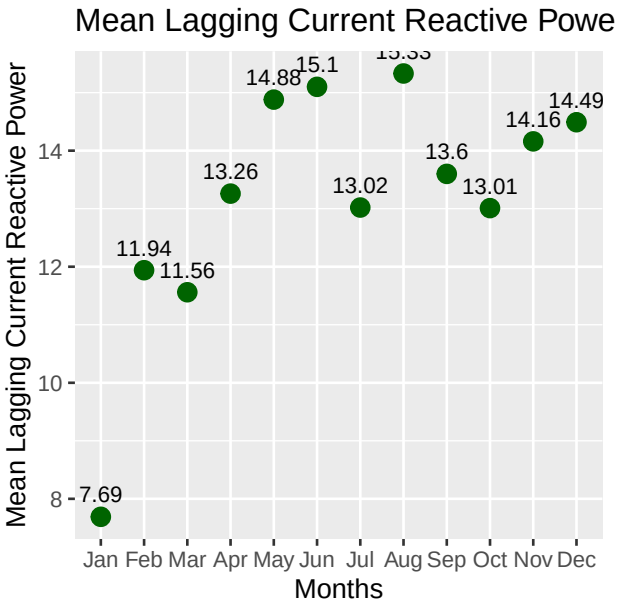
#Leading current reactive power

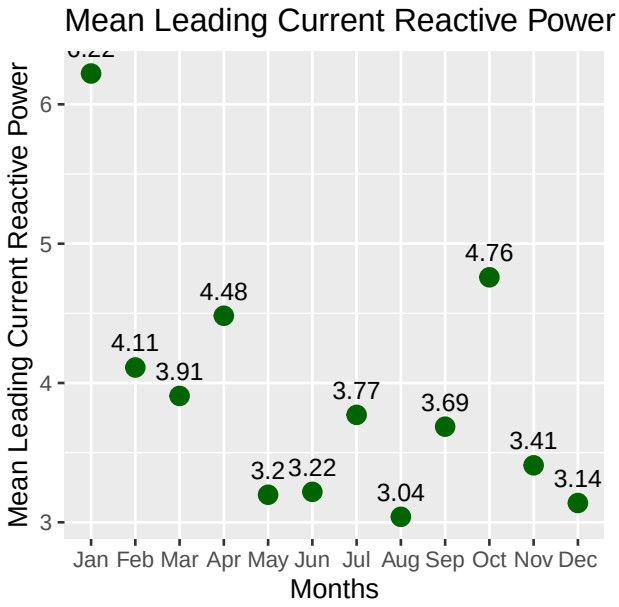
```
#The Below code is use to show Mean Leading Current Reactive Power
steel_data3 <- steel_data%>%
  select(Leading_Current_Reactive_Power_kVarh, months) %>%
  group_by(months)%>%
  summarize(Avgleading = mean(Leading_Current_Reactive_Power_kVarh))%>%
  na.omit

month_xInt3 <- as.integer(steel_data3$months)
```

```
steel_data3$months<- month.abb[month_xInt3]
steel_data3$Avgleading_roff <- round(steel_data3$Avgleading, digits = 2)


graph_3<- ggplot(steel_data3, aes(x=fct_inorder(months), y=Avgleading)) +
  geom_point(shape=20, color="dark green",size=5)+
    geom_text(aes(label = Avgleading_roff), vjust = -0.9, size = 3.5,
              position = position_dodge(0.9))+
  labs(title="Mean Leading Current Reactive Power VS Months",x="Months", y= "Mean Leading Current Reacti
graph_3
```




```
#The Below Code is Use to Group 2 graph together
grid.arrange(graph_2,graph_3,ncol=2,nrow=2)
```

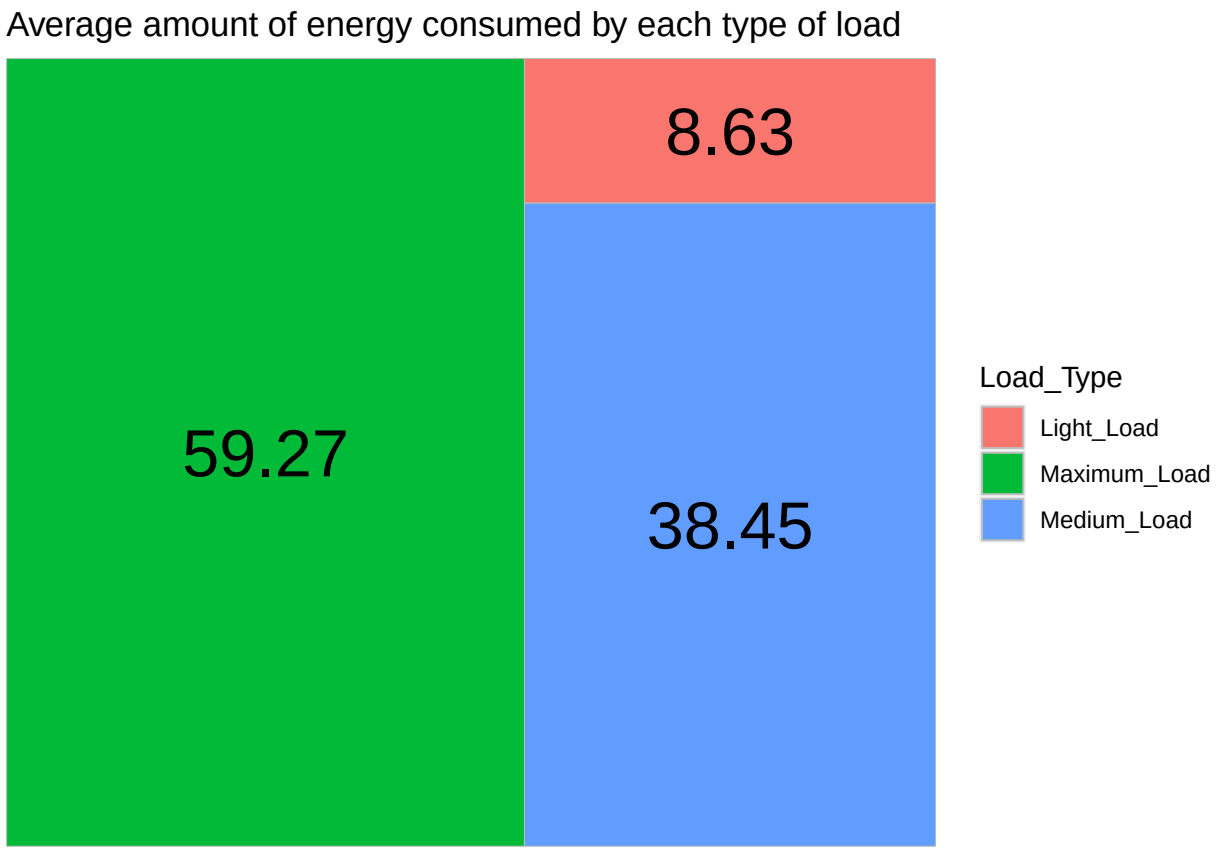




3] Average amount of energy consumed by each type of load

```
#The Below Code is use to Plot a Tree Map Which tells us Different Types of Load
#According to the Avg  Usage kWh

steel_data4<-steel_data%>%
  group_by(Load_Type)%>%
  summarize(AvgUsage_kWh = mean(Usage_kWh))

steel_data4$AvgUsage_kWh_roff <- round(steel_data4$AvgUsage_kWh, digits = 2)
```

```
ggplot(steel_data4, aes(area = AvgUsage_kWh_roff, fill = Load_Type, label = AvgUsage_kWh_roff)) +
  geom_treemap() +
  geom_treemap_text(colour = "black",
                    place = "centre",
                    size = 25,
                    )+
labs(title="Average amount of energy consumed by each type of load ")
```



Using the tree map, we can identify the load that uses the most energy as well as the real average energy consumption.

It turns out that the Maximum load uses the most energy, which is 59.27 kWh, compared to the other two types of loads. A medium load uses 38.45 kWh, but a light load only uses 8.63 kWh.

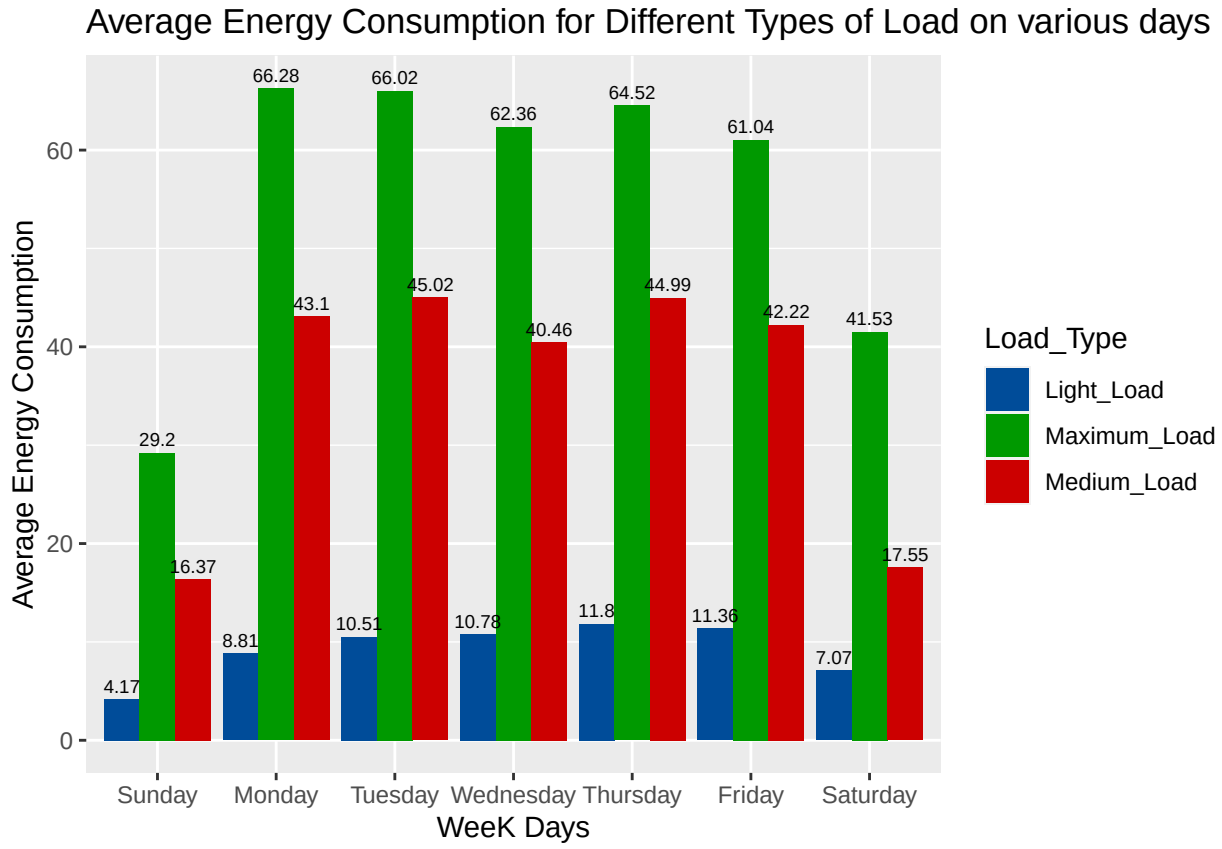
4] Energy usage by each type of load throughout the week and the ratio for weekends and weekdays

```
#The below Code Is use to Show Mean Energy Consumption According to Different Types of Load
steel_data5<-steel_data%>%
  group_by(Day_of_week,Load_Type)%>%
  summarize(AvgUsage_kWh = mean(Usage_kWh), .groups = 'drop')
steel_data5$AvgUsage_kWh_roff <- round(steel_data5$AvgUsage_kWh, digits = 2)
steel_data5$Day_of_week <- factor(steel_data5$Day_of_week, levels= c("Sunday", "Monday",
```

```
                                                          "Tuesday", "Wednesday", "Thursday", "Friday
steel_data5[order(steel_data1$Day_of_week), ]
```

```
## # A tibble: 7 x 4
##   Day_of_week Load_Type    AvgUsage_kWh AvgUsage_kWh_roff
##   <fct>       <chr>               <dbl>             <dbl>
## 1 Monday      Light_Load           8.81              8.81
## 2 Friday      Maximum_Load        61.0              61.0
## 3 Monday      Medium_Load         43.1              43.1
## 4 Saturday    Light_Load           7.07              7.07
## 5 Monday      Maximum_Load        66.3              66.3
## 6 Friday      Light_Load          11.4              11.4
## 7 Friday      Medium_Load         42.2              42.2
```

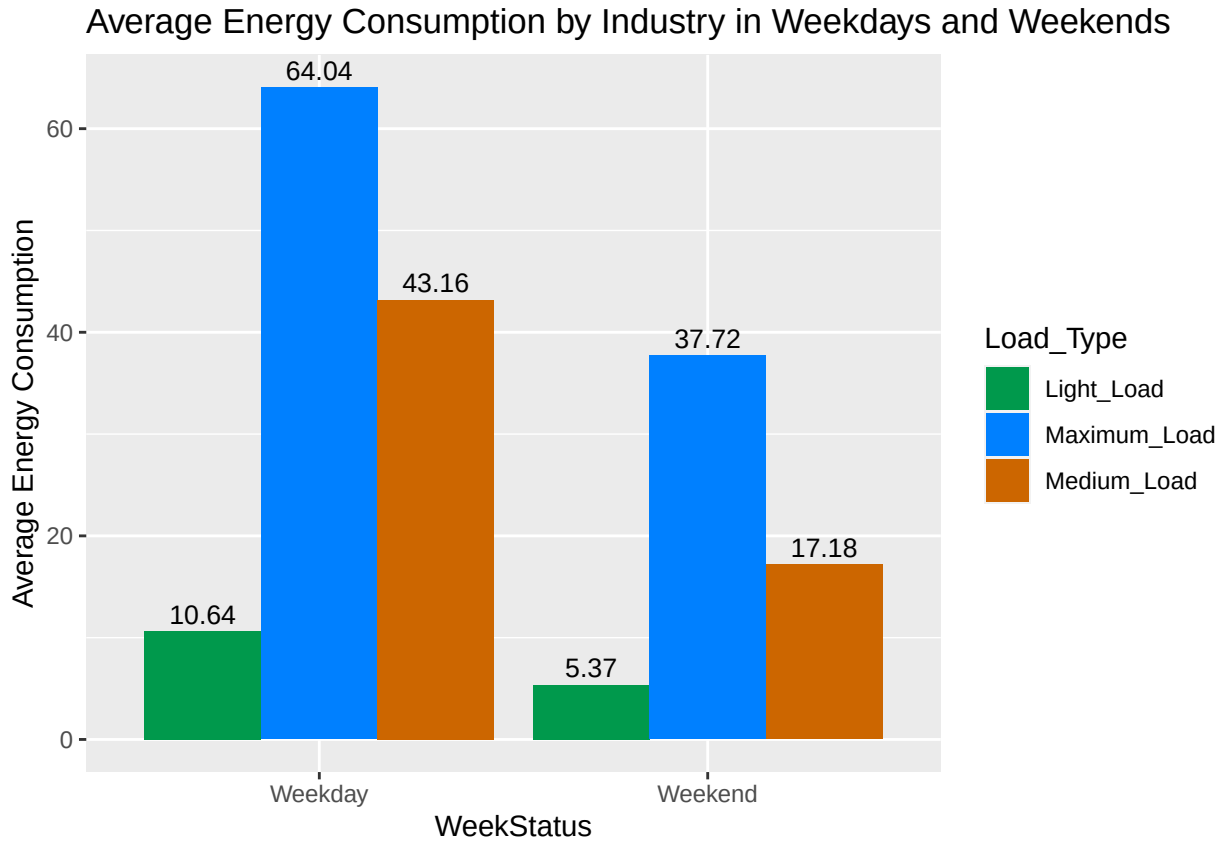
```
ggplot(steel_data5, aes(fill=Load_Type, y=AvgUsage_kWh, x=Day_of_week)) +
  geom_bar(position="dodge", stat="identity")+scale_fill_manual(values = c("#004C99","#009900","#CC0000
  geom_text(aes(label = AvgUsage_kWh_roff), vjust = -0.5, size = 2.5,
            position = position_dodge(0.9))+
  labs(title="Average Energy Consumption for Different Types of Load on various days",x="WeeK Days", y=
```



This stacked bar represents the average of several energy-consuming loads over the course of the week, including "Light load," "Maximum Load," and "Medium Load." The maximum load has been using the most energy for the process, as seen in the graph since they are larger equipment with higher power requirements.

Monday through Friday are essentially the same, but Saturday is only halfway productive because of the half-day work week and Sunday's reduced energy use because of the holiday.

```r
#The Below code is use to Show Mean Energy Consumption by a Industry Over Weekdays and Weekends.
steel_data6<-steel_data%>%
  group_by(Load_Type,WeekStatus)%>%
  summarize(AvgUsage_kWh = mean(Usage_kWh), .groups = 'drop')
steel_data6$AvgUsage_kWh_roff <- round(steel_data6$AvgUsage_kWh, digits = 2)
ggplot(steel_data6, aes(fill=Load_Type, y=AvgUsage_kWh, x=WeekStatus)) +
  geom_bar(position="dodge", stat="identity")+scale_fill_manual(values = c("#00994C","#0080FF","#CC6600
  geom_text(aes(label = AvgUsage_kWh_roff), vjust = -0.4, size = 3.5,
            position = position_dodge(0.9))+
  labs(title="Average Energy Consumption by Industry in Weekdays and Weekends",x="WeekStatus", y= "Avera
```



This stacked bar displays the average energy use for weekdays and weekends for the whole year of 2018.

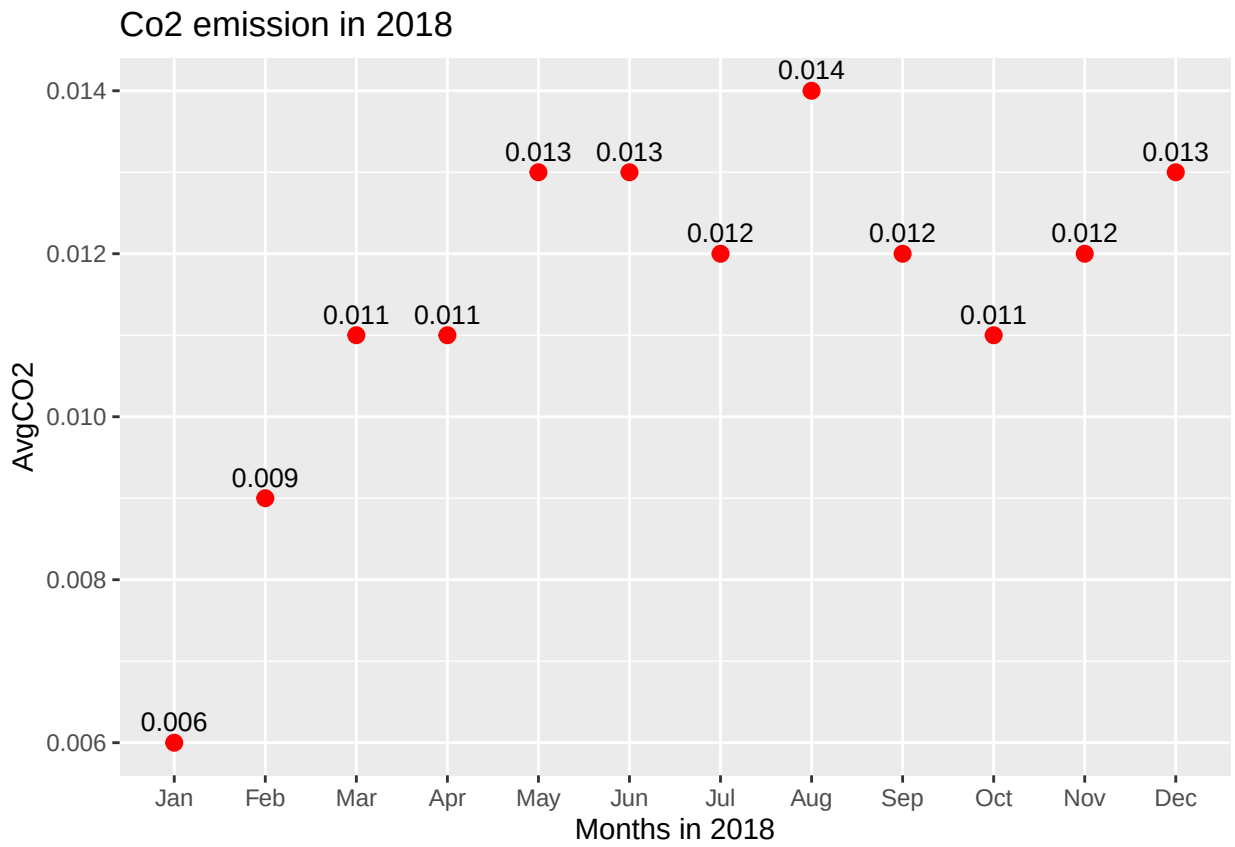
It is clear that weekdays are more productive, which leads to higher power consumption, while weekend energy use is significantly lower than weekday energy use.

5] Mean Co2 emission across each month due to the generation of electrical energy

```r
#The Below code is use to show Months in Year 2018 vs  average Co2 emission
steel_data7 <- steel_data %>%
select(CO2.tCO2., months) %>%
group_by(months)%>%
```

```
summarize(AvgCO2 = mean(CO2.tCO2.))%>%
na.omit

month_xInt7 <- as.integer(steel_data7$months)
steel_data7$months<- month.abb[month_xInt7]
steel_data7$AvgCO2_roff <- round(steel_data7$AvgCO2, digits = 3)

ggplot(steel_data7, aes(x = fct_inorder(months), y = AvgCO2_roff))+
  geom_line(color = "#0099f9", size = 2)+
  geom_point(size = 2.5, color = "red")+
  geom_text(aes(label = AvgCO2_roff), vjust = -0.6, size = 3.5,
            position = position_dodge(0.9))+
  xlab("Months in 2018") + ylab("AvgCO2")+
  ggtitle("Co2 emission in 2018")
```

```
## Warning: Using 'size' aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use 'linewidth' instead.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.

## 'geom_line()': Each group consists of only one observation.
## i Do you need to adjust the group aesthetic?
```

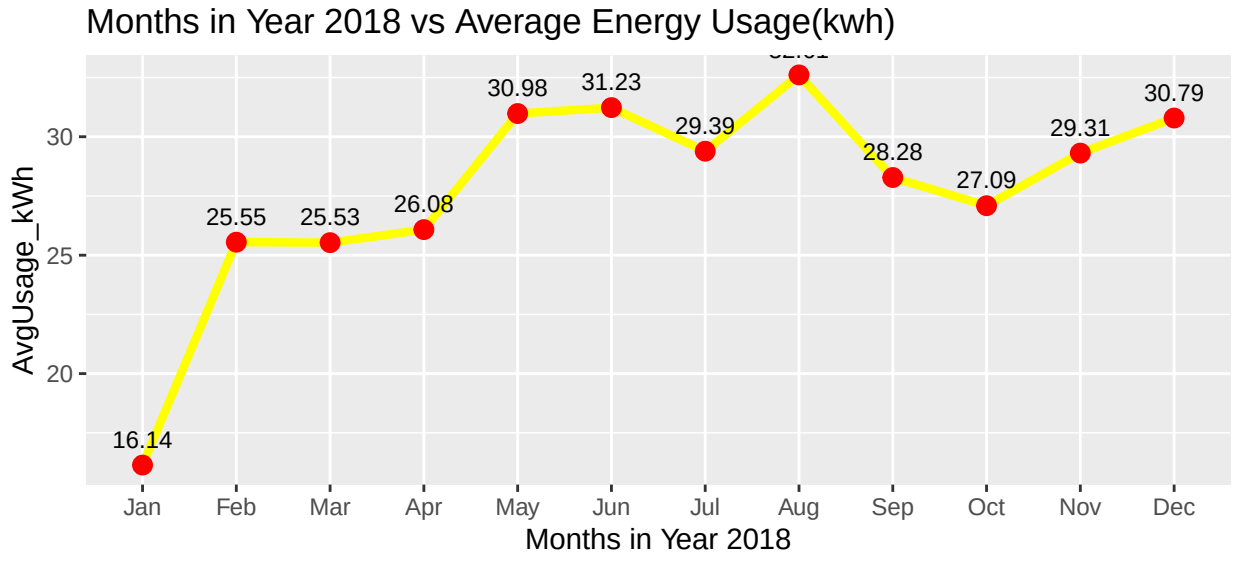


We're attempting to determine the typical carbon dioxide emissions associated to the production of electricity for each month of 2018.

The hot weather in South Korea causes people to consume more power than they do during other seasons, which leads to a huge quantity of electrical energy generation and, as can be seen in the scatter plot above, more CO2 emissions in August.

6] Mean energy usage throughout the year

```
#The Below code is use to show Months in Year 2018 vs Average Energy Usage(kwh)
steel_data8 <- steel_data %>%
select(Usage_kWh, months) %>%
group_by(months)%>%
summarize(AvgUsage_kWh = mean(Usage_kWh))%>%
na.omit

month_xInt8 <- as.integer(steel_data8$months)
steel_data8$months<- month.abb[month_xInt8]
steel_data8$AvgUsage_kWh_roff <- round(steel_data8$AvgUsage_kWh, digits = 2)

ggplot(data=steel_data8, aes(x=fct_inorder(months), y=AvgUsage_kWh_roff, group=1)) +
  geom_line(color = "yellow", size = 1.5)+
  geom_point(color = "red", size = 3)+
  geom_text(aes(label = AvgUsage_kWh_roff), vjust = -1, size = 3,
            position = position_dodge(0.9))+
   xlab("Months in Year 2018") + ylab("AvgUsage_kWh")+
  ggtitle("Months in Year 2018 vs Average Energy Usage(kwh)")
```

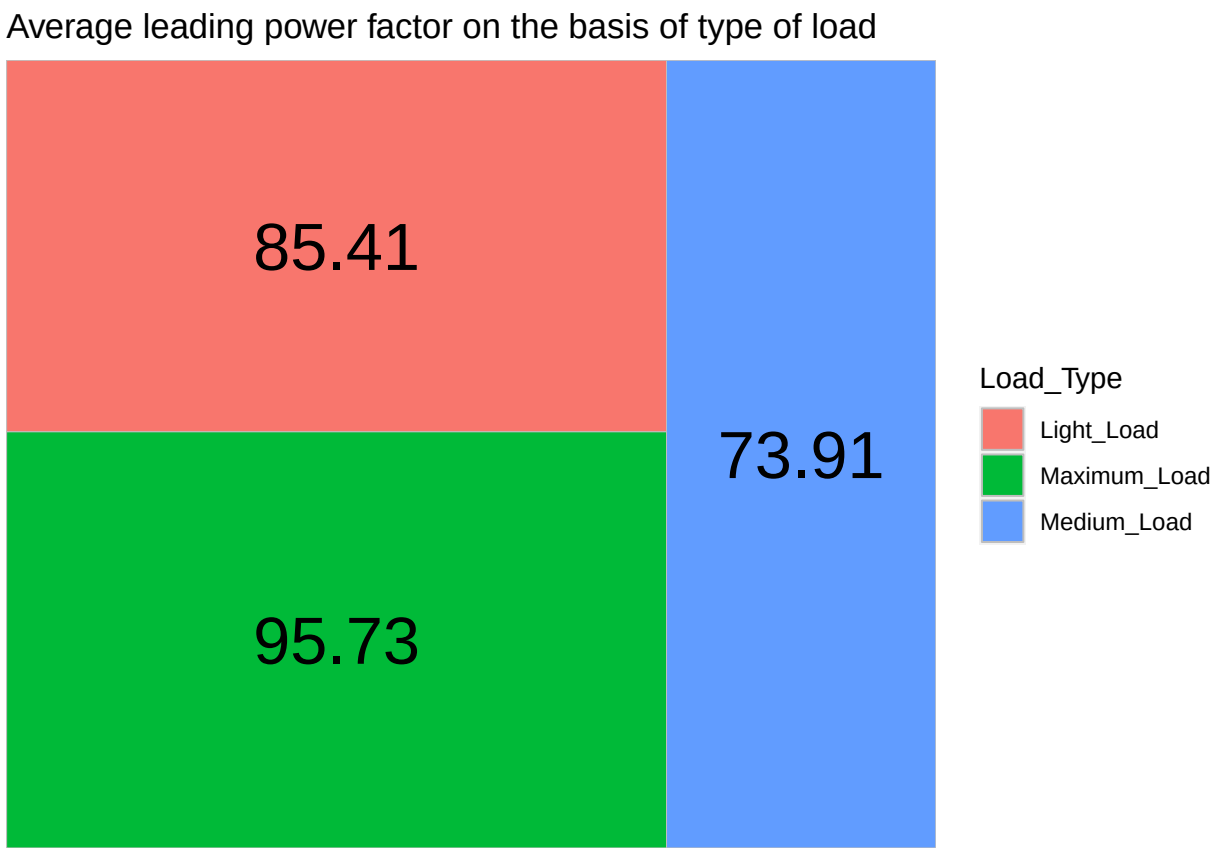


The line graph demonstrates that August uses the most energy of all the months in our effort to determine which month consumed the most energy in 2018.

Due to South Korea's hot weather, customers often use more electricity in August than they do in other months, leading to massive increases in both the supply and consumption of electrical energy.

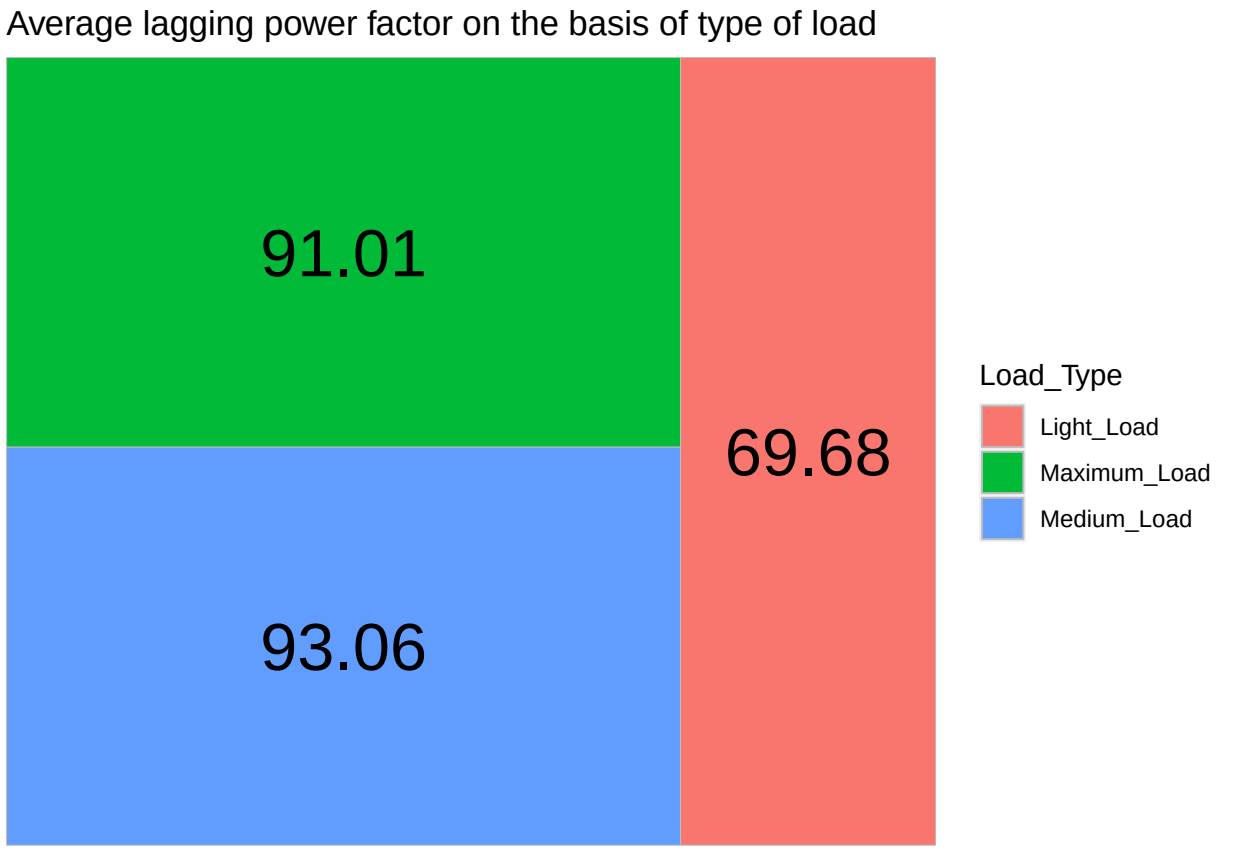
7] Comaprison of Leading power factor and Lagging power factor on the basis of different type of loads.

```
#The Below Code is use to Plot a Tree Map Which tells us Different Types of Load
#According to the Avg  Usage kWh

steel_data9<-steel_data%>%
  group_by(Load_Type)%>%
  summarize(AvgReactive = mean(Leading_Current_Power_Factor))

steel_data9$AvgReactive_roff <- round(steel_data9$AvgReactive, digits = 2)

graph_9 <- ggplot(steel_data9, aes(area = AvgReactive_roff, fill = Load_Type, label = AvgReactive_roff)
  geom_treemap() +
  geom_treemap_text(colour = "black",
                    place = "centre",
                    size = 25)+
labs(title="Average leading power factor on the basis of type of load ")
graph_9
```

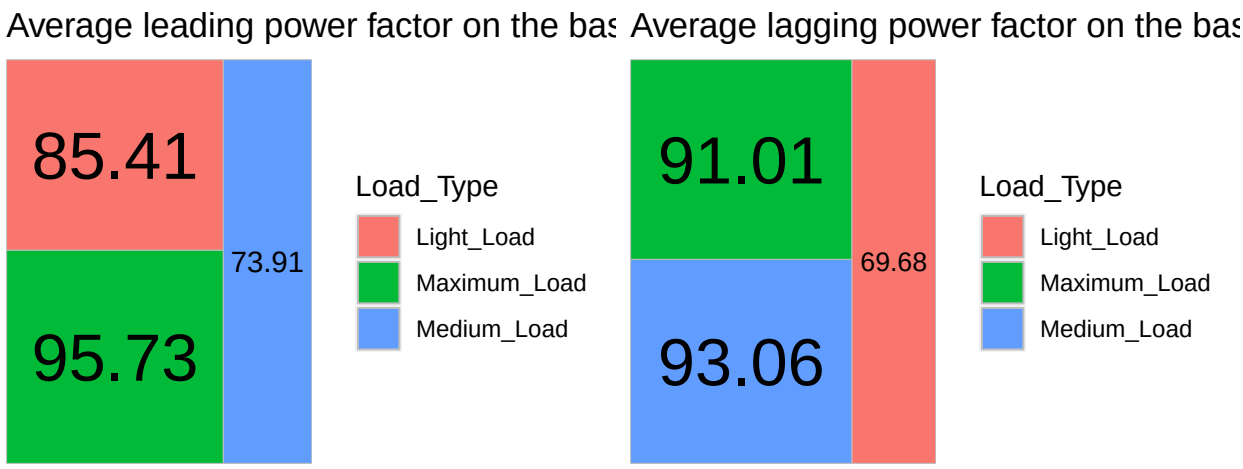



```
steel_data10<-steel_data%>%
  group_by(Load_Type)%>%
  summarize(AvgReactive = mean(Lagging_Current_Power_Factor))
```

```
steel_data10$AvgReactive_roff <- round(steel_data10$AvgReactive, digits = 2)

graph_10 <- ggplot(steel_data10, aes(area = AvgReactive_roff, fill = Load_Type, label = AvgReactive_rof
  geom_treemap() +
  geom_treemap_text(colour = "black",
                    place = "centre",
                    size = 25)+
labs(title="Average lagging power factor on the basis of type of load ")
graph_10
```




```
#The Below Code is Use to Group 2 graph together
grid.arrange(graph_9,graph_10,ncol=2,nrow=2)
```



This tree map shows the average power factor for the leading current and lagging current for several sorts of loads, including “Maximum load,” “Medium load,” and “Light load.” Higher power factor values represent more effective and efficient operation of the equipment.

The tree map now reveals that “Maximum load” has a better power factor than the other loads for the leading power factor, and “Medium load” has a better power factor than the other loads for the lagging power factor.