Springboard - Capstone project

Jorge Moscat

July 2016

Capstone project: Analyzing ENDESA's energy dataset

For this capstone project I will be performing some initial analysis to a smartmeterdataset that Endesa (utilities company) published for a datathon they organized at the beginning of 2016. The dataset includes >7Gb of data from customers' electricity smartreaders installed across their network. In the dataset there are 1-hr interval readings of the electricity consumed by more than 100k customers accross Spain for 2+ years. Nowadays with the roll-out of smartmers that substitute the analog meters, utilities company can potentially capture detailed data about energy usage for each of their customers. This detailed information has never been available to electricity company and the deep analysis of this data open a world of opportunities to better be able to model customer behavior, predict electricity demand to the household level, etc.

The original dataset was published in this website (http://www.endesaenergychallenges.com/datathon/) but since the datathon has officially ended, the download link is not available anymore.

Structure of the ENDESA dataset:

The dataset that ENDESA published in the website above is a pipe-delimited file. Each records correspond to a customer and a day of between March 2013 and October 2015; for each record there are 24 columns that correspond to the electricity consumption readings for each hour of the day.The full dataset contains only subsets of customers for each of the region. Below I am including a quick description of the each of the fields in the dataset and a short explanation of why did or I did not consider each one.

- **DAY**: The date on which the energy usage occurred, in the following format: YYYY/MM/DD
- **H1, H2,., H25**: The time at which the energy usage occurred. There are 25 entries due to the time change when the clocks are set back one hour on the last Sunday of October each year, which leads to a total of 25 hours on this day.
- **ACTIVA_H1, ACTIVA_H2,., ACTIVA_H25**: Active energy consumed per hour as measured in kWh (kilowatt-hour). This is the useful energy that the customers absorb from the grid and transform into work and/or heat at home.
- **CITY**: The municipal district to which the customer belongs. Geographical reference.
- **TARGET_TENENCIA_CUPS**: Probability that the municipal district in question is already equipped with a natural gas distribution network (which does not imply that the customer had contracted natural gas service).
  - Not part of the analysis: This column did not add any relevant information to the specific analyses I performed, therefore I decided to not include it.
- **CLIENT_ID**: The unique, customer reference number which allows for segmentation of usage per customer.
- **CNAE**: (National Classification of Economic Activities) This value indicates whether the customer is domestic (T1) or not (T2).
  - Domestic customer: To reduce the scope and complexity of the analysis I decided to only include domestic clients. Trying to model the behavior of non-domestic clients (i.e. commercial clients) can be far more complex as there are many other factors that impact the electricity consumption of this kind of customers (type of business, size of the company, seasonality, etc.)
- **PRODUCT**: Tariff / electrical product that the customer has contracted; there are up to 120 products.

- **MARKET**: This value indicates whether the customer has a regulated tariff (M1) or a free market tariff (M2).Regulated Tariff: the price of the electricity is regulated periodically by the corresponding authority. Free market: the price of the electricity is freely agreed upon by the provider and the customer.
  - For all the analysis below, I am only considering the M2 customers. For an utilities company, free market tariff customers are more profitable than regulated ones (free market vs. electricity price fixed by government), and therefore more relevant to analyze in detail.

High level remarks about the dataset

Months before I signed-up for Springboards's Intro to Data Science course, I performed some quick analysis on the dataset using Alteryx (GUI-based analytics tool). Below are some high level characteristics of the dataset:

- **Dataset Scope**: It contains 2 years worth of individual customers data with the energy consumption level by hour for each day, collected from smart meters.
- **Total number of records**: 32millon entries (combinations of customer + day + hour of the day) -> 7.5Gbs
- **Date range**: There are records from 12-09-2013 till 14-10-2015 (most of the volume is concentrate between 2014-2015)
- **Geographical locations**: 808 municipalities in Spain
- **Customers**: There are >100k unique customer IDs, which just to give a sense of the dimensions, this is only <1% of total customer base of Endesa in Spain.
- **Additional Attributes**: The data set contains some really interesting additional attributes: Product contracted, Probability of the municipality is already equipped with a natural gas distribution network, tariff type (regulated vs non-regulated), etc.

Parsing the 8Gb dataset

The original dataset was a "pipe"-delimited dataset of more than 7.5GB of size. Using the full dataset on R was not a viable option since R loads up all the data into memory. Given the RAM limitation of my computer, using R directly up was not feasible and therefore, I had find a workaround to extract a relevant subset of the full dataset to run some analysis. To do so, I used command-line tools to parse through the large dataset and take a smaller sample to analyze on R.

The **command-line** tools I used were:

- **SED**: I used SED toolkit to convert the full 7.5GB dataset from a pipe-delimited file to a CSV
- **CSVKIT**: Once I had my dataset in CSV format I used CSVKIT to select just a small subset based on the criteria described below.

Leveraging these command-lines tools I created a subset of the original dataset to be able to start doing some preliminary analysis. This subset was the result of filtering according to the following criteria:

- CITY: I included 10 municipalities (including Madrid)
- DAY: Only selected data from the months of January and July of 2015
- CNAE: Only included data from domestic clients (i.e. no businesses)
- Market: Only data corresponding to "free market" clients (i.e. electricity price determine by demand and supply rather than fixed by the regulator)

Exploratory analysis of the dataset

These are libraries I am using throughout the capstone

```
library("tseries")
library("dplyr")
library("tidyr")
library("ggplot2")
library("cluster")
library("NbClust")
```

As mentioned above, the dataset I am intially using for my exploratory analysis includes electricity readings for:

- **Dates**: January 2015 and July 2015
- **Locations**: 11 cities in Spain, including Madrid (the capital)
- **Clients**: 7757
- **Electricity readings**: 1-hr interval readings, therefore for each customer per day there are 24 readings (labeled H1 through H24)

Reading the file:

```
energy_data = read.csv("capstone_dataset_large.csv", sep = ";", header = TRUE)
str(energy_data)
```

```
## 'data.frame':    423779 obs. of  34 variables:
##  $ ACTIVA_H1           : int  157 0 225 213 0 120 192 1094 0 161 ...
##  $ ACTIVA_H10          : int  58 0 83 256 0 227 171 208 0 235 ...
##  $ ACTIVA_H11          : int  141 0 87 284 0 110 993 212 0 185 ...
##  $ ACTIVA_H12          : int  35 0 89 295 0 68 445 195 0 211 ...
##  $ ACTIVA_H13          : int  118 0 85 308 0 86 1340 209 0 236 ...
##  $ ACTIVA_H14          : int  99 0 87 320 0 79 316 183 0 855 ...
##  $ ACTIVA_H15          : int  60 0 87 310 0 111 399 210 0 920 ...
##  $ ACTIVA_H16          : int  158 0 84 291 0 85 751 242 0 702 ...
##  $ ACTIVA_H17          : int  27 0 115 287 0 71 1026 644 0 825 ...
##  $ ACTIVA_H18          : int  156 0 130 297 0 125 568 849 0 567 ...
##  $ ACTIVA_H19          : int  69 0 109 306 0 110 172 836 0 424 ...
##  $ ACTIVA_H2           : int  40 0 181 176 0 119 178 756 0 174 ...
##  $ ACTIVA_H20          : int  114 0 690 301 0 133 178 737 0 835 ...
##  $ ACTIVA_H21          : int  109 0 315 293 0 71 176 821 0 815 ...
##  $ ACTIVA_H22          : int  80 0 367 296 0 129 170 650 0 890 ...
##  $ ACTIVA_H23          : int  143 0 324 293 0 111 162 893 0 834 ...
##  $ ACTIVA_H24          : int  46 0 265 257 0 151 175 608 0 484 ...
##  $ ACTIVA_H25          : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ ACTIVA_H3           : int  121 0 101 156 0 128 138 175 0 225 ...
##  $ ACTIVA_H4           : int  100 0 101 147 0 116 175 179 0 173 ...
##  $ ACTIVA_H5           : int  63 0 91 143 0 127 178 160 0 176 ...
##  $ ACTIVA_H6           : int  150 0 92 143 0 115 156 210 0 188 ...
##  $ ACTIVA_H7           : int  28 0 94 155 0 136 167 198 0 154 ...
##  $ ACTIVA_H8           : int  123 0 140 181 0 117 169 276 0 165 ...
##  $ ACTIVA_H9           : int  91 0 135 214 0 130 148 232 0 228 ...
##  $ CNAE                : Factor w/ 1 level "T1": 1 1 1 1 1 1 1 1 1 1 ...
```

```
##  $ CITY                : Factor w/ 11 levels "ALCOBENDAS","ALCORCON",..: 6 6 4 6 6 6 6 6 6 6 ...
##  $ DAY                 : int  20150706 20150709 20150705 20150731 20150729 20150727 20150728 2015070
##  $ CLIENT_ID           : int  85422 81227 74418 98939 99051 81949 69862 79630 81227 78906 ...
##  $ MARKET              : Factor w/ 1 level "M2": 1 1 1 1 1 1 1 1 1 1 ...
##  $ PRODUCT             : Factor w/ 35 levels "P11","P110","P111",..: 21 5 5 14 14 5 14 24 5 24 ...
##  $ TARGET_TENENCIA_CUPS: Factor w/ 15 levels "","0.000000",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ MONTH               : int  7 7 7 7 7 7 7 7 7 7 ...
##  $ ACTIVA_HT           : int  2286 0 4077 5922 0 2775 8543 10777 0 10662 ...
```

Data Preparation

From the original dataset, the **day** field is a integer, so in the following code I am converting it to a proper R date format and also creating a column to specify the day of the week (DOW). The DOW is stored as an integer (0=sunday, 6=saturday) to make plotting a bit more intuitive later on.

```
energy_data$DAY <- sapply(energy_data$DAY, toString)
energy_data$DAY <- as.Date(energy_data$DAY, "%Y%m%d")
energy_data$DAY <- as.Date(energy_data$DAY, "%Y%m%d")
#Create additional column that indicates the day of the week
energy_data <- energy_data %>%
  mutate(DOW = toupper(substr(weekdays(as.Date(DAY)), 1,2)))
```

Based on some exploratory and priliminary analysis, I have realized that I need to create two versions of my energy_datasets reshaping the layout for a more efficient use of memory and better plotting speed.

- daily_energy_dataset: In this dataset, I got rid of the hourly readings and created two columns to summarize the energy consumption for each particular day
  - AVG_CONS_PER_HR: Average hourly energy consumption per day and client, which is the total energy consumed in the day over 24hrs
  - TOTAL_DAY_CONS: Total energy consumed on a given day for a particular client

```
daily_energy_dataset <- energy_data %>%
  select(-MARKET, -TARGET_TENENCIA_CUPS,-CNAE) %>%
  mutate(AVG_CONS_PER_HR= ACTIVA_HT/24) %>%
  mutate(TOTAL_DAY_CONS = ACTIVA_HT) %>%
  select(-starts_with("ACTIVA_H"))
```

- hourly_energy_dataset: This dataset contains all the hourly readings for each combination of day and client. I have used the "gather" function to condese the hourly reading columns into rows

For both datasets, I am getting rid of MARKET, TARGET_TENENCIA_CUPS and CNAE columns as I am not going to use them for my analysis. These variable contain very technical information about the customer that I decided to leave out of the analysis as they do not add any additional information to the use cases I am going to implement later on.

```
hourly_energy_dataset <- energy_data %>%
  select(-MARKET, -TARGET_TENENCIA_CUPS,-CNAE, -ACTIVA_HT) %>%
  gather(key=hour_of_day, value = active_consumption, ACTIVA_H1, ACTIVA_H2,ACTIVA_H3,ACTIVA_H4,ACTIVA_H5
```

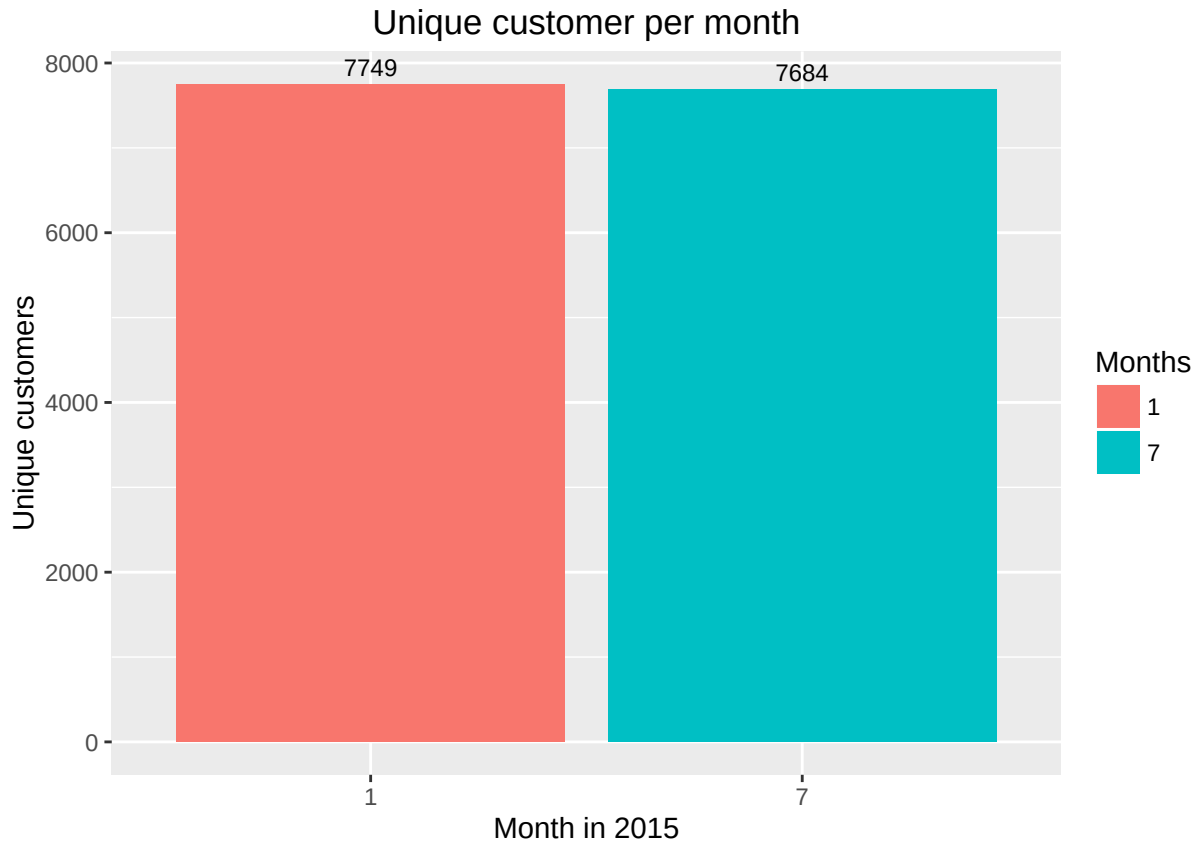
Initial data exploration

1. First, check how many unique customers are included in our dataset:

```
##   n_distinct(CLIENT_ID)
## 1                  7757
```

2. Checking the number of **unique customers per month** shows a slight difference with a bit less customers in July vs January

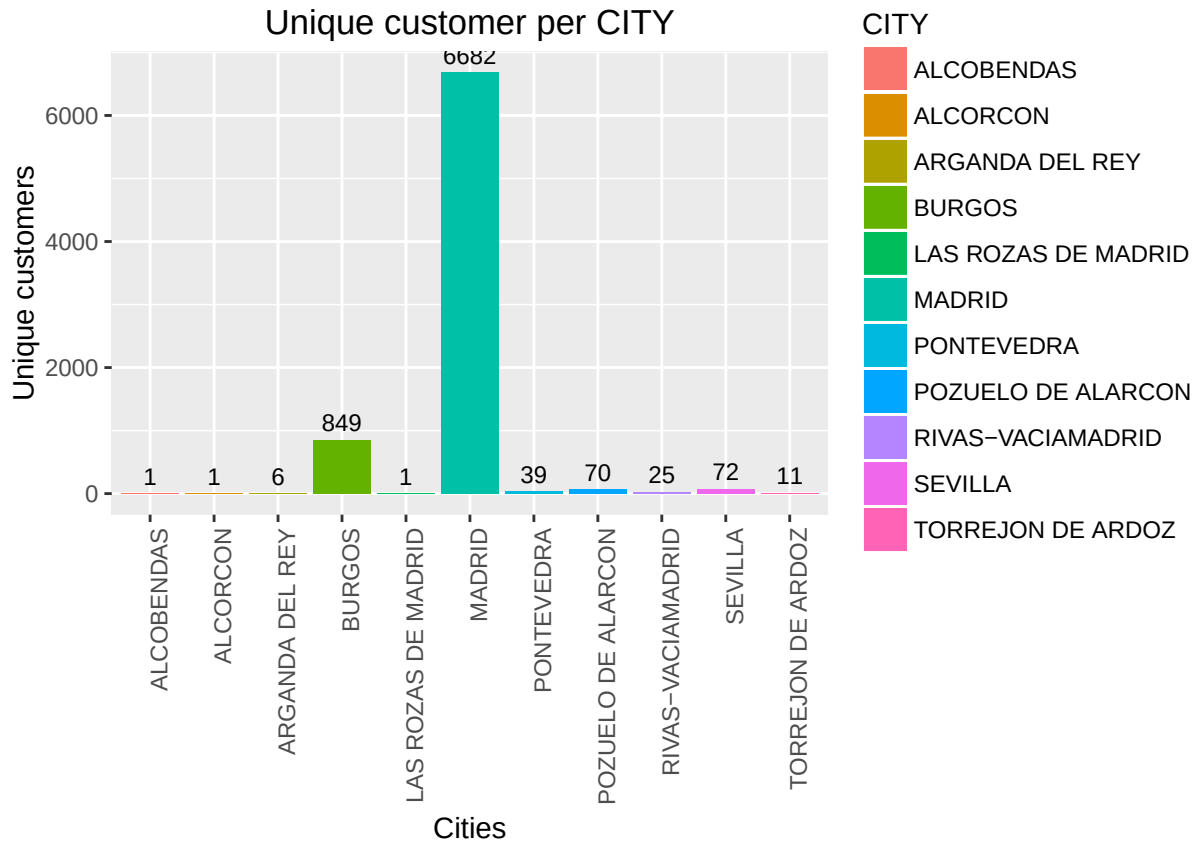
```
energy_data %>%
  group_by(MONTH) %>%
  summarise(unique_client_count= n_distinct(CLIENT_ID)) %>%
  ggplot(data=., aes(x=factor(MONTH), y = unique_client_count, fill = factor(MONTH))) + geom_bar(stat="
  geom_text(aes(label = unique_client_count), size = 3, position = position_dodge(0.6),vjust = -0.5) +
  labs(title = "Unique customer per month", x = "Month in 2015", y = "Unique customers") +
  scale_fill_discrete(guide=guide_legend(title = "Months"))
```



3. Checking the number of **unique customers per region** included in the sample dataset. In this dataset, Madrid has significantly higher number of customers.

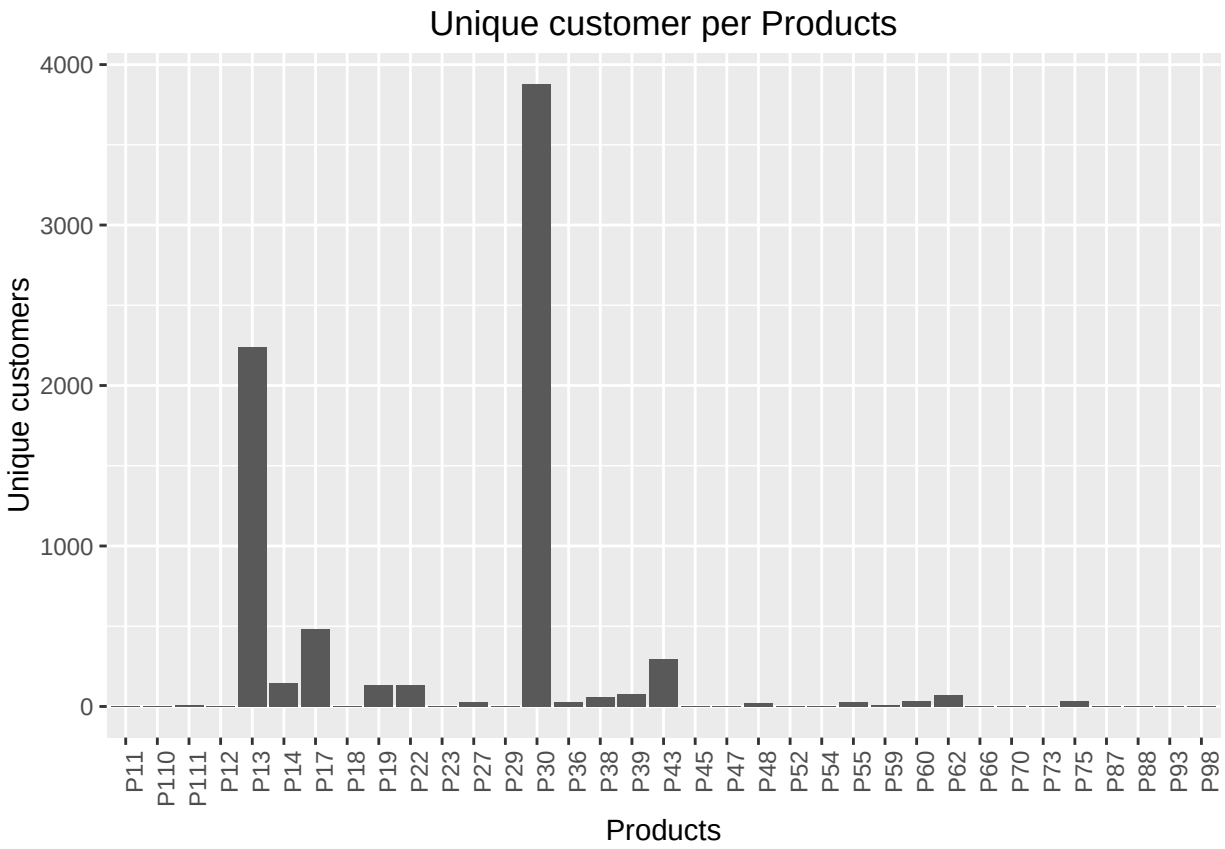
```
energy_data %>%
  group_by(CITY) %>%
  summarise(unique_client_count = n_distinct(CLIENT_ID)) %>%
```

```
  ggplot(data = ., aes(x = CITY, y = unique_client_count, fill = CITY)) + geom_bar(stat = "identity") +
  geom_text(aes(label = unique_client_count), size = 3, position = position_dodge(0.6),vjust = -0.5) +
  labs(title = "Unique customer per CITY", x = "Cities", y = "Unique customers") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



4. **Customer by product type**: Understanding how many customer do we have by product type.
    - Looking at the plot, it is clear that most customer have contracted P13 and P30 products

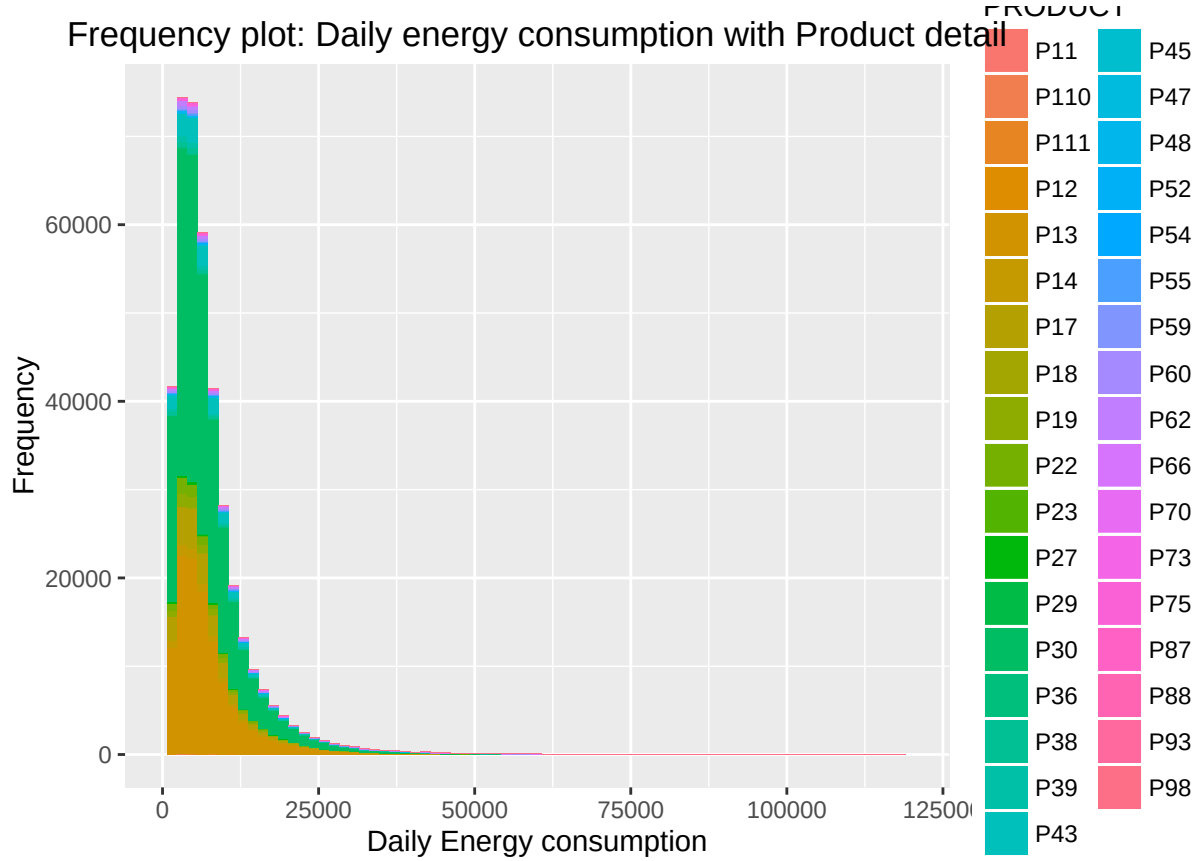
```
energy_data %>%
  group_by(PRODUCT) %>%
  summarise(unique_client_count = n_distinct(CLIENT_ID)) %>%
  ggplot(data=., aes(x=PRODUCT, y = unique_client_count)) + geom_bar(stat="identity") +
  labs(title = "Unique customer per Products", x = "Products", y = "Unique customers") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



5. **Energy consumed distribution**: Another way to see the distribution of product types and the energy consumed, we can create a histogram that clearly shows which two products are the most popular and can help us get an understanding of distribution of the total energy consumed per day by client

```
#Using Freedman-Diaconis rule to determine the optimal bin size for the histogram
bw <- diff(range(daily_energy_dataset$TOTAL_DAY_CONS)) / (2 * IQR(daily_energy_dataset$TOTAL_DAY_CONS) /
daily_energy_dataset %>%
ggplot(data = ., aes(x = TOTAL_DAY_CONS, fill = PRODUCT)) +
  geom_histogram(binwidth = bw/4) +
  xlim(c(0,120000)) +
  labs(title = "Frequency plot: Daily energy consumption with Product detail", x = "Daily Energy consump
```

```
## Warning: Removed 12 rows containing non-finite values (stat_bin).
```



6. **Hourly energy profiles**: Since P13, P17 and P30 are the most commun products, lets check how customers's electricity average usage looks like by hour of the day for these three products. The goal of this visualization is twofold:

   a- Understand the amount of electricity that customers use in each of these three products (products could be tailored to specific levels of electricity usage)

   b- Understand if the hourly behavior (i.e. line pattern) is different as some of the products might be more beneficial for people that consume energy at night vs during the day. To do so, lets calculate the average total consumption by day hour of the day, just for the month of January.

In addition, lets also check if there are any clear difference in the average hourly electricity consumption per city.

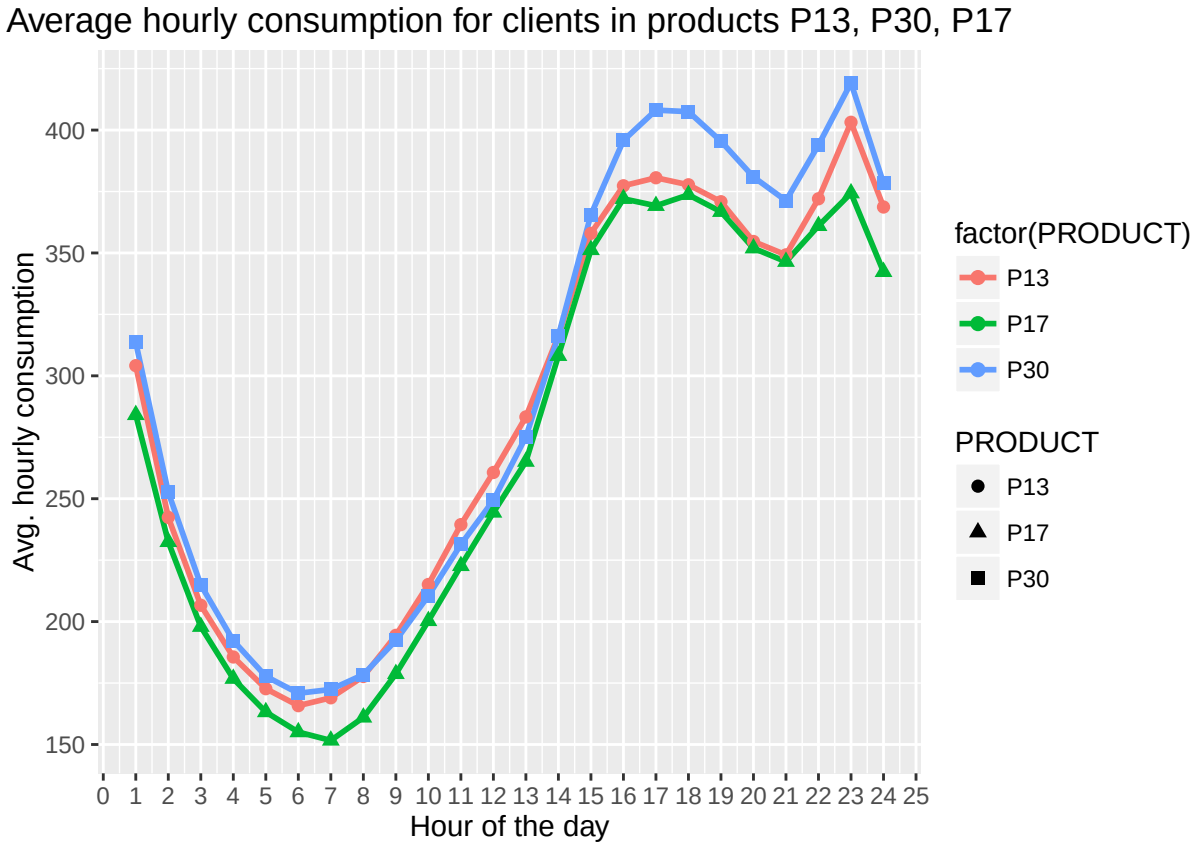
```
#Average hourly behavior by product
hourly_usage_product <- hourly_energy_dataset %>%
  filter(MONTH == 7, PRODUCT %in% c("P13", "P30", "P17"), hour_of_day != "ACTIVA_H25") %>%
  group_by(PRODUCT, hour_of_day) %>%
  summarise(avg_usage = mean(active_consumption))

#Average hourly behavior by city
hourly_usage_city <- hourly_energy_dataset %>%
  filter(MONTH == 7, PRODUCT %in% c("P13", "P30", "P17"), hour_of_day != "ACTIVA_H25") %>%
  group_by(CITY, hour_of_day) %>%
  summarise(avg_usage = mean(active_consumption))
```

To ensure that the x-axis of the following plot shows the hours of the day in the correct order, I am modifying the format of the "hour of the day" column getting rid of "ACTIVA_H" and just leaving the hour number.

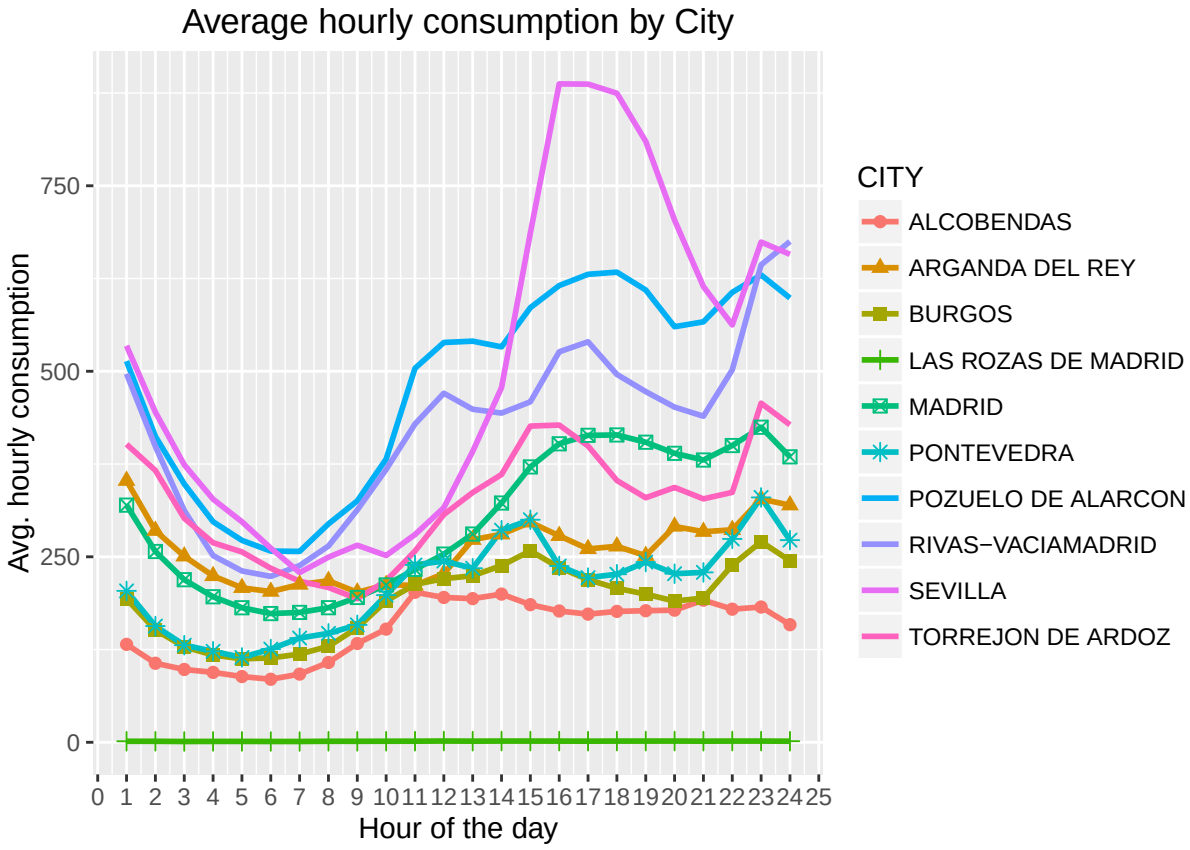
```
hourly_usage_product$hour_of_day <- gsub("ACTIVA_H", "", hourly_usage_product$hour_of_day)
hourly_usage_city$hour_of_day <- gsub("ACTIVA_H", "", hourly_usage_city$hour_of_day)
```

Now, plotting the hourly behavior by product and city:

```
#Plot by product
ggplot(data = hourly_usage_product, aes(x = as.integer(hour_of_day), y = avg_usage, group = PRODUCT, sha
  geom_line(size=1) + geom_point(size=2) + xlab("Hour of the day") + ylab("Avg. hourly consumption") +
  ggtitle("Average hourly consumption for clients in products P13, P30, P17") +
  scale_fill_discrete(guide=guide_legend(title = "Product")) +
  scale_x_continuous(breaks=seq(0, 25, 1))
```




```
#Plot by city
ggplot(data = hourly_usage_city, aes(x = as.integer(hour_of_day), y = avg_usage, group = CITY, shape = C
  geom_line(size=1) +  geom_point(size=2) + xlab("Hour of the day") + ylab("Avg. hourly consumption") +
  ggtitle("Average hourly consumption by City") +
  scale_x_continuous(breaks=seq(0, 25, 1))
```



Conclusions from hourly behavior exploration

Product type plot:

- From the first plot (by product) we see that on average the hourly behviour of clients in products P13, P17 and P30 is very similar in terms of the amount of energy consumed and the pattern of consumption per hour.
  - With one exeception: clients that have contracted P30 seems to have a higher rate of electricity usage in the afternoon portion of the day (from 16:00 till midnight)
- On average, clients in P30 consume more energy than P13 and P17. Clients in P13 spend slightly more energy than those in P17

City plot:

- In the plot there is clear outlier with a large amount of energy consumption: **Sevilla**. For this region, as we saw before, we have data from 72 customers. Therefore, to better understand if the average usage of these 72 customers can be extrapolated to the whole region, we would need to better understand what kind of customers they are (i.e. size of the household, etc.) and, if available, include more customers in the analysis
- One of the hypothesis that could explain the reason by energy consumption is much higher for these clients in Seville is the fact this city is probably one of the hottest places in Spain, specially in the month on July. Between 16:00-18:00 is by far the hottest hours of the day and when people in households will increase the AC usage

- The second region with higher energy consumption is **Pozuelo de Alarcon** (70 customers in dataset). This is most certainly due to the socio-economic level of this small region in the state of Madrid. Pozuelo de Alarcon is one of the wealthiest municipalities in Spain and most of the households in this regions are large houses with very few flats.
- For Madrid municipality (>6.5k customers in dataset), we see that the peak of energy consumption on average for a given day is at 23.00, followed by 17.00.

Deep-dive in Madrid: Analyzing the energy consumption of customers in Madrid

After some preliminary analysis in the whole dataset, we are going to deep dive into better understanding the behavior of consumers in Madrid. In this section we will do the following

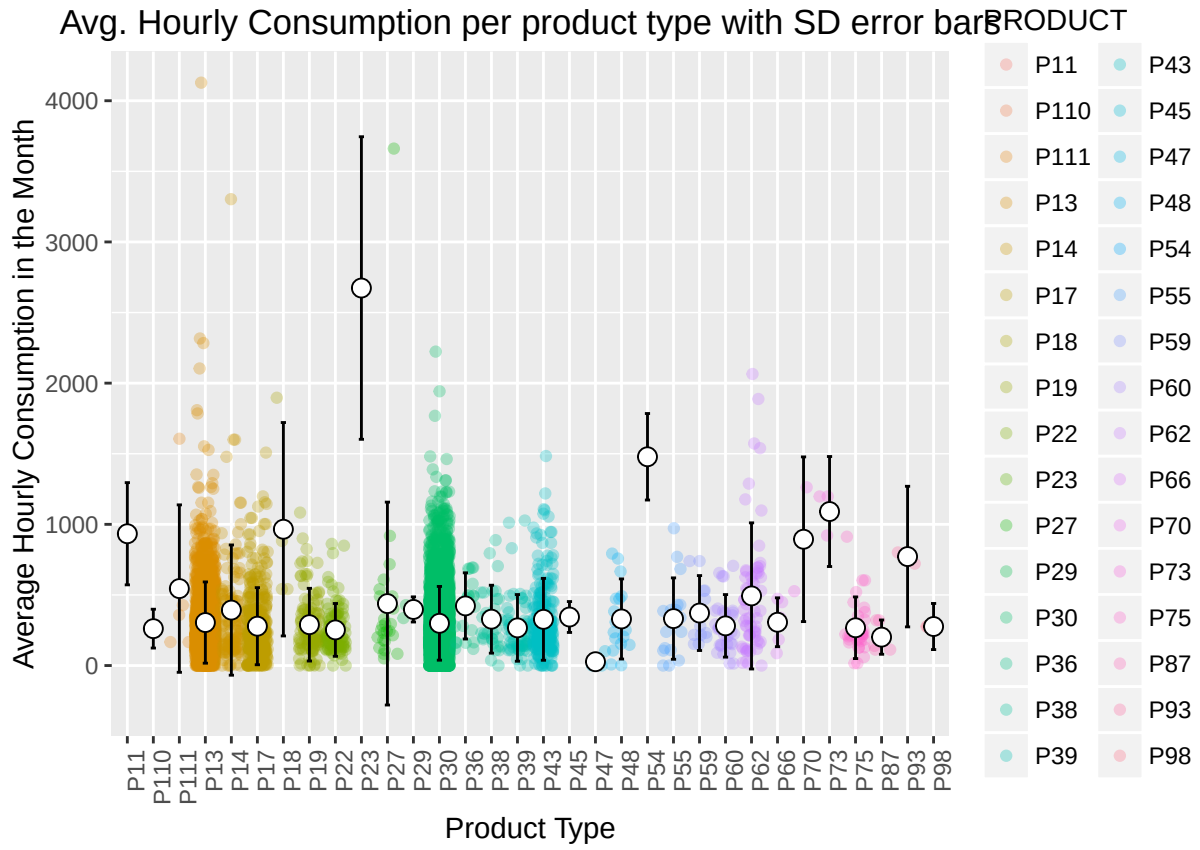
- **1. Data preparation and exploration (Product Type)**: Given that there are more than 30 different energy products in the dataset, we are going to try to determine in which product type there are more customer and which product group show the lowest variability (based on std deviation)
- **2. Impact of AC** Determine if there is clear difference between the energy consumption in summer (July) vs. winter (January) due to the usage of AC in Madrid during July
- **3. Correlation of weather data with energy consumption**: We are going to explore whether the weather conditions (temp and humidity) have a correlation with the energy consumed by customers in the Madrid region. For this we are going to explore data for the months of April, May and June. If we can prove that AC usage impacts the total electricity bill, then it only makes sense that the temperature can "predict" the level of energy consumed by a given customer (higher temps, more AC usage therefore higher energy consumption)
- **4. Weekly behavior clustering**: Using k-means clustering we are going to do some unsupervised analysis to try to cluster together customers with similar weekly energy consumption.
- **5. Customer away from home**: By analyzing the behavior of customers in Madrid throughout three months, we are going to see if we can identify when customers are not home in a given day.

Deep-dive Madrid: 1.Data preparation and exploration (Product Type)

In the previous section we saw that most of the customer have contracted products P13, P17 or P30. Lets quickly check the dispersion of the energy consumptions across products to select for our subsequent analysis only customers in products with relatively low variability

```
temp_data_jitter <-
  daily_energy_dataset %>% filter(CITY == "MADRID", MONTH == "7") %>%
  group_by(PRODUCT, CLIENT_ID) %>%
  summarise(av_hourly_consumption = mean(AVG_CONS_PER_HR))

daily_energy_dataset %>% filter(CITY == "MADRID", MONTH == "7") %>%
  group_by(PRODUCT) %>%
  summarise(av_hourly_consumption = mean(AVG_CONS_PER_HR), std = sd(AVG_CONS_PER_HR), num_clients = n_di
  ggplot(data=., aes(x = PRODUCT, y = av_hourly_consumption)) +
  geom_jitter(data = temp_data_jitter, mapping=aes(x = PRODUCT, y = av_hourly_consumption, colour = PROD
  geom_errorbar( aes(ymin=av_hourly_consumption-std, ymax=av_hourly_consumption+std),
                 width = .2,                    # Width of the error bars
                 position = position_dodge(.9)) +
  geom_point(shape = 21, size = 3, fill = "white") +
  xlab("Product Type") +
```

```
  ylab("Average Hourly Consumption in the Month") +
  ggtitle("Avg. Hourly Consumption per product type with SD error bars") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



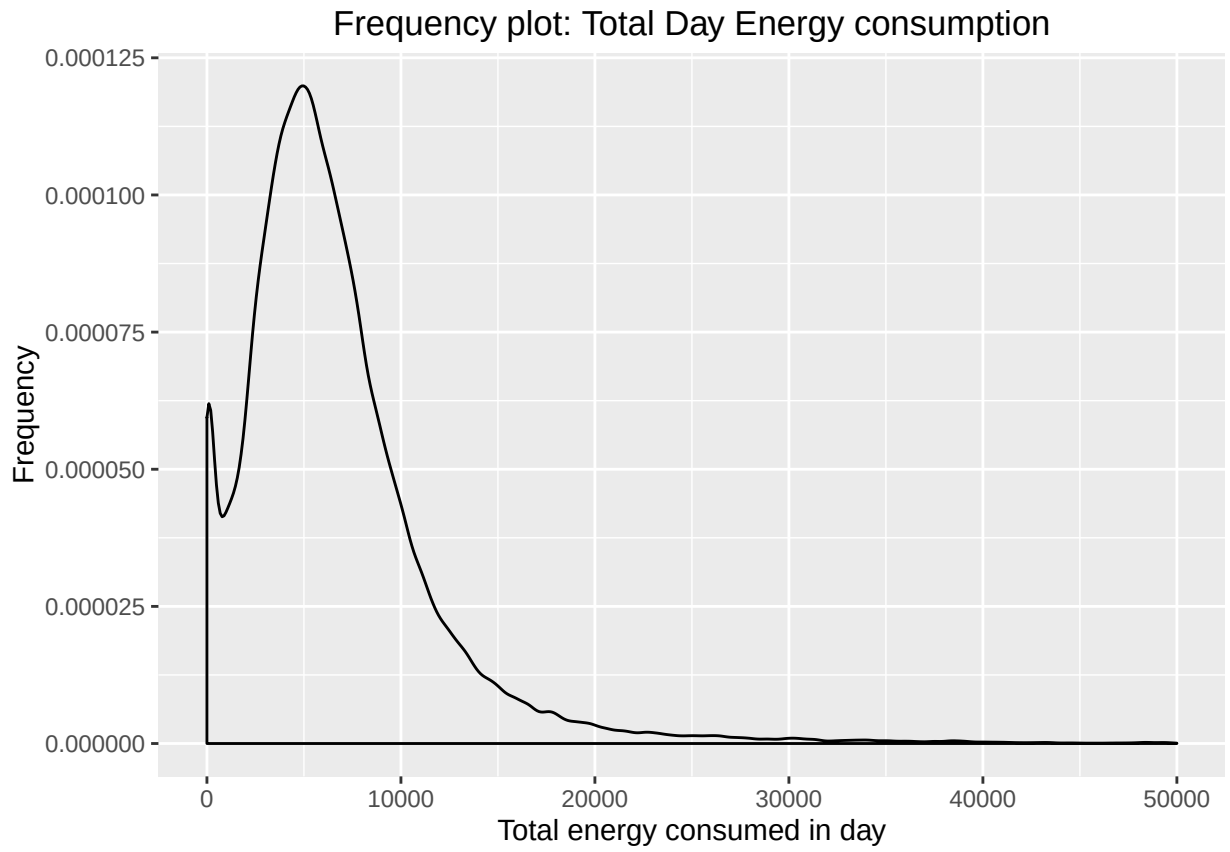
From the plot above, we see that P13, P17 and P30 have most of the customers and also have the **lowest standard deviation**, therefore relatively low variability.

Deep-dive Madrid: 2. Impact of AC

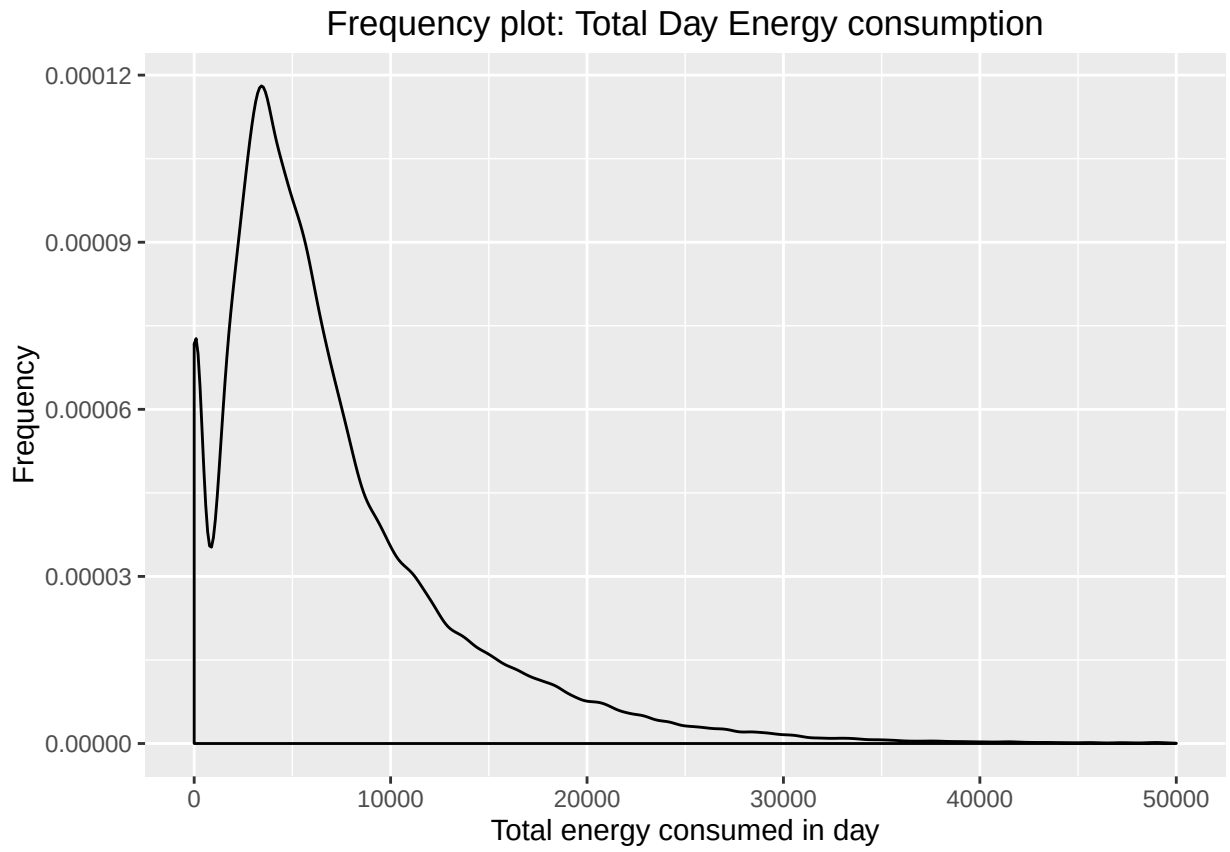
To determine if AC usage has significant impact in the total energy consumed of clients in Madrid, we are going to perform a t-test to see if the average energy consumption in the month of January (coldest month in Madrid) is significantly lower than in July (hottest month, with high AC usage). If it is, we could safely assume that higher energy consumption in July is most probably driven by higher AC usage due to very high temperatures.

Before using a t-test, first we need to determine if the January and July samples follow a normal distribution

```
january <- daily_energy_dataset %>%
  filter(CITY == "MADRID", MONTH == "1", PRODUCT %in% c("P13", "P30", "P17")) %>%
  select(TOTAL_DAY_CONS)
ggplot(january, aes(x=TOTAL_DAY_CONS)) + geom_density(alpha=.3) + xlim(c(0,50000)) +
  labs(title = "Frequency plot: Total Day Energy consumption", x = "Total energy consumed in day", y =
```




```
july <- daily_energy_dataset %>%
  filter(CITY == "MADRID", MONTH == "7", PRODUCT %in% c("P13", "P30", "P17")) %>%
  select(TOTAL_DAY_CONS)
ggplot(july, aes(x=TOTAL_DAY_CONS)) + geom_density(alpha=.3) + xlim(c(0,50000)) +
  labs(title = "Frequency plot: Total Day Energy consumption", x = "Total energy consumed in day", y =
```

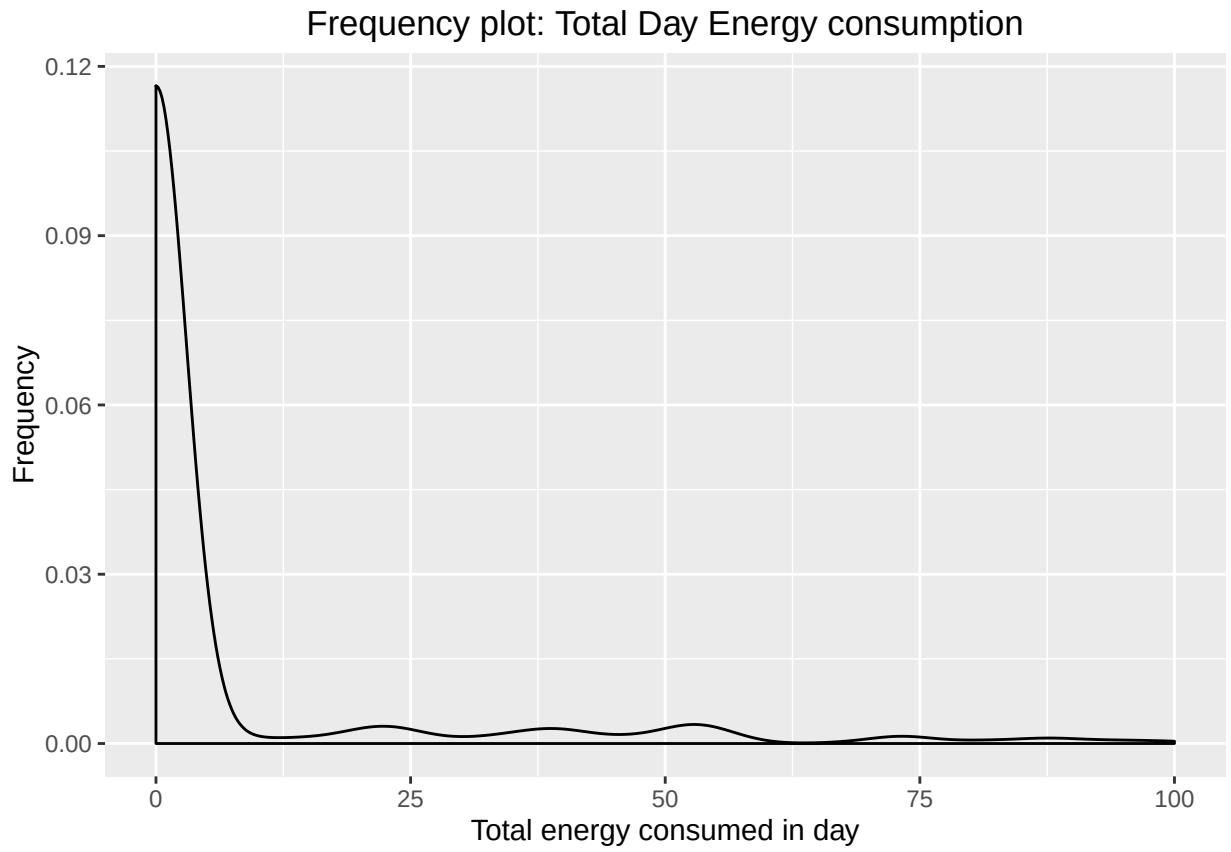


From the plots above we can eye-ball that neither the January nor the July dataset seem to follow a normal distribution. Specially relevant is the fact that there seems to be a lot of samples (days in the month) with daily energy consumption very close to 0.

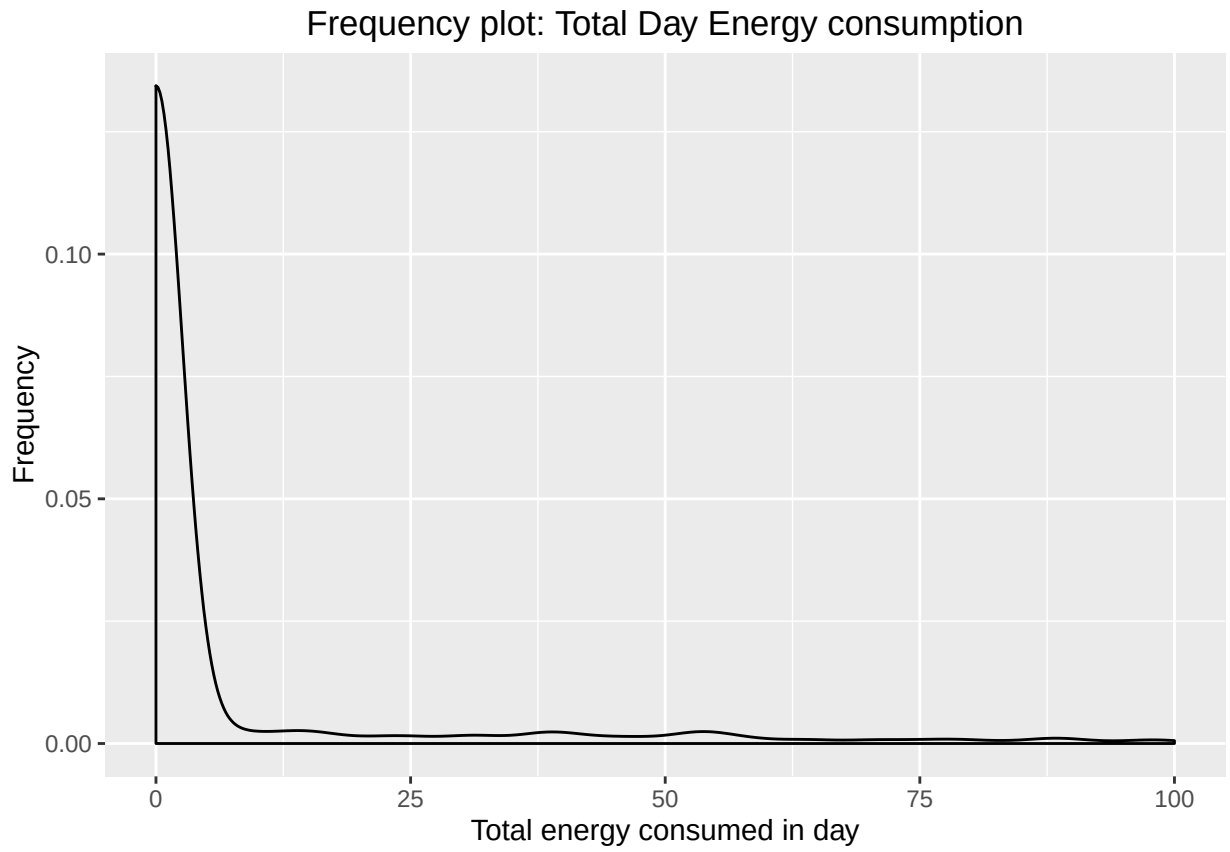
- We could assume that days with energy consumption close to 0 are empty households or specific days when clients were not at home. Therefore, for our analysis we should get rid of these days and just focus our comparision in the days when there are in fact clients using energy at home.

Let's zooming into the density plots above to see what is a reasonable cut-off to get rid of days with energy consumption close to 0:

```
#January density plot
ggplot(january, aes(x=TOTAL_DAY_CONS)) + geom_density(alpha=.3) + xlim(c(0,100)) +
  labs(title = "Frequency plot: Total Day Energy consumption", x = "Total energy consumed in day", y =
```




```
#July density plot
ggplot(july, aes(x=TOTAL_DAY_CONS)) + geom_density(alpha=.3) + xlim(c(0,100)) +
  labs(title = "Frequency plot: Total Day Energy consumption", x = "Total energy consumed in day", y =
```

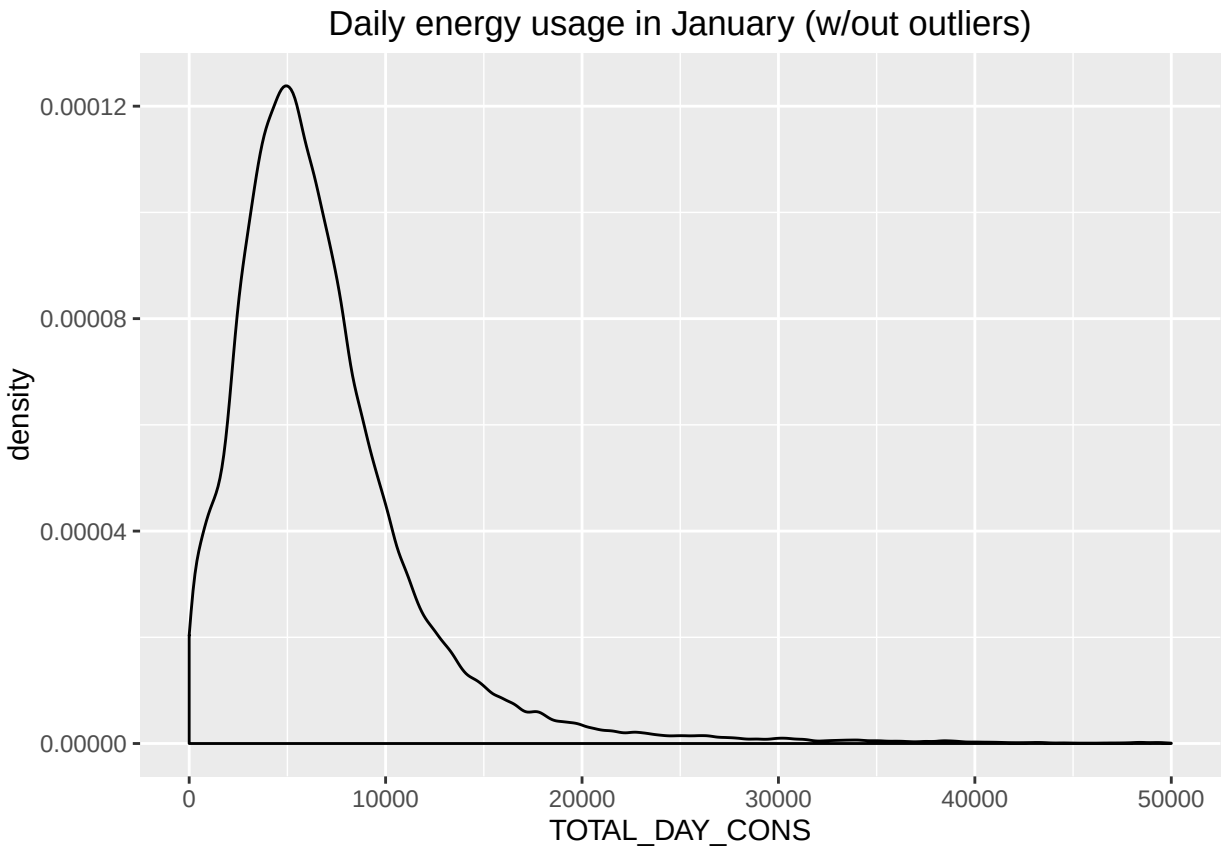


Based on the plots above, lets remove all instances with energy daily energy consumption lower than 20.

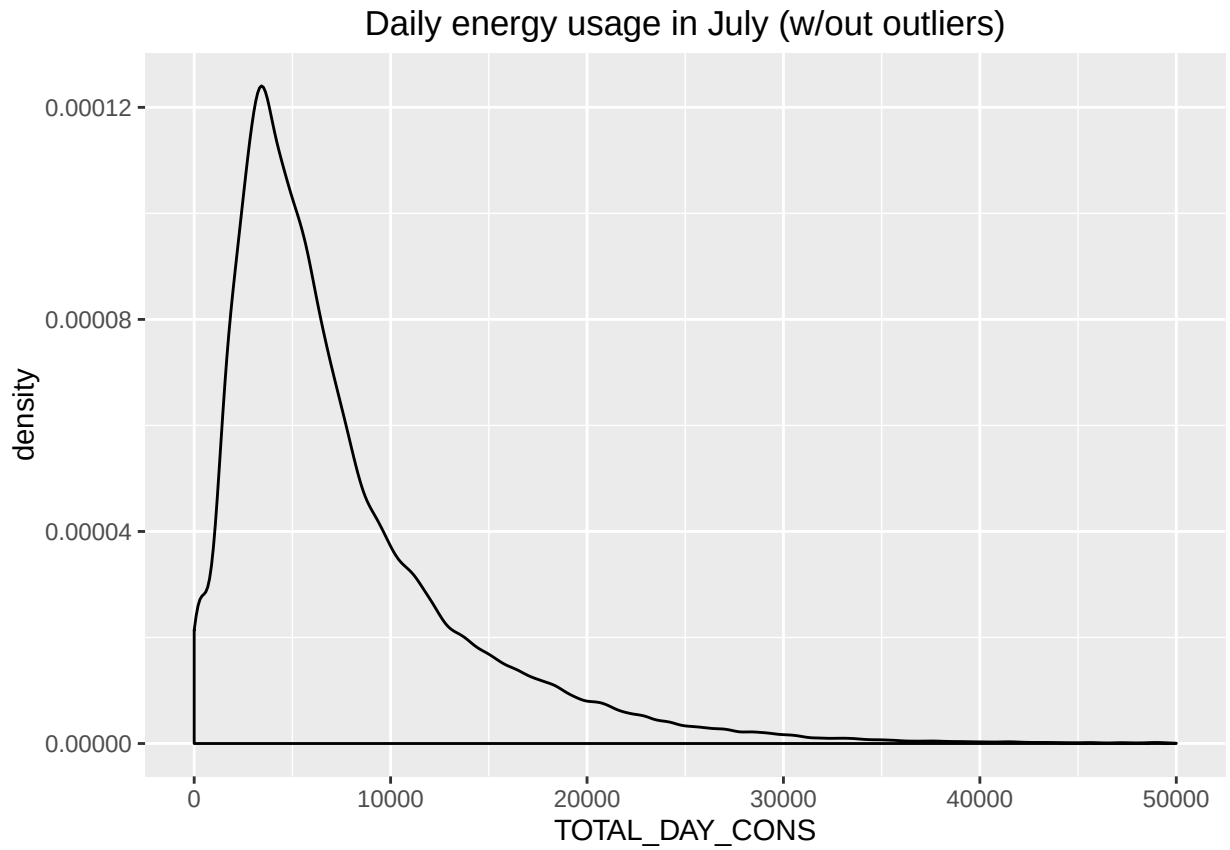
```
january_cleaned <- daily_energy_dataset %>%
  filter(CITY == "MADRID", MONTH == "1", PRODUCT %in% c("P13", "P30", "P17")) %>%
  filter(TOTAL_DAY_CONS >= 20) %>%
  select(TOTAL_DAY_CONS)

july_cleaned <- daily_energy_dataset %>%
  filter(CITY == "MADRID", MONTH == "7", PRODUCT %in% c("P13", "P30", "P17")) %>%
  filter(TOTAL_DAY_CONS >= 20) %>%
  select(TOTAL_DAY_CONS)
```

If we draw new density plots after cleaning out those outliers with usual energy consumption, our distributions look closer to a normal distribution:

```
ggplot(january_cleaned, aes(x=TOTAL_DAY_CONS)) + geom_density(alpha=.3) + xlim(c(0,50000)) +
  ggtitle("Daily energy usage in January (w/out outliers)")
```




```
ggplot(july_cleaned, aes(x=TOTAL_DAY_CONS)) + geom_density(alpha=.3) + xlim(c(0,50000)) +
  ggtitle("Daily energy usage in July (w/out outliers)")
```



Now, lets check for normality to see if we can use a t-test to determine if the difference in energy consumption in July is significantly higher than in January. To do so, we will use the "jarque.bera" test; if our p-value is low (<0.005) we will conclude that our samples are not normally distributed and therefore will need to use non-parametric test instead of a t-test.

```
#Normality test for January dataset
jarque.bera.test(january_cleaned$TOTAL_DAY_CONS)
```

```
##
##  Jarque Bera Test
##
## data:  january_cleaned$TOTAL_DAY_CONS
## X-squared = 1.4125e+11, df = 2, p-value < 2.2e-16
```

```
#Normality test for July dataset
jarque.bera.test(july_cleaned$TOTAL_DAY_CONS)
```

```
##
##  Jarque Bera Test
##
## data:  july_cleaned$TOTAL_DAY_CONS
## X-squared = 1972600, df = 2, p-value < 2.2e-16
```

Based on the jarque.bera tests above, both samples (January and July) are **not normally distributed**. Therefore, we will need to use a non-parametric test to determine whether the difference in mean between the two samples are significantly different.

- We will use Wilcoxon.test to determine if january average is lower than july. If we get a p-value< 0.05 we will reject the null hypothesis (january_average == july average) and conclude that january_average < july_average.

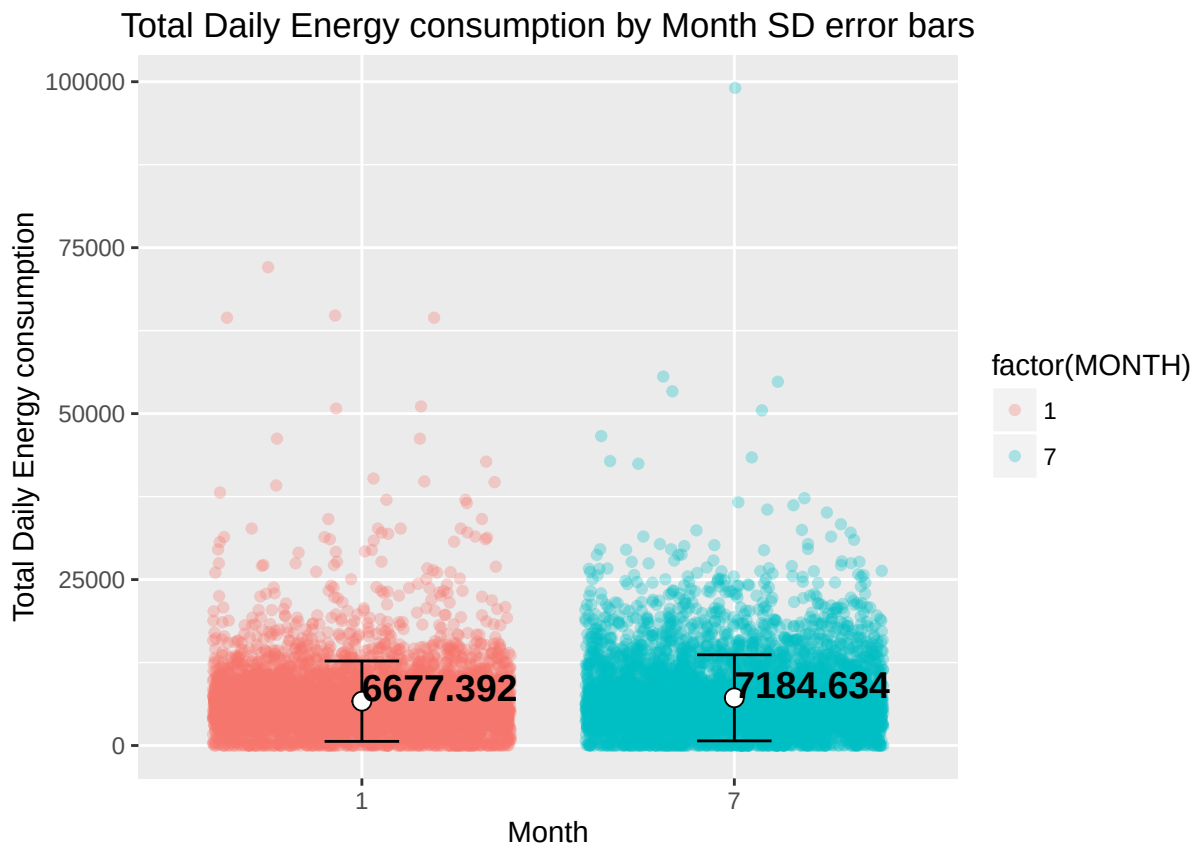
```
wilcox.test(january_cleaned$TOTAL_DAY_CONS, july_cleaned$TOTAL_DAY_CONS, alternative = "less")
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  january_cleaned$TOTAL_DAY_CONS and july_cleaned$TOTAL_DAY_CONS
## W = 1.0236e+10, p-value = 5.088e-06
## alternative hypothesis: true location shift is less than 0
```

We obtained a p-value< 5.088e-06, and therefore we can conclude that **energy consumption in January is significantly lower than in July**. Given that temperatures in Madrid in July vary between 30º and 40º and that AC usage is fairly common, we could conclude that AC usage has a clear impact on the energy usage across the Madrid region.

Lets visualize the difference using a bar plot with the average energy consumption per month and standard deviation bars.

```
temp_data_jitter <-
  daily_energy_dataset %>% filter(CITY == "MADRID", PRODUCT %in% c("P13", "P30", "P17")) %>%
  group_by(MONTH, CLIENT_ID) %>%
  summarise(av_hourly_consumption = mean(TOTAL_DAY_CONS))

daily_energy_dataset %>% filter(CITY == "MADRID", PRODUCT %in% c("P13", "P30", "P17")) %>%
  group_by(MONTH) %>%
  summarise(av_hourly_consumption = mean(TOTAL_DAY_CONS), std = sd(TOTAL_DAY_CONS)) %>%
  ggplot(data=., aes(x = factor(MONTH), y = av_hourly_consumption)) +
  geom_jitter(data = temp_data_jitter, mapping=aes(x = factor(MONTH), y = av_hourly_consumption, colour
  geom_errorbar( aes(ymin=av_hourly_consumption-std, ymax=av_hourly_consumption+std),
                 width = .2,                    # Width of the error bars
                 position = position_dodge(.9)) +
  geom_point(shape = 21, size = 3, fill = "white") + geom_text(aes(label=round(av_hourly_consumption, 3
  xlab("Month") +
  ylab("Total Daily Energy consumption") +
  ggtitle("Total Daily Energy consumption by Month SD error bars") +
  scale_fill_discrete(guide=guide_legend(title = "Months"))
```



Deep-dive Madrid: 3. Correlation of weather data with energy consumption

In the previous section we have concluded that indeed the energy consumption in Madrid for the month of July is significantly higher than in January, most certainly due to the use of AC units. Now, we are going to determine if there is correlation between temperature of a given day and the energy consumed for that day. For this specific test, we are taking a new sample from the large ENDESA dataset, with the following additional criteria:

- **Products**: P13, P30, P17
- **CITY**: MADRID
- **MONTHS**: MAY, JUNE, JULY. I decided to include June too as from a weather perspective it shows a bit more variability in temperatures.

External Weather dataset for Madrid

In addition, I have downloaded the **historical weather data for MAY, JUNE, JULY in Madrid in 2015** which we will use to determine if there is a correlation between energy consumption and the weather parameters (temp, humidity, rain). Data was downloaded from www.wunderground.com.

Loading weather dataset and formating the date column

```
weather_madrid = read.csv("TemperatureData/Madrid_weather_may_jun_jul.csv", sep = ";", header = TRUE)
str(weather_madrid)
```

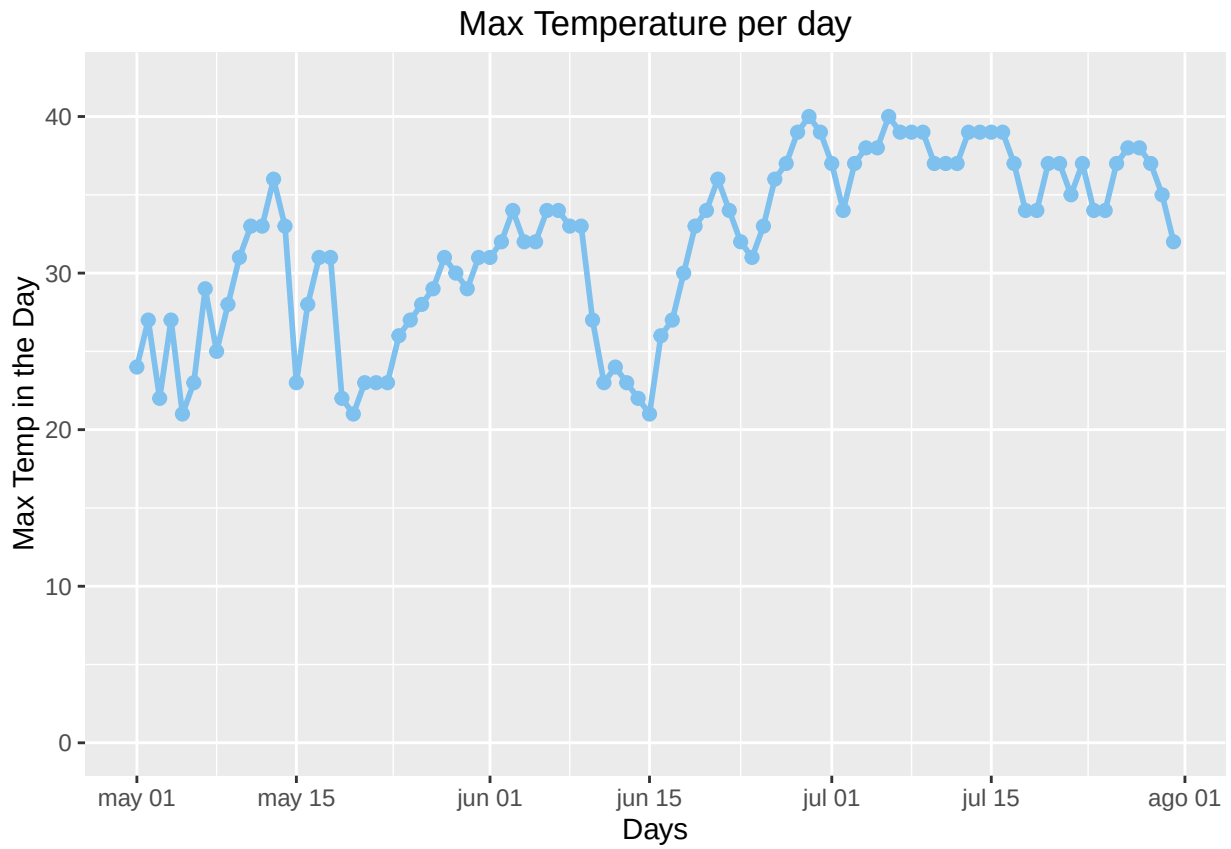
```
## 'data.frame':    92 obs. of  5 variables:
```

```
##  $ CEST              : Factor w/ 92 levels "01/05/2015","01/06/2015",..: 1 4 7 10 13 16 19 22 25 28
##  $ Max.TemperatureC  : int  24 27 22 27 21 23 29 25 28 31 ...
##  $ Max.Humidity      : int  82 100 82 88 77 93 81 67 88 72 ...
##  $ Max.Wind.SpeedKm.h: int  21 24 24 40 34 21 16 24 11 21 ...
##  $ Precipitationmm   : num  0 0 0 0 0 0 0 0 0 0 ...
```

```
weather_madrid$CEST <- sapply(weather_madrid$CEST, toString)
weather_madrid$CEST <- as.Date(weather_madrid$CEST, "%d/%m/%Y")
```

Visualizing the Weather dataset (temperature changes for the period selected)

```
weather_madrid %>%
ggplot(., aes(x = CEST, y = Max.TemperatureC)) + geom_line(size=1, colour = I("skyblue2"), aes(group =
```



Loading the new dataset with data for Madrid, for 3 specific products (P13, P30, P17) and for Jan, Feb, March, May, June, July

- This new dataset also includes data for Jan, Feb, March, which we will not use for the correlation analysis but we will be later used in the clustering analysis.

```
energy_madrid = read.csv("MadridSubset/Madrid_deep_dive_v2.csv", sep = ";", header = TRUE)
```

Lets clean-up the date column and include the following additional columns:

- DOW: Day of the week in number format (Sunday = 0)

- week_in_year: This column shows for a given day, the correspoding week number in the year.

```
energy_madrid$DAY <- sapply(energy_madrid$DAY, toString)
energy_madrid$DAY <- as.Date(energy_madrid$DAY, "%Y%m%d")
energy_madrid$DAY <- as.Date(energy_madrid$DAY, "%Y%m%d")

#Create additional column that indicates the day of the week
energy_madrid <- energy_madrid %>%
  mutate(DOW = as.POSIXlt(DAY)$wday)# TO-DO: CONVERT TO NUMBER as.POSIXlt(D)$wday

#Create additional column that shows the week of the year
energy_madrid <- energy_madrid %>%
  mutate(week_in_year = strftime(DAY, format = "%W"))
```

For the subsequent analysis we do not need the hourly energy consumption, so lets create a reduced dataset with just the total energy consumed per day and per client.

```
daily_energy_madrid <- energy_madrid %>%
  select(-MARKET, -TARGET_TENENCIA_CUPS,-CNAE) %>%
  mutate(AVG_CONS_PER_HR= ACTIVA_HT/24) %>%
  mutate(TOTAL_DAY_CONS = ACTIVA_HT) %>%
  select(-starts_with("ACTIVA_H"))

#Create additional column with the season information (jan-mar= winter, may-july = summer)
daily_energy_madrid <- daily_energy_madrid %>%
  mutate(SEASON = ifelse(MONTH <= 3, "winter","summer"))
```

Correlation 1: Correlation between temperature and energy consumption for may,june and july

The first step is to merge together our Madrid energy dataset (season = summer) with the external information about weather. The common ID to join both datasets is the DATE field

```
daily_energy_madrid_summer <- daily_energy_madrid %>%
  filter(MONTH >= 3)
daily_energy_madrid_summer_weather <- inner_join(daily_energy_madrid_summer, weather_madrid,by = c("DAY
str(daily_energy_madrid_summer_weather)
```

```
## 'data.frame':    528270 obs. of  14 variables:
##  $ CITY              : Factor w/ 1 level "MADRID": 1 1 1 1 1 1 1 1 1 1 ...
##  $ DAY               : Date, format: "2015-06-15" "2015-06-15" ...
##  $ CLIENT_ID         : int  77061 69210 76367 81227 86952 79974 79515 98939 99051 81949 ...
##  $ PRODUCT           : Factor w/ 3 levels "P13","P17","P30": 1 3 1 1 3 3 1 3 3 1 ...
##  $ MONTH             : int  6 6 6 7 6 5 6 7 7 7 ...
##  $ DOW               : int  1 1 5 4 3 2 3 5 3 1 ...
##  $ week_in_year      : chr  "24" "24" "25" "27" ...
##  $ AVG_CONS_PER_HR   : num  83.6 0 982.5 0 265.2 ...
##  $ TOTAL_DAY_CONS    : int  2006 0 23579 0 6365 6918 6560 5922 0 2775 ...
##  $ SEASON            : chr  "summer" "summer" "summer" "summer" ...
##  $ Max.TemperatureC  : int  21 21 36 39 34 33 34 32 37 38 ...
##  $ Max.Humidity      : int  100 100 63 52 59 72 59 65 49 50 ...
##  $ Max.Wind.SpeedKm.h: int  21 21 21 21 24 16 24 35 26 27 ...
##  $ Precipitationmm   : num  1.02 1.02 0 0 0 0 0 0 0 0 ...
```

Perform linear regression (lm function) for the **total energy consumed per day** and the temperature of the day

```
sat.mod.energy <- lm(TOTAL_DAY_CONS ~ Max.TemperatureC,
                     data=daily_energy_madrid_summer_weather)
summary(sat.mod.energy)
```

```
##
## Call:
## lm(formula = TOTAL_DAY_CONS ~ Max.TemperatureC, data = daily_energy_madrid_summer_weather)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -7246  -2888   -932   1626 135278
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        432.071     39.301   10.99   <2e-16 ***
## Max.TemperatureC   170.357      1.215  140.20   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4886 on 528268 degrees of freedom
## Multiple R-squared:  0.03588,    Adjusted R-squared:  0.03587
## F-statistic: 1.966e+04 on 1 and 528268 DF,  p-value: < 2.2e-16
```

Clear correlation between temperature of the day and the total energy consumed in a day for a given customer.

- p-value < 2.2e-16, therefore the correlation is clearly significant

Correlation 2: Correlation between temperature and avg.hourly energy consumption per day

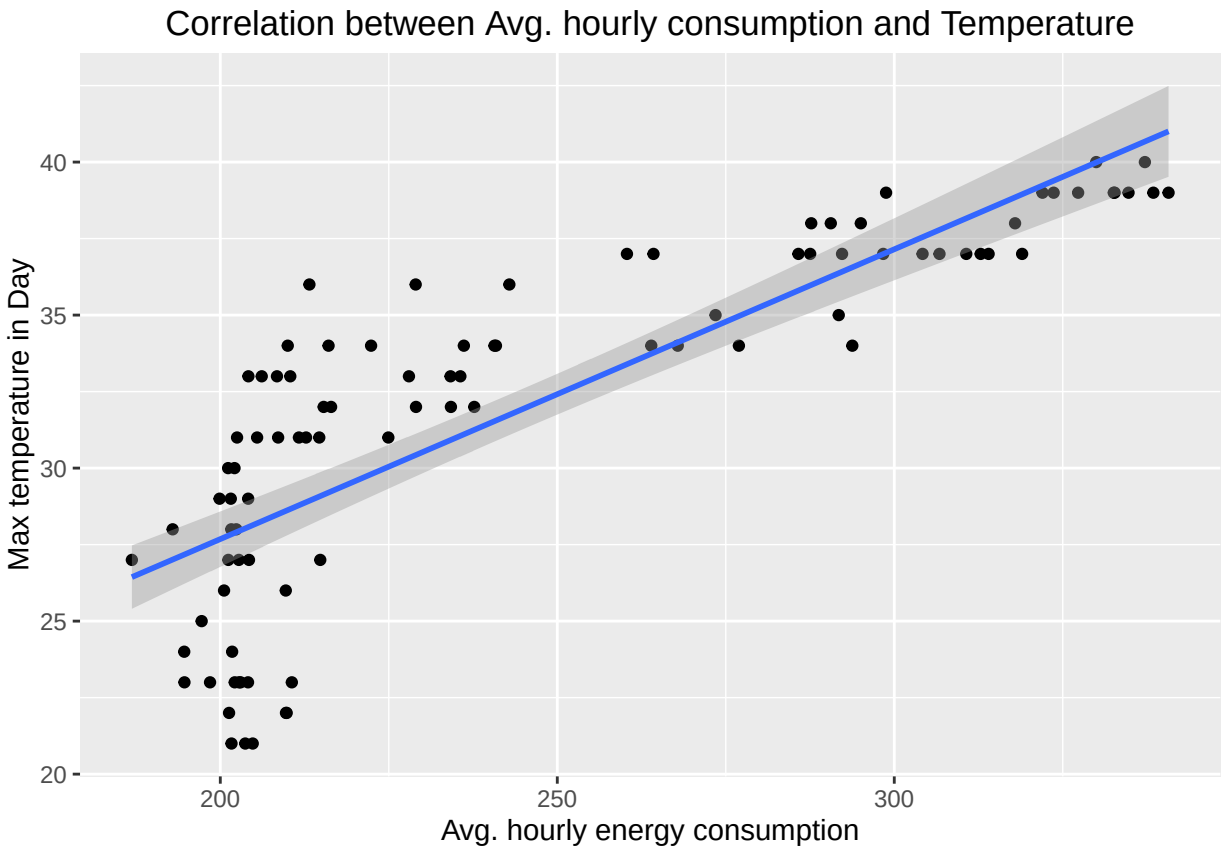
Now, lets see if there is a significant linear correlation between the average hourly energy consumption and the temperature. The average hourly energy consumption for a given day and customer is the result of the following simple calculation:
$ avg_hourly = total energy consumed in a day / 24 hrs. This variable is calculated earlier in the code.

First, lets visually observe the correlations with x-y plot and leveraging the smooth ggplot function.

```
#Average energy consumed per hour in the region per day
madrid_avg_hourly_consumption <- daily_energy_madrid_summer_weather %>%
  group_by(DAY) %>%
  summarise(avg_hourly = mean(AVG_CONS_PER_HR), temp = mean(Max.TemperatureC), humidity = mean(Max.Humi
```

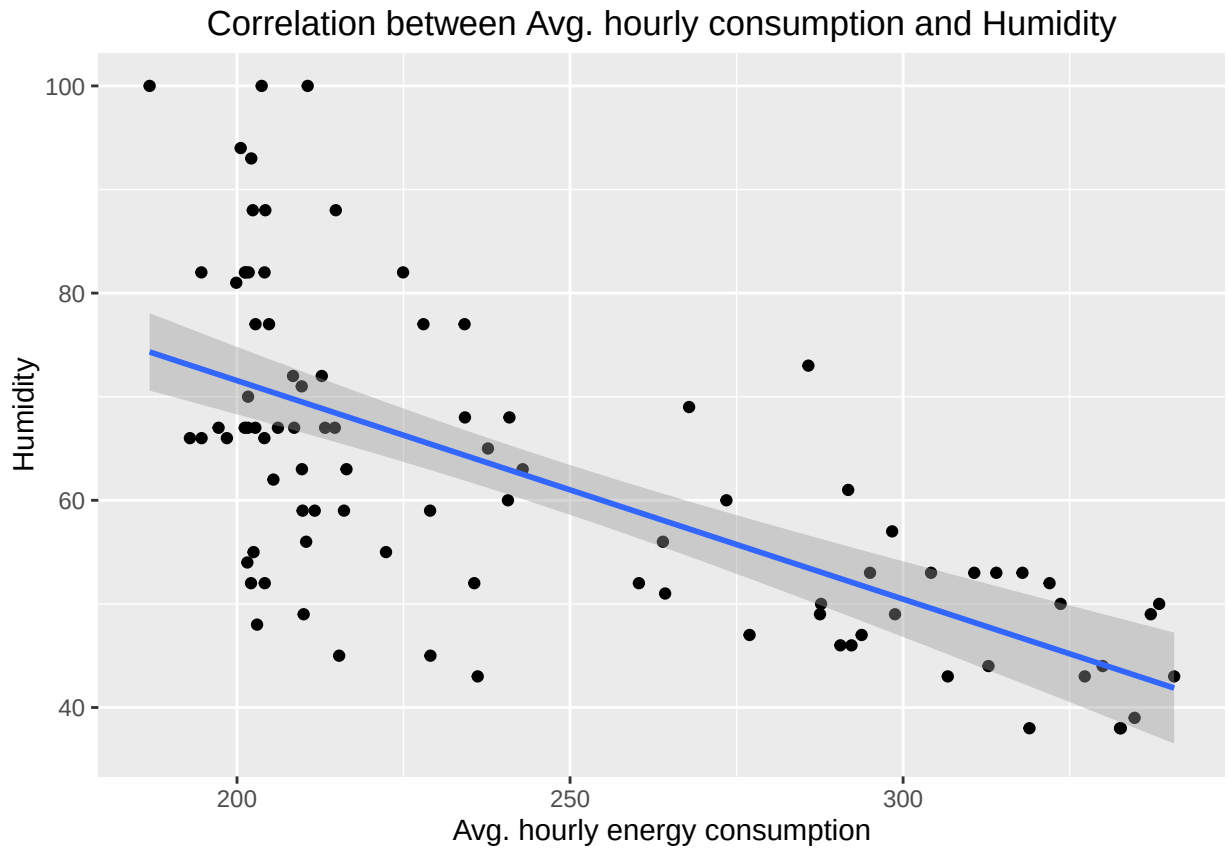
The plot below shows a clear correlation between average hourly consumption and temperature but we need to determine the p-value to confirm that it is statistically significant.

```
ggplot(madrid_avg_hourly_consumption, aes(x = avg_hourly , y = temp)) + geom_point() + geom_smooth(meth
```



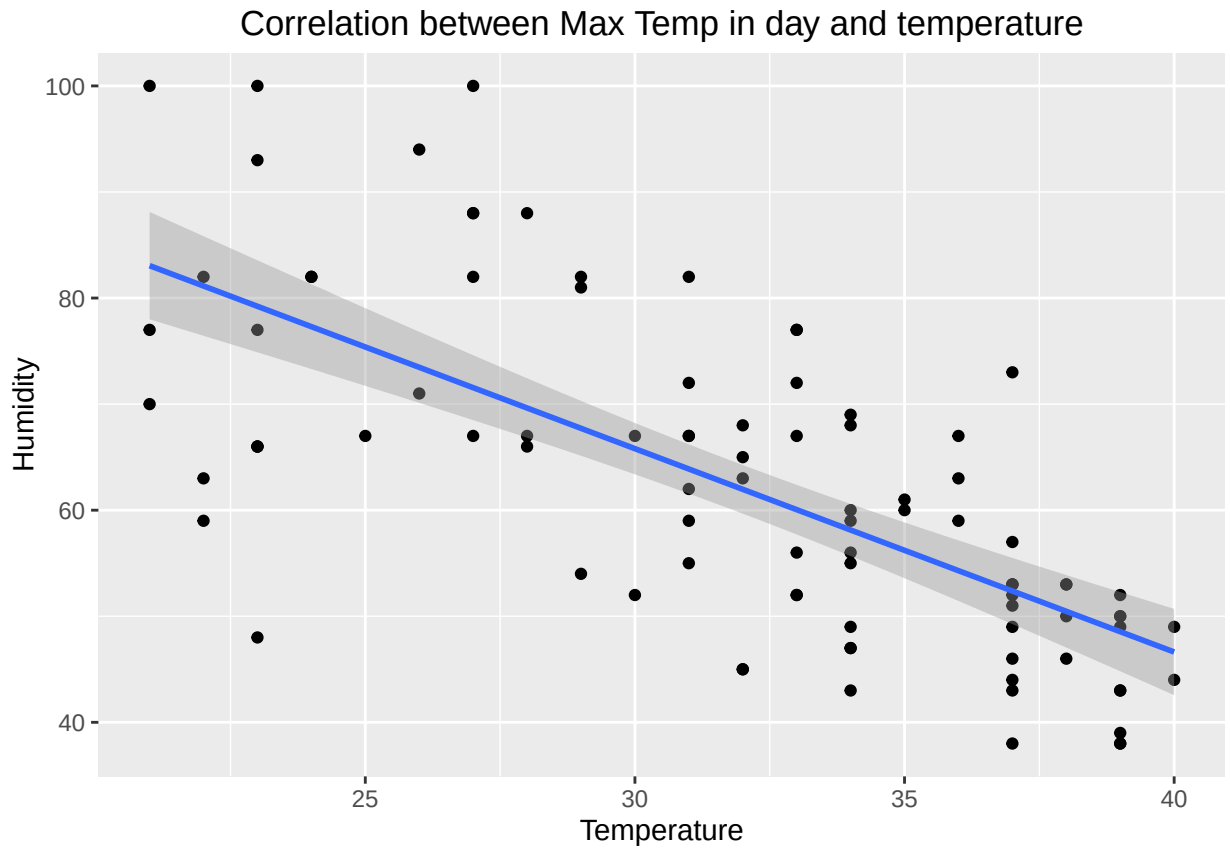
Just for exploration purposes, we also see that there is a clear negative correlation between average hourly consumption and humidity.

```
ggplot(madrid_avg_hourly_consumption, aes(x = avg_hourly , y = humidity)) + geom_point() + geom_smooth(
```



Again for pure exploration purposes, let's check the correlation between the temperature and humidity. From the plot we see that there is a clear negative correlation, which means that with higher temperatures in Madrid there is less humidity. From a weather perspective, Madrid has a very dry summer season and the hotter the day is the less humidity we experience. For our correlation model between temp and avg. energy consumed we could consider also adding the humidity variable into the model. However, it does not really add any value as there is already a very strong correlation of this variable with temperature (which we definitely want to include in the model).

```
ggplot(madrid_avg_hourly_consumption, aes(x = temp , y = humidity)) +  geom_point() + geom_smooth(metho
```

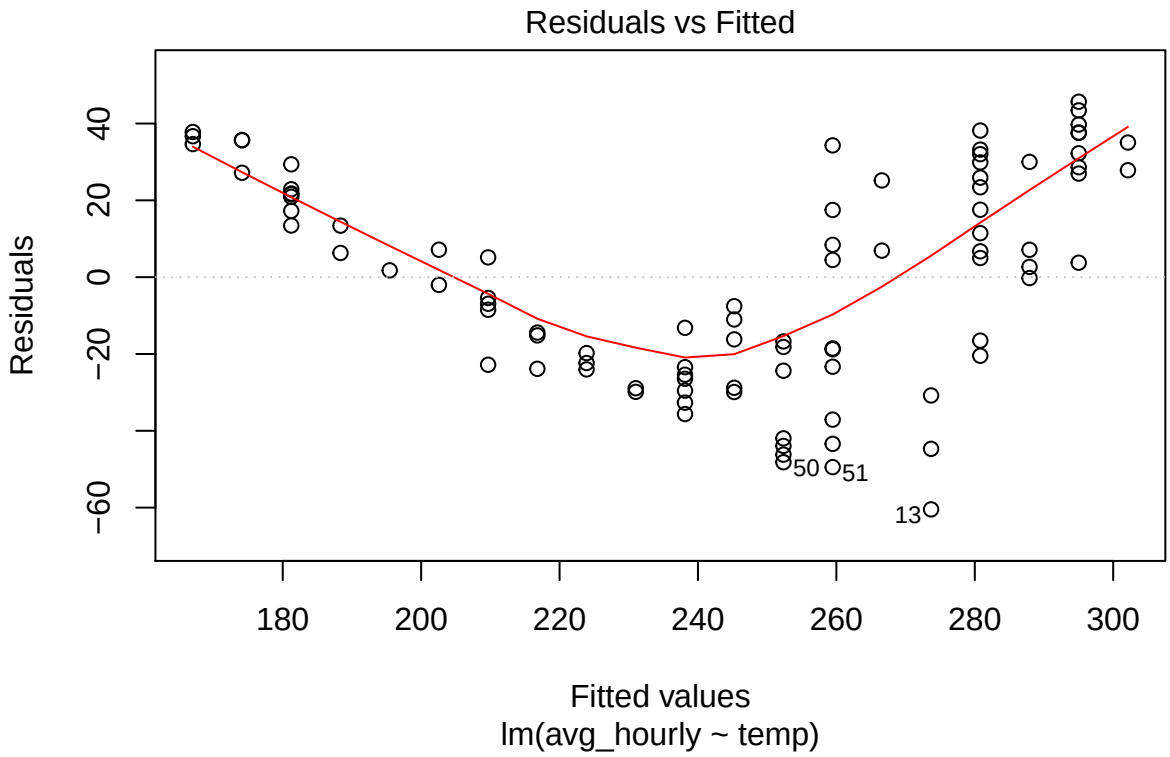


Regression model: Lets now create our regression model between temperature of the day and the avg. hourly energy consumed in a given day.

```
sat.mod.energy_2 <- lm(avg_hourly ~ temp,
                       data=madrid_avg_hourly_consumption)
summary(sat.mod.energy_2)
```

```
##
## Call:
## lm(formula = avg_hourly ~ temp, data = madrid_avg_hourly_consumption)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -60.47 -23.35   3.22  26.18  45.68
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   17.662     16.895   1.045    0.299
## temp           7.112      0.522  13.625   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27.71 on 90 degrees of freedom
## Multiple R-squared:  0.6735, Adjusted R-squared:  0.6699
## F-statistic: 185.6 on 1 and 90 DF,  p-value: < 2.2e-16
```

```
plot(sat.mod.energy_2, which = c(1))
```



Conclusions from the regression between avg. hourly energy consumption and temperature of the day:

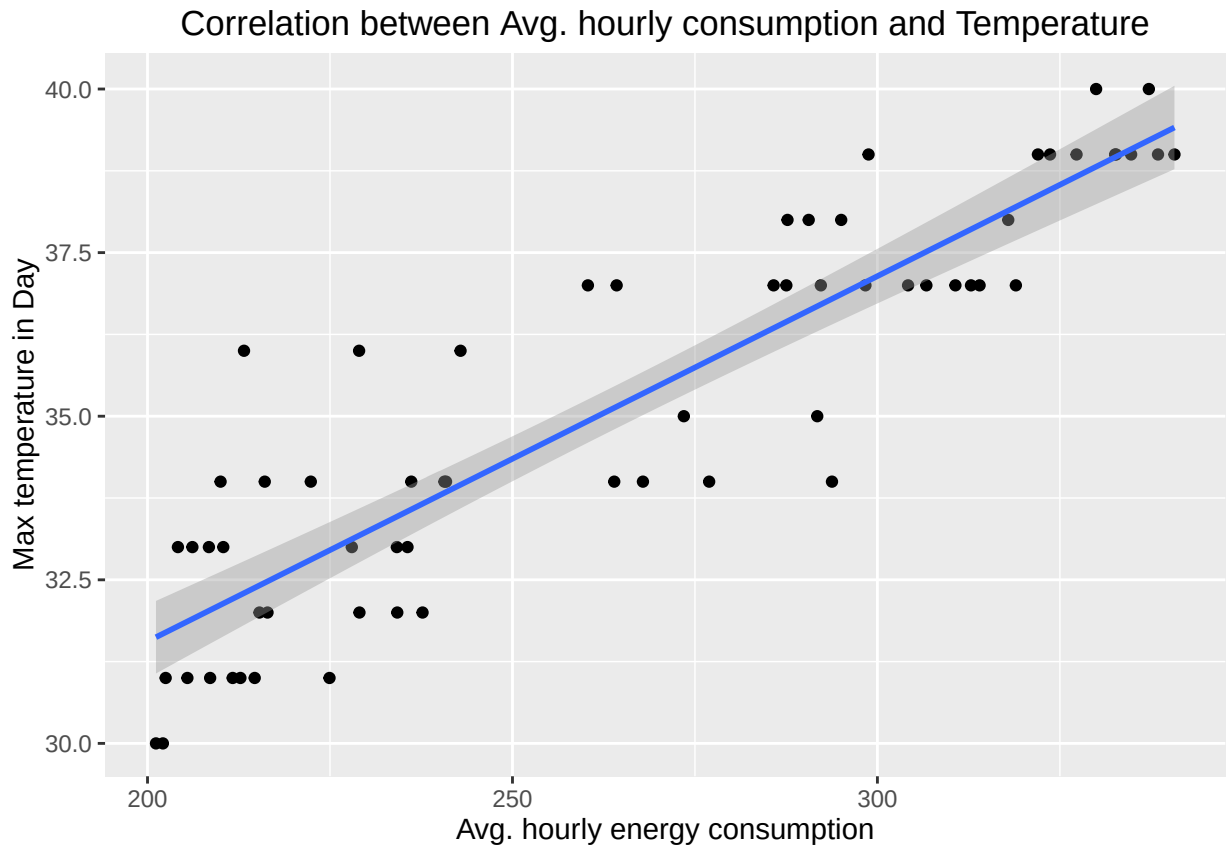
- **Significance**: There is a clear and strong correlation between the temperature and the avg consumption of energy for a given day: p-value $<$ 2.2e-16
- **Residuals**: Residual plots show that at lower levels of energy consumption the fit is skewed and therefore the linear model has room for improvement. The data points in the residuals plots should show a random pattern (i.e. random errors). In our case there is a clear visible pattern which means that there is a consistent and clear trend in the errors (fitted - actual data)

Investigating the **residuals plot** a bit further, we realize that in low temperature days the avg. hourly energy rate does not clearly correlate with the variation of temperature. We can then initially conclude that for lower temperature, exterior termperature and energy consumption are not clearly correlated. A very strong hypothesis could be that below a certain threshold of temperature, customers do not turn their AC units. On the other hand however, the hotter the day is the longer the AC units are kept on and therefore translates to higher avg. hourly energy consumption.

Looking both a the x-y plot with the correlation line and the residuals plot we can eye-ball that below 30 degree the correlation errors between the actual values and the correlation line is higher than for the rest of the value. This means that below 30 degrees customers do not turn their AC units and subsequently, temperature levels do not affect the avg. hourly consumption for a given day.

Therefore, if we take a subset of the days with **temp $>$ 30 degrees** we again see a strong positive correlation between hourly energy consumption but additionally, the residuals plot look less skewed.

```r
#Filterting out those days with temperatures lower than 30 degrees
days_high_temps <- madrid_avg_hourly_consumption %>%
  filter(temp >= 30)
ggplot(days_high_temps, aes(x = avg_hourly , y = temp)) + geom_point() + geom_smooth(method=lm) + labs(
```



Regression model on reduced dataset:

```r
#Linear regression model with new data set
sat.mod.energy_hightemps <- lm(avg_hourly ~ temp,
                               data=days_high_temps)
summary(sat.mod.energy_hightemps)
```

```
##
## Call:
## lm(formula = avg_hourly ~ temp, data = days_high_temps)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -63.247 -10.879   3.278  14.283  46.030
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -240.5294    32.2364  -7.461 3.72e-10 ***
## temp          14.3610     0.9146  15.702  < 2e-16 ***
## ---
```

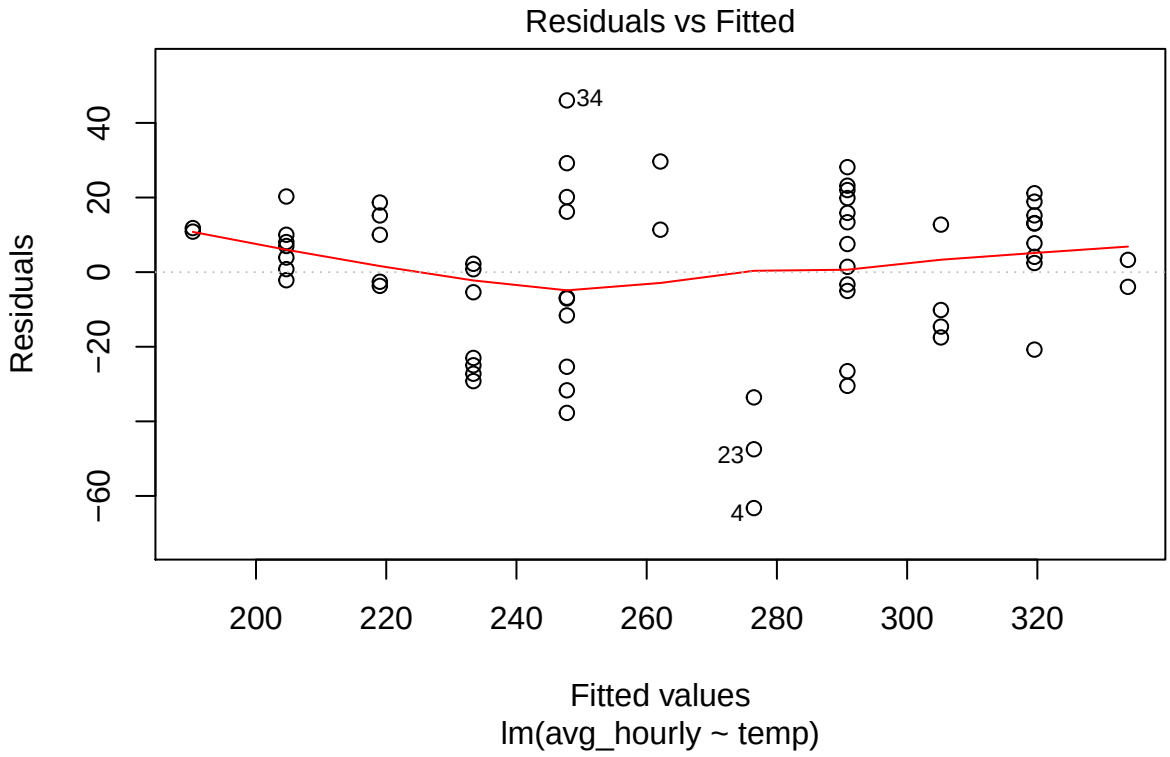
```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20.96 on 61 degrees of freedom
## Multiple R-squared:  0.8017, Adjusted R-squared:  0.7984
## F-statistic: 246.5 on 1 and 61 DF,  p-value: < 2.2e-16
```

Based on the plot above and the output of the regression model, we see that after getting rid of those data points with temperature < 30 degrees we are still getting a highly significant correlation.

Residuals plots of new regression model:

The residuals plot of the new regression models shows a random pattern, indicating a **much better fit** for a linear model when compared to the previous one.

```
#New residuals plot
plot(sat.mod.energy_hightemps, which = c(1))
```



Deep-dive Madrid: 4.Weekly behavior clustering

The goal of this section is to try to cluster together those customers that exhibit on average a similar energy consumption behavior during the week. With the proliferation of smartreaders, Utilities companies can now model the behavior of their customers to offer them tailored-made specific products, create more accurate demand models, etc. Therefore, clustering customers together based on their behavior can, for example, help Utilities' marketing teams design targeted electricity products and promotions.
For this specific analysis, we will first do some preliminary data cleansing before we feed the data to k-means clustering algorithm to get rid of outliers that might skew the clustering model.

For this clustering analysis we will use the same dataset from the previous section. The dataset includes daily energy readings for customers in Madrid and with the following constraints:

- Products: P13, P30, P17
- CITY: MADRID,
- MONTHS: January, Feb, March.

I selected the first three months of the year since:

- It is a low holiday season, therefore customers' weekly routines should be to some extent more consistent
- It is winter, therefore usage of AC units does not affect the energy variability. Also, since most houses in Madrid have natural gas heaters, the impact of low temperatures in electricity consumption would be none

Data cleansing before clustering

0. Create initial dataset

```
clean_madrid_data <- daily_energy_madrid %>%
  filter (SEASON == "winter") %>%
  group_by(CLIENT_ID, DOW) %>%
  summarise(avg_hourly_season = mean(AVG_CONS_PER_HR), std = sd(AVG_CONS_PER_HR))
```

Structure of the datasets: the dataset that we will be using for this clustering analysis is an aggregated view of the full dataset that shows the average and standard deviation of the daily avg. hourly energy consumed per customer and day of the week(DOW), In other words, for a given customer we have calculated the average and std.dev energy consumption per each day of the week across the three months. We will run this dataset through a k-means algorithm to see if there are any relevant clusters that group together customers with similar weekly energy consumption patterns.

Below it is shown the first 10 rows of the dataset:

```
head(clean_madrid_data)
```

```
## Source: local data frame [6 x 4]
## Groups: CLIENT_ID [1]
##
##   CLIENT_ID   DOW avg_hourly_season      std
##       (int) (int)             (dbl)    (dbl)
## 1     20630     0          271.1632 14.71133
## 2     20630     1          306.3646 16.80244
## 3     20630     2          308.6389 16.71939
## 4     20630     3          310.7689 15.53511
## 5     20630     4          308.1856 19.25209
## 6     20630     5          305.1439 17.61068
```

1. Identify Which customers have had no consumption during the three months; will later be removed from the sample

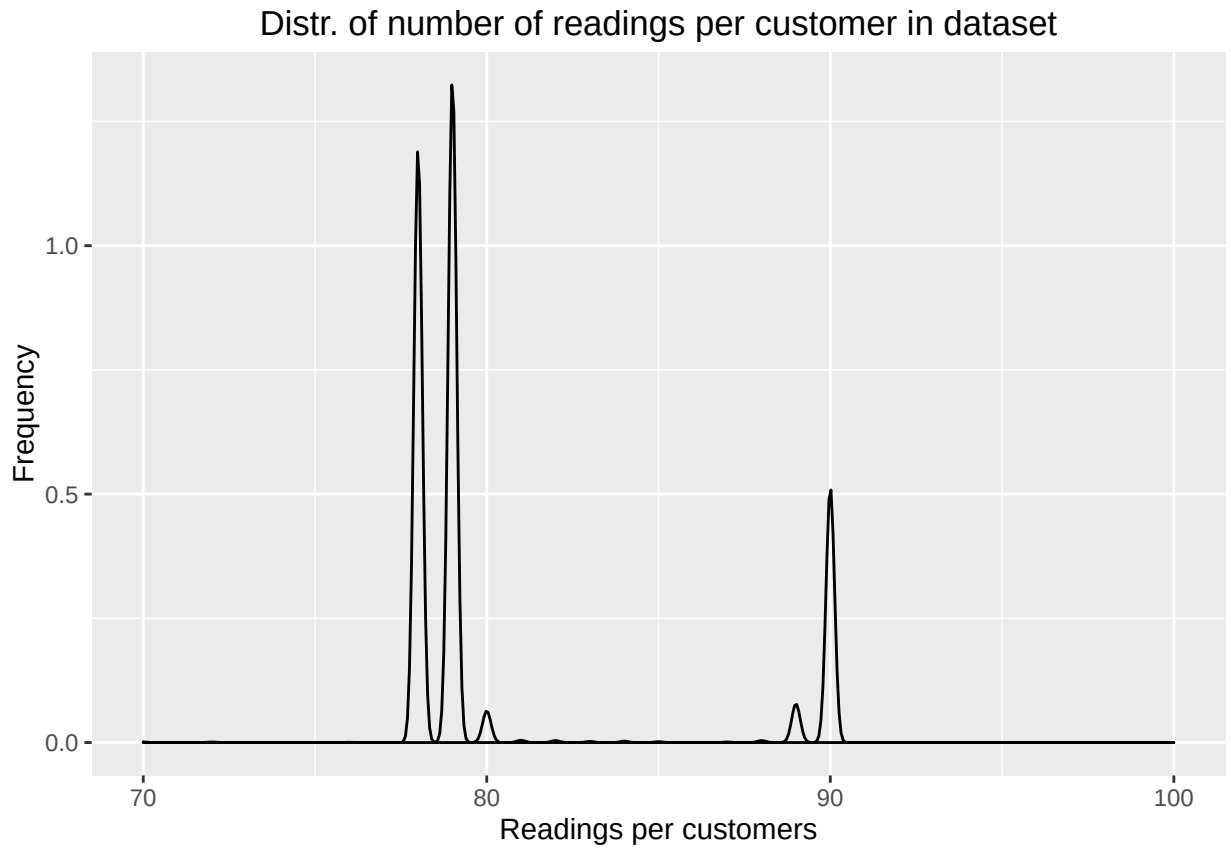
```
clients_cero <- daily_energy_madrid %>%
  filter (SEASON == "winter") %>%
  group_by(CLIENT_ID) %>%
  summarise(avg_hourly_season = mean(AVG_CONS_PER_HR)) %>%
  filter(avg_hourly_season <= 0)
```

2. Identify which customers have close to a full 3-months of data Lets first calculate how many total daily readings we have for each of the customersin the dataset. Ideally, all customers should have close to 90 datapoints (30days x 3 months)

```
missing_client_data <- daily_energy_madrid %>%
  filter (SEASON == "winter") %>%
  group_by(CLIENT_ID) %>%
  summarise(readings = n_distinct(DAY))
```

Lets create a simple density plot to see the distributions of the number of readings per customers in the dataset. In the plot below, we clearly see that there is a significant number of customers that are **missing around 10-15 days of energy readings**. To have enough population to run our clustering algorithm but at the same time ensure that the customers included have enough data to extrapolate their weekly behavior, we are going to get rid of all customers with less than 80 days of readings. In addition, we are also getting rid of those customer with 0 energy consumed in the last 3 months.

```
ggplot(missing_client_data, aes(x=readings)) + geom_density(alpha=.3) + xlim(c(70,100)) +
  ylab("Frequency") +
  xlab("Readings per customers") + ggtitle("Distr. of number of readings per customer in dataset")
```



Lets clean up our original dataset and get rid of the customers with no energy consumption in the three months and those customers that do not have enough data points (<75)

```
missing_client_data <- missing_client_data %>%
  filter(readings >= 75)
```

```r
n_distinct(missing_client_data$CLIENT_ID)
```

```
## [1] 5670
```

```r
#Get rid of customers with no energy consumption
clean_madrid_data <- clean_madrid_data %>%
  filter (!CLIENT_ID %in% c(clients_cero$CLIENT_ID))

clean_madrid_data <- clean_madrid_data %>%
  filter (CLIENT_ID %in% c(missing_client_data$CLIENT_ID))
n_distinct(clean_madrid_data$CLIENT_ID)
```

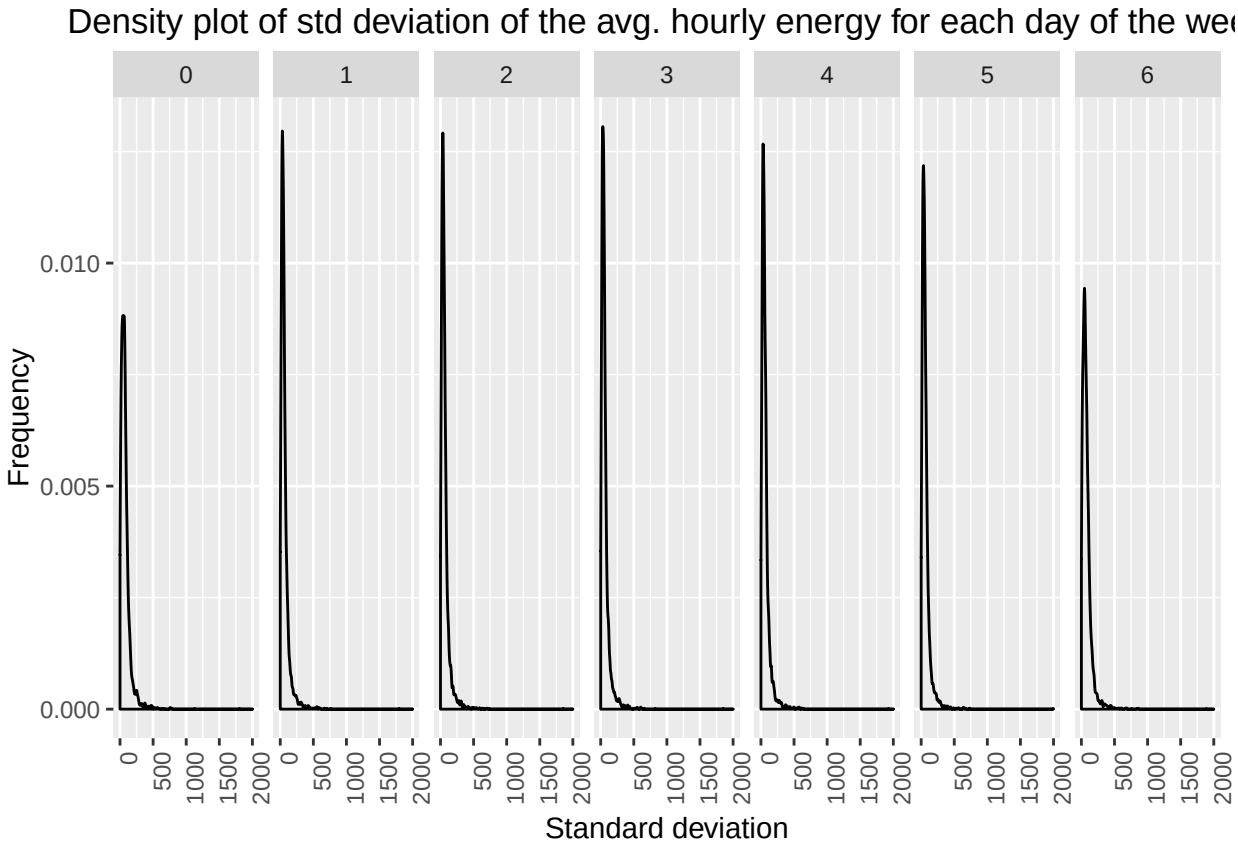
```
## [1] 5627
```

3. Identify customers with high variability

Now, after taking the average and Std. Dev. of the avg. consumption per day of the week for each of the customers in scope, it is also important to leave out of our clustering model those customers that exhibit a significantly high variability in their weekly data. To do so, let's see how the distribution of the standard deviation by day of the week for all customers in the dataset look like.

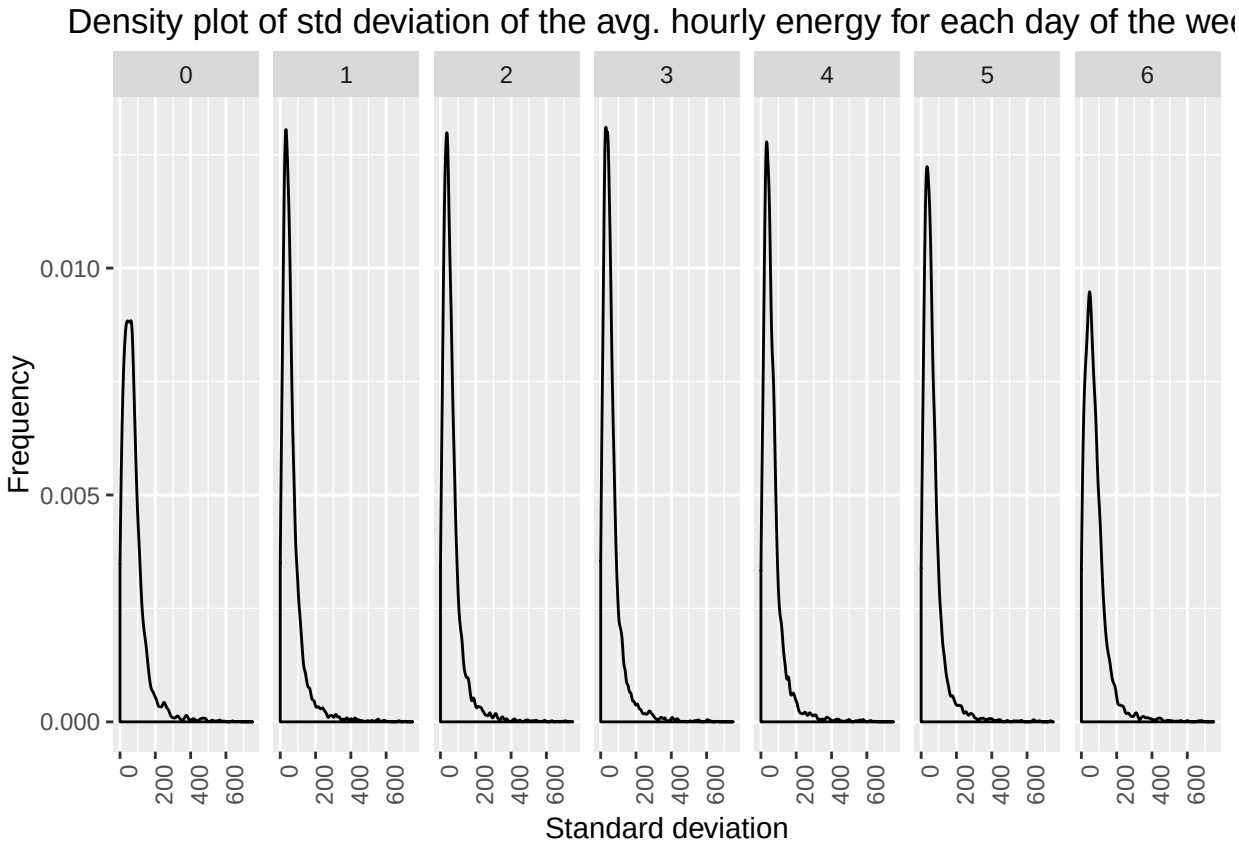
Density plots below show the distribution of the Std. Dev. per day of the week

```r
ggplot(clean_madrid_data, aes(x=std)) + geom_density(alpha=.3) + xlim(c(0,2000)) + facet_grid (.~DOW) +
  ylab("Frequency") +
  xlab("Standard deviation") +
  ggtitle("Density plot of std deviation of the avg. hourly energy for each day of the week") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



Zooming to identify if there is a reasonable cut-off that we should use to exclude customers with comparably high standard deviations per day of the week.

```
ggplot(clean_madrid_data, aes(x=std)) + geom_density(alpha=.3) + xlim(c(0,750)) + facet_grid (.~DOW) +
  ylab("Frequency") +
  xlab("Standard deviation") +
  ggtitle("Density plot of std deviation of the avg. hourly energy for each day of the week") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



By looking at the frequency plot, we can eye-ball that most of the customers have a Std. Dev. below 200 for each day of the week. Therefore, we could consider all the instances with Std. Dev. > 200 as outliers. Taking an aggresive approach, lets get rid of customers that have at least one of their days of the week average hourly consumption with standard deviation higher that 200.

To identify these customers that have at least of the days of the week with Std. Dev. higher than 200, lets add up their weekly standard deviation and filter out those that their sums of weekly std is higher than 1400 (200x7). We will be effectively leaving out of the analysis those customers that for the average consumption of energy per day of the week, have at least one of the days with higher than usual variability.

```
outliers <- clean_madrid_data %>%
  group_by(CLIENT_ID) %>%
  summarize(sum_std = sum(std)) %>%
  filter(sum_std > 1400)

n_distinct(outliers$CLIENT_ID)
```

```
## [1] 206
```

There are 206 customers that are flagging as outliers based on the criteria described above. So lets now remove these customers from our dataset.

```
clean_madrid_data <- clean_madrid_data %>%
  filter (!CLIENT_ID %in% c(outliers$CLIENT_ID))
```

After all cleanup the final number of customers that we are going to try to run through our clustering algorithm is 5421

```
n_distinct(clean_madrid_data$CLIENT_ID)
```

```
## [1] 5421
```

K-means clustering 1: all customers in scope

Dataset details:

- CITY: Madrid
- Timeframe: Jan-Mar 2015
- Products: P13, P17, P30
- Total Number of customers: 5421 (after removing outliers)

Goal and approach to clustering

At this point we have been able to clean our original dataset and remove all missing data and some customers that we considered outliers based on the criteria described above. As it was described previously, the dataset that we will run through a k-means algorithm is a summary of the average hourly consumption per day of the week for Madrid customers between January and March. The goal then is to be able to create a set of useful and significant clusters that group together customers that show similar weekly behavior both in terms of expending patterns (shape of the consumption line across the week) and the quantity of electricity that is consumed.

Clustering algorithm (k-means)

We will be using k-means algorithm, which is an unsupervised method to group together similar data points (similarity measured in terms of Euclidean distance). A priori, we do not know how many distinct clusters of customers we have in our dataset. Therefore, to determine what is the approximated number of clusters that best describe our dataset we will

- Run the k-means algorithm for several values of k (k 2 through 15) -> Using the WSSPLOT function below
- Measure the within-groups sums of squares (WSS) for each of the runs. The WSS measures
- Plot the total within-groups sums of squares against the number of clusters. A bend in the graph can suggest the appropriate number of clusters

Lets transform the dataset spreading the avg. hourly consumption data into columns, one for each day of the week. Then, lets normalize the dataset by centering the data around cero. For k-means clustering to work correctly, the data needs to be normalized (i.e. all variables should be represented in the same scale)

```
#Spread the dataset
clean_madrid_data_c1 <- clean_madrid_data
n_distinct(clean_madrid_data_c1$CLIENT_ID)
```

```
## [1] 5421
```

```
clean_madrid_data_cluster <- clean_madrid_data_c1 %>%
  select(-std) %>%
  spread(DOW, avg_hourly_season)

#Scale the data
data_scaled <- clean_madrid_data_cluster[,2:8]
data_scaled <- scale(data_scaled, center = TRUE)
```

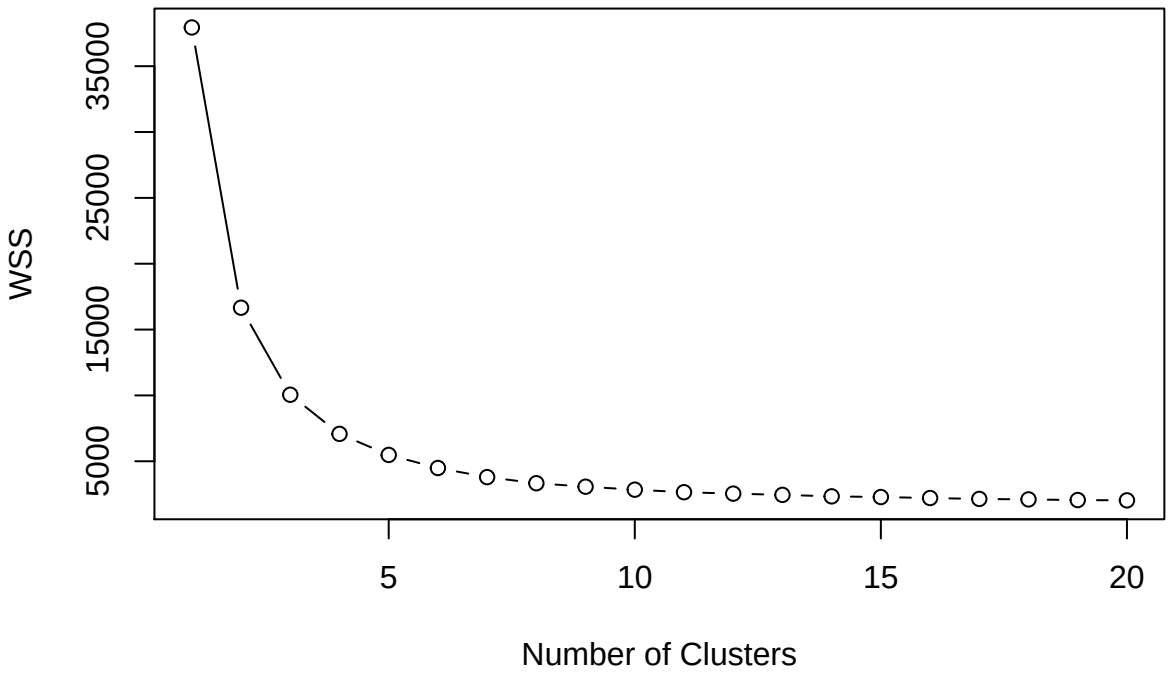
WSS function To be able to determine the number of clusters (k) that best suit the dataset, we are creating a function that runs the k-means algorithm from k=2 to the a maximum number of k specified in the parameters. For each run, the function will capture the WSS (measure of clusters compactness) to later be plotted against the number of clusters (WSS vs. # of clusters). As it was mentioned before, we will select the number of clusters where we see a "bend" in the curve of the plot. This specific point in the curve represents the point from which adding more clusters does not significantly decrease the WSS

Number of customers included in the dataset: **5421**

```
#Function to determine the optimal number of clusters
wssplot <- function(data, nc, seed=1234){
  wss <- (nrow(data)-1)*sum(apply(data,2,var))
  for (i in 2:nc){
    set.seed(seed)
    wss[i] <- sum(kmeans(data, centers=i)$withinss)
  }

  plot(1:nc, wss, type="b", xlab="Number of Clusters",
       ylab="WSS")
}
```

Lets runs the wssplot from k=2 to k=20 to determine the optimal number of clusters

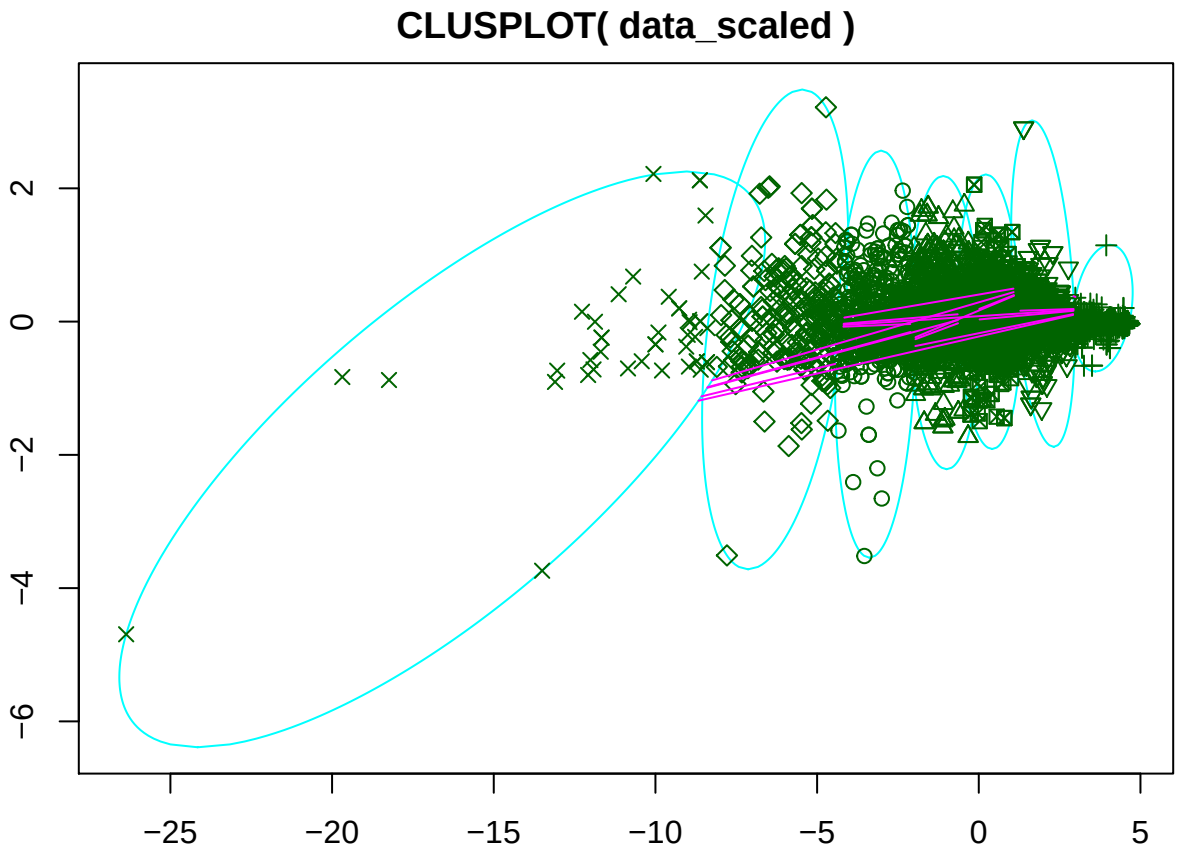
```
wssplot(data_scaled,20)
```



By looking at the WSS vs. # of clusters plot, we see that the "bend" occurs around k=8, which means that

after than point the compactness of the clusters do not improve significantly. Lets then run our k-means for k = 7 and explore the results we obtain:

```
fit.km_1 <- kmeans(data_scaled, centers = 7)
par(mar = rep(2, 4))
clusplot(data_scaled,fit.km_1$cluster)
```




```
#WSS (measure of compactness of clusters -> minimize)
fit.km_1$tot.withinss
```

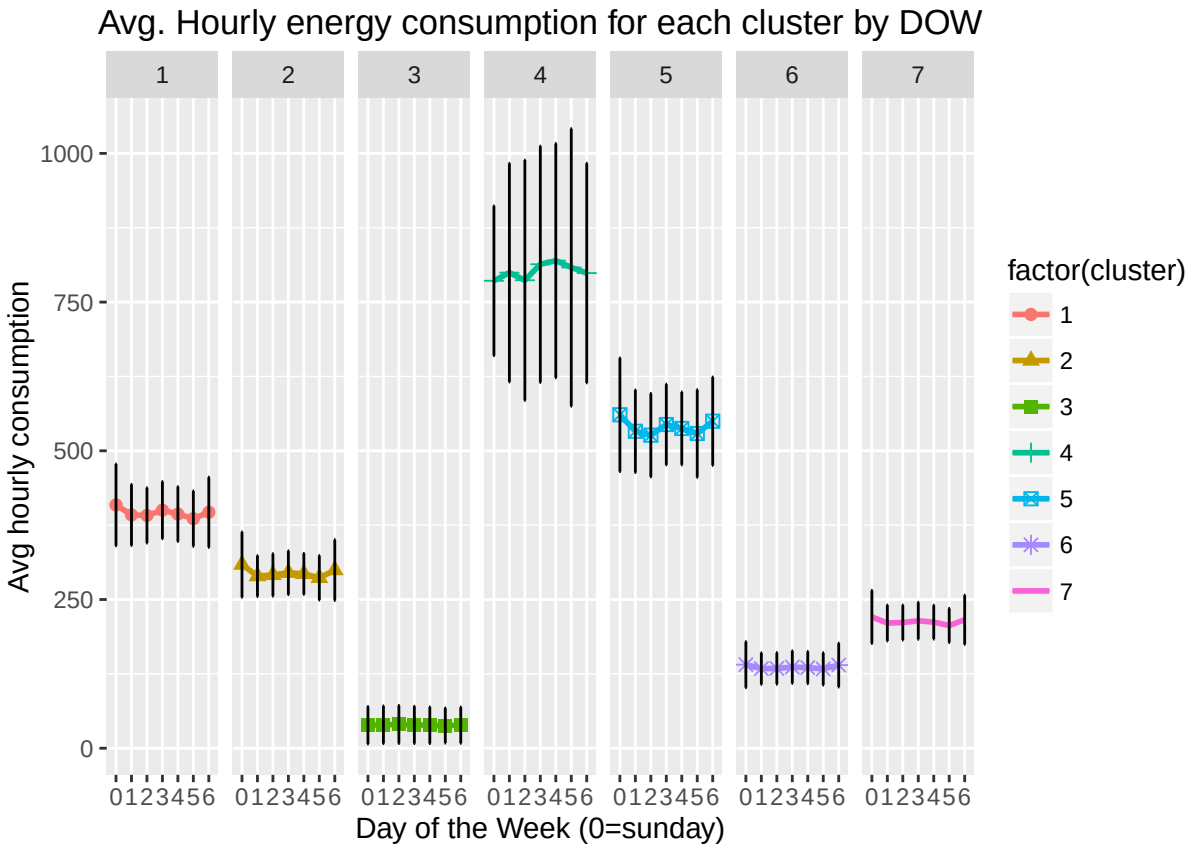
```
## [1] 3795.581
```

```
#BSS (measure of the distance between clusters -> maximize)
fit.km_1$betweenss
```

```
## [1] 34144.42
```

Now let's visualize how our clusters look like. To do so, we will append the clustering output to original dataset. We will then aggregate the energy consumption data by clusters and DOW to see the average weekly behavior of each of the clusters. We will also calculate the standard deviation for each cluster to determine the significance in term of variability.

```r
#Append cluster column to our original dataset
clean_madrid_data_cluster$cluster <- fit.km_1$cluster

#Aggregate by clusters and day of the week
clean_madrid_data_results <- clean_madrid_data_cluster %>%
  gather(key = "DOW", value = "avg_hourly_season",2:8) %>%
  group_by(DOW, cluster) %>%
  summarise(dow_avg_consumption = mean(avg_hourly_season), standDev = sd(avg_hourly_season))

#Visualizing the average hourly consumption per day of the week for each of the clusters
clean_madrid_data_results %>%
  ggplot(., aes(x = DOW, y = dow_avg_consumption, group = cluster, shape=factor(cluster), colour=factor
  ggtitle("Clusters") + scale_linetype_discrete(name="x") + facet_grid( . ~ cluster) +
  geom_errorbar(aes(x = DOW, ymin = dow_avg_consumption-standDev, ymax = dow_avg_consumption+standDev,
                width=.1, position=position_dodge(width = .1)) +
  ggtitle("Avg. Hourly energy consumption for each cluster by DOW")
```



From the plot above we can draw some conclusion about the clusters we obtained:

- Clusters represents groups of customers with different levels of hourly energy consumptions
- "Flat clusters": There are set of clusters that contain customers that seem to have a very regular consumption of energy throughout the week (flat lines without spikes)
  - For these clusters the std. dev. lines indicate low variability within the clusters, and therefore very significant behavior among the customers included in these clusters

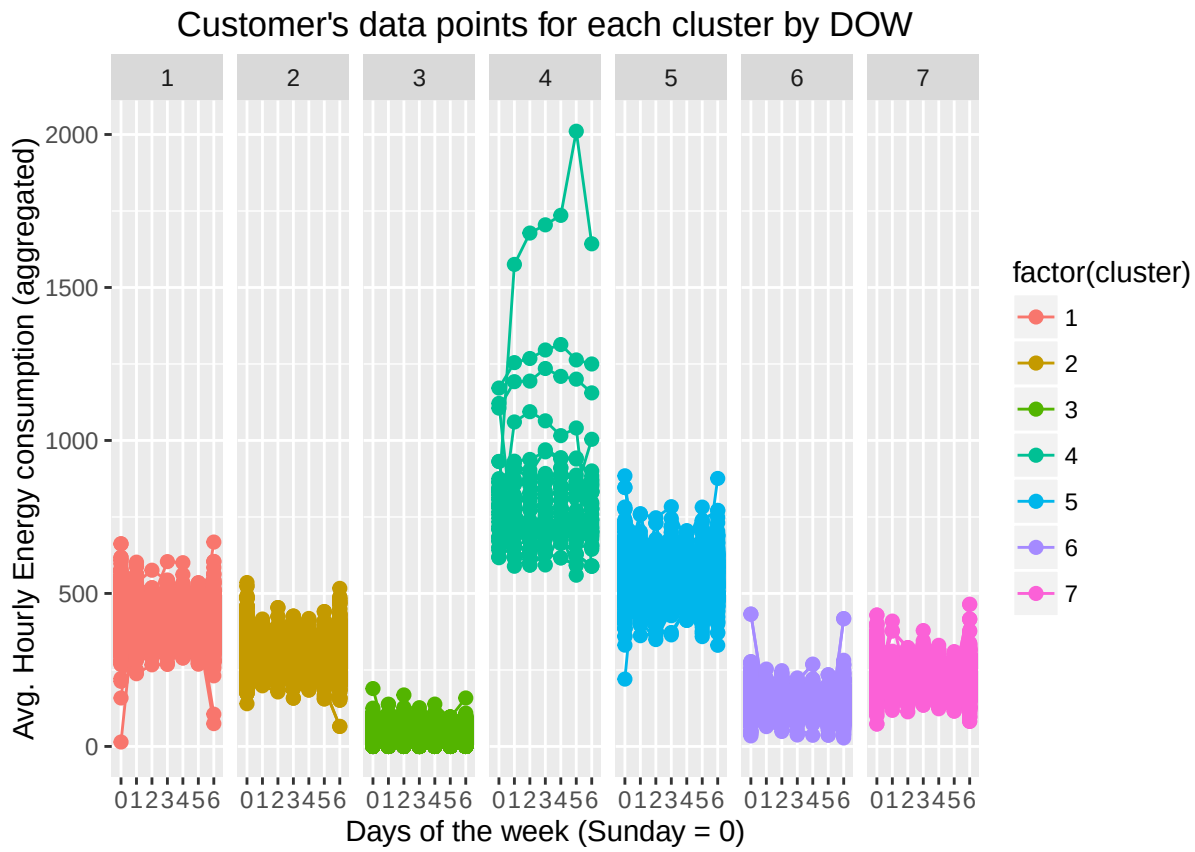
- "Spike clusters": Some of the other clusters exhibit a less constant behavior with spikes of avg. hourly energy consumption for certain days of the week.

With this kind of information, Utilities companies can get a better understanding of how their customer base behave: groups of customers that tend to spend more time at home during the weekends vs. the party-goers, group customers based on their average energy usage etc. Having a more detailed profile of your customers can potentially be a differentiating factor when designing new products tailored to specific types of customers or to plan promotions targeted to customers based on their energy usage.

In the visualization below, we can get a better sense of the volume of customers included on each of the clusters. This can also be an additional criteria to determine which clusters are really significant.

Shown below is the data from each customer for each clusters and day of the week (not aggregated):

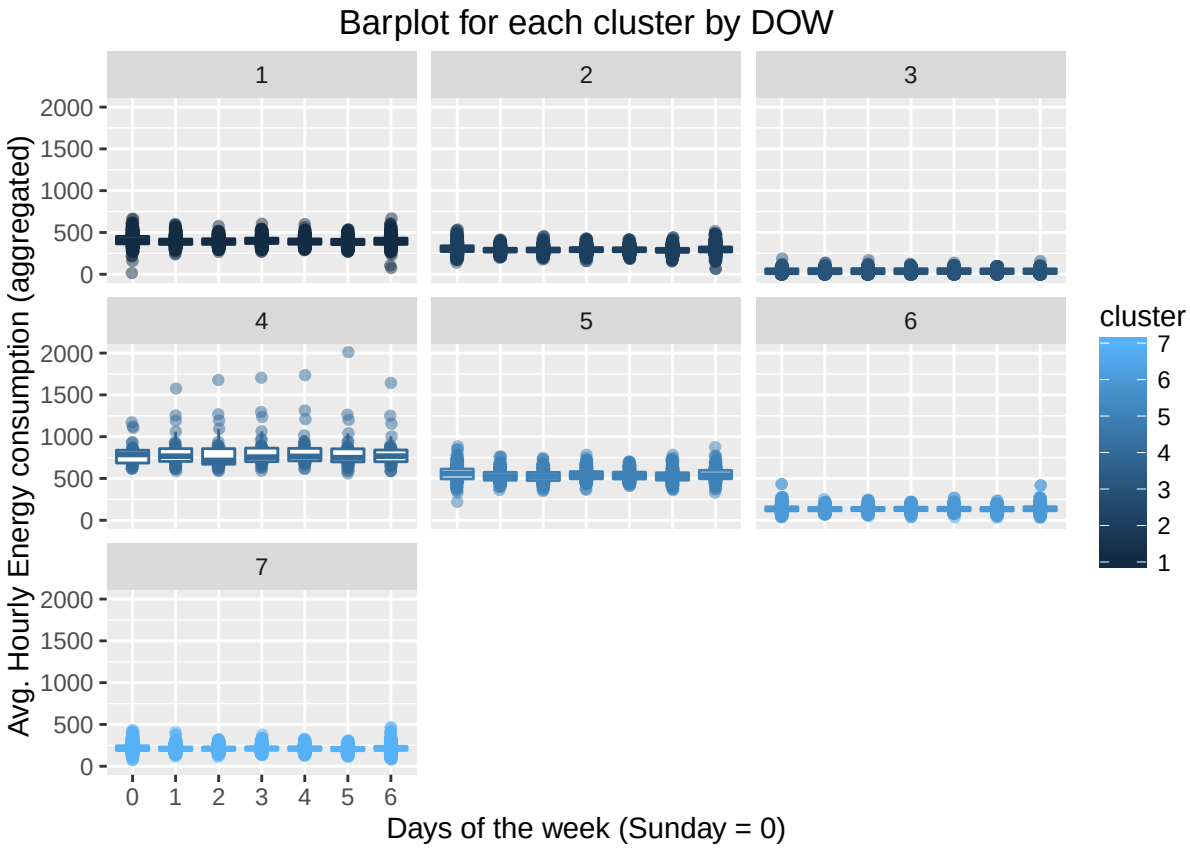
```
clean_madrid_data_cluster %>%
  gather(key = "DOW", value = "avg_hourly_season",2:8) %>%
  ggplot(., aes(x = DOW, y = avg_hourly_season, group = CLIENT_ID, colour=factor(cluster))) + geom_line
```



Shown below is the data from each customer for each clusters and day of the week using a barplot to better understand the distribution of values per cluster

```
clean_madrid_data_cluster %>%
  gather(key = "DOW", value = "avg_hourly_season",2:8) %>%
  ggplot(.,aes(x=DOW , y=avg_hourly_season)) +
  scale_x_discrete() +
  geom_jitter(aes(colour = cluster, x = DOW),
```

```
            position = position_jitter(width = .05), alpha = 0.5) +
  geom_boxplot(aes(colour = cluster), outlier.colour = NA, position = "dodge") +
  facet_wrap(~ cluster) + labs(title = "Barplot for each cluster by DOW", x = "Days of the week (Sunday
```



K-means clustering 2: customers with product P30

Dataset details:

- CITY: Madrid
- Timeframe: Jan-Mar 2015
- Products: P30
- Total Number of customers: 3563 (after removing outliers)

We saw in the previous k-means analysis that the clustering algorithm was grouping customers not just based on their energy usage patterns (i.e. spikes vs. flat behavior) but also based on the different levels of energy consumed (i.e. high values of avg. hourly energy consumption). We are interested in understanding the usage patterns to, for example, cluster together customers that tend to stay home during the weekends or customers that spend most of their weekly energy during the middle of the week. Therefore, if we just want to focus our clustering algorithm to group customers based on their usage pattern we should avoid mixing together customers with different levels (i.e. amount) of avg. energy consumptions. Therefore, to do so in this part of the analysis we will follow the same clustering approach as we did in the previous section, but this time just with customers that have contracted product P30. This will to some extent ensure that all customers analyzed on average spend the same amount of energy (in orders of magnitude at least).

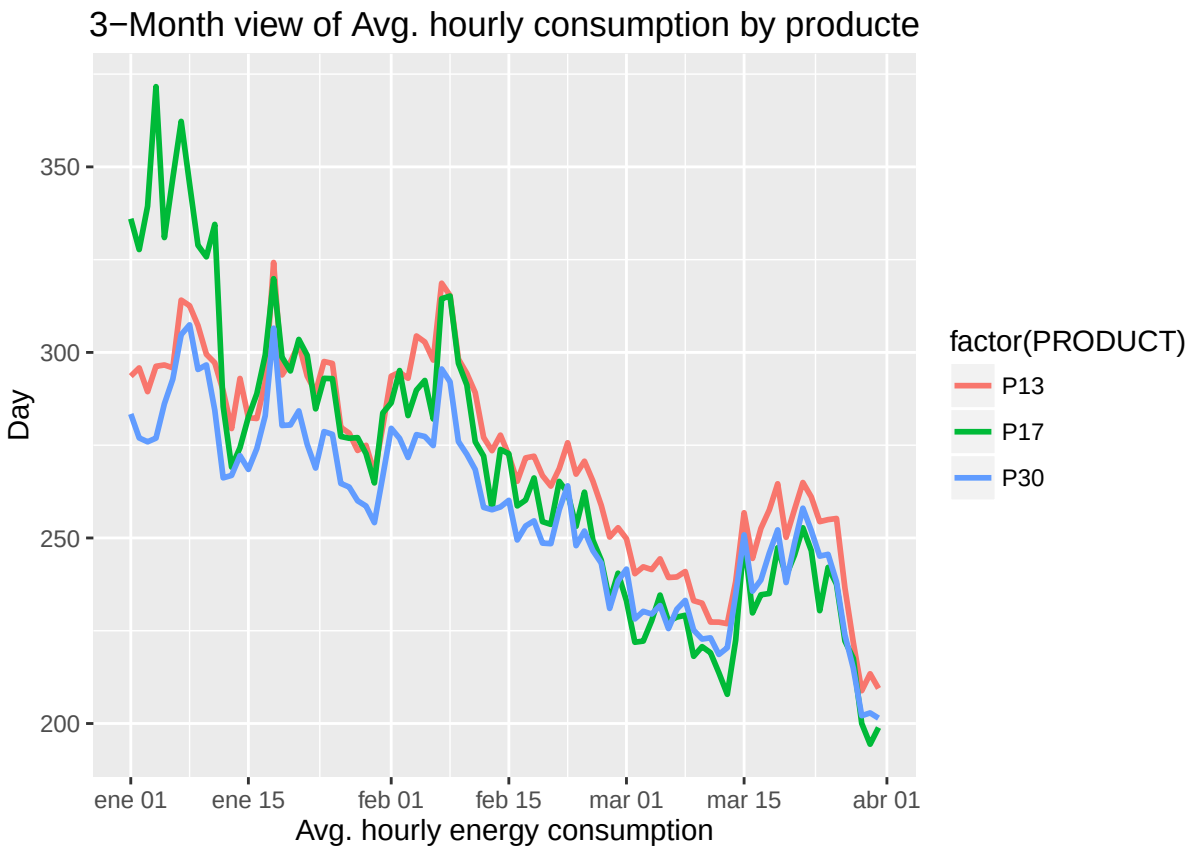
Preparing the dataset getting rid of the clients with cero consumption and those with missing data (identified in previous sections)

```
daily_energy_madrid_clus2 <- daily_energy_madrid %>%
  filter (!CLIENT_ID %in% c(clients_cero$CLIENT_ID))

daily_energy_madrid_clus2 <- daily_energy_madrid_clus2 %>%
  filter (CLIENT_ID %in% c(missing_client_data$CLIENT_ID))
n_distinct(daily_energy_madrid_clus2$CLIENT_ID)
```

```
## [1] 5627
```

In the plot below we can see that indeed there is a visible difference in the level of energy usage between customers in P13, P17 and P30. Lets focus our clusterting algorithm on P30 which we know from previous findings that is the product with the highest number of customers.

```
daily_energy_madrid_clus2 %>%
  filter (SEASON == "winter") %>%
  group_by(PRODUCT, DAY) %>%
  summarise(avg_cons = mean(AVG_CONS_PER_HR)) %>%
  ggplot(., aes(x = DAY, y = avg_cons,colour = factor(PRODUCT))) + geom_line(size=1) + labs(title = "3-M
```



Select customers in P30 and aggregate the dataset to be feed into the k-means algorithm. As we see from the output below there are 3563 customers in this dataset.

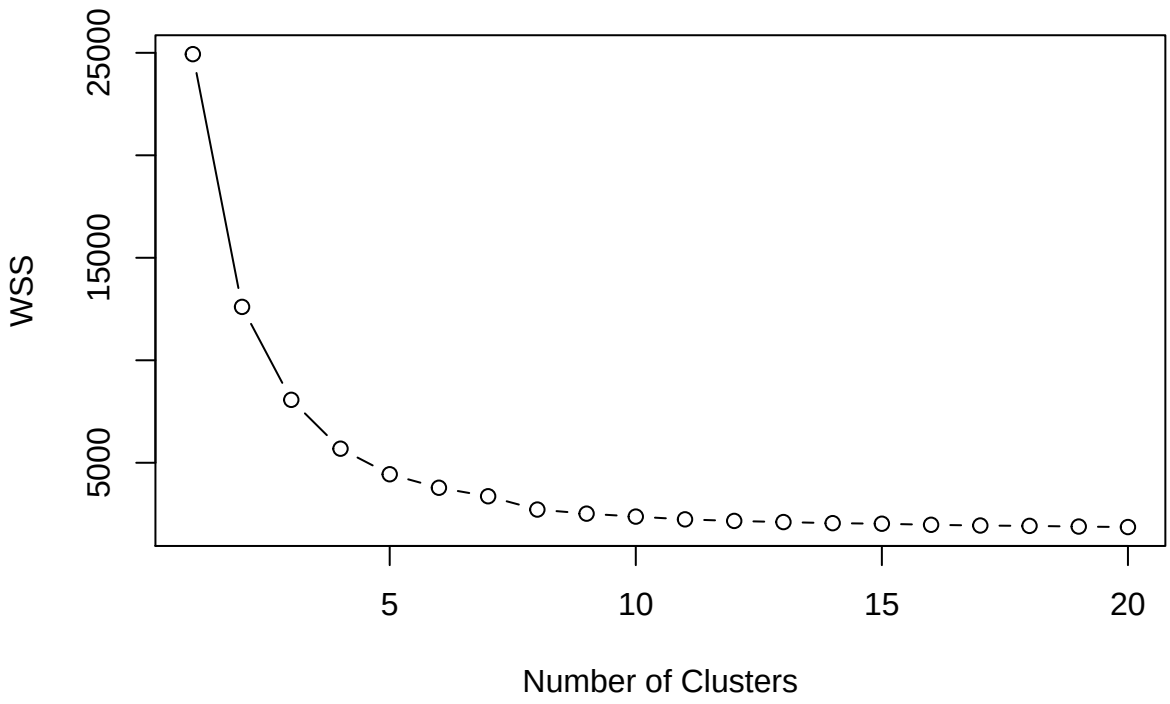
```
clean_madrid_data_p30 <- daily_energy_madrid_clus2 %>%
  filter (SEASON == "winter") %>%
```

```
  filter (PRODUCT == "P30") %>%
  group_by(CLIENT_ID, DOW) %>%
  summarise(avg_hourly_season = mean(AVG_CONS_PER_HR), std = sd(AVG_CONS_PER_HR))
n_distinct(clean_madrid_data_p30$CLIENT_ID)
```

```
## [1] 3563
```

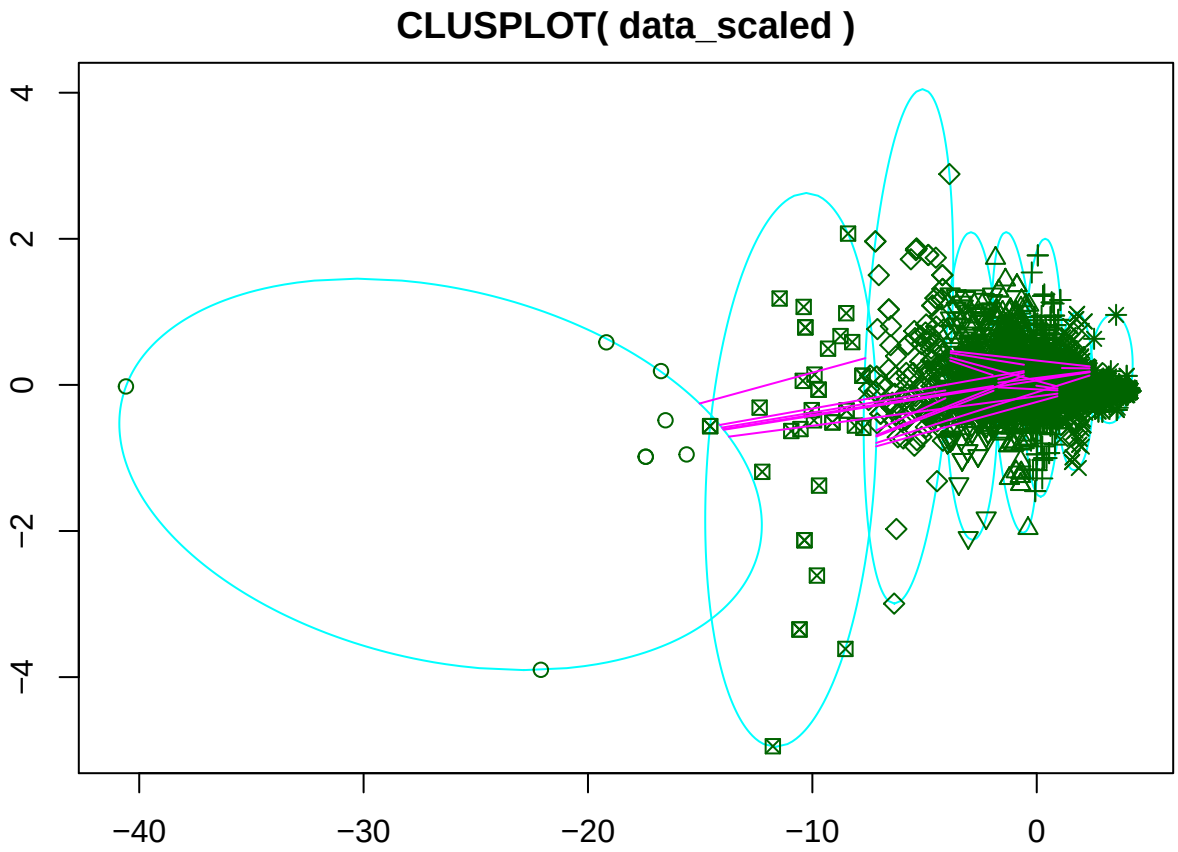
Lets scale our dataset and run the WSS function as we did previously.

```
clean_madrid_data_cluster <- clean_madrid_data_p30 %>%
  select(-std) %>%
  spread(DOW, avg_hourly_season)
data_scaled <- clean_madrid_data_cluster[,2:8]
data_scaled <- scale(data_scaled, center = TRUE)
#Run the WSS function
wssplot(data_scaled, 20)
```



Based on the WSS plot it seems that k=8 is the optimal number of clusters

```
fit.km_2 <- kmeans(data_scaled, centers = 8)
par(mar = rep(2, 4))
clusplot(data_scaled,fit.km_2$cluster)
```




```
#WSS (measure of compactness of clusters -> minimize)
fit.km_2$tot.withinss
```

```
## [1] 2716.944
```

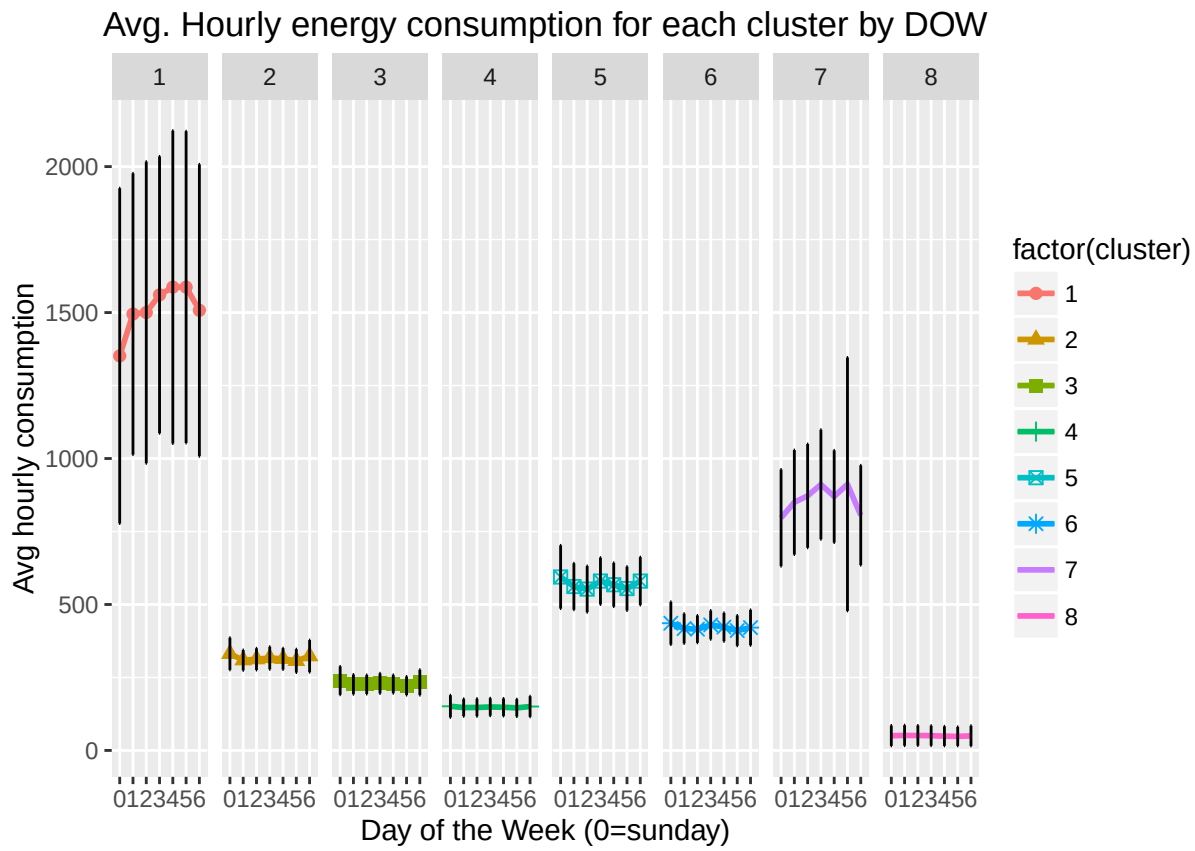
```
#BSS (measure of the distance between clusters -> maximize)
fit.km_2$betweenss
```

```
## [1] 22217.06
```

Lets visualize our clusters in the same manner as we did in the previous section:

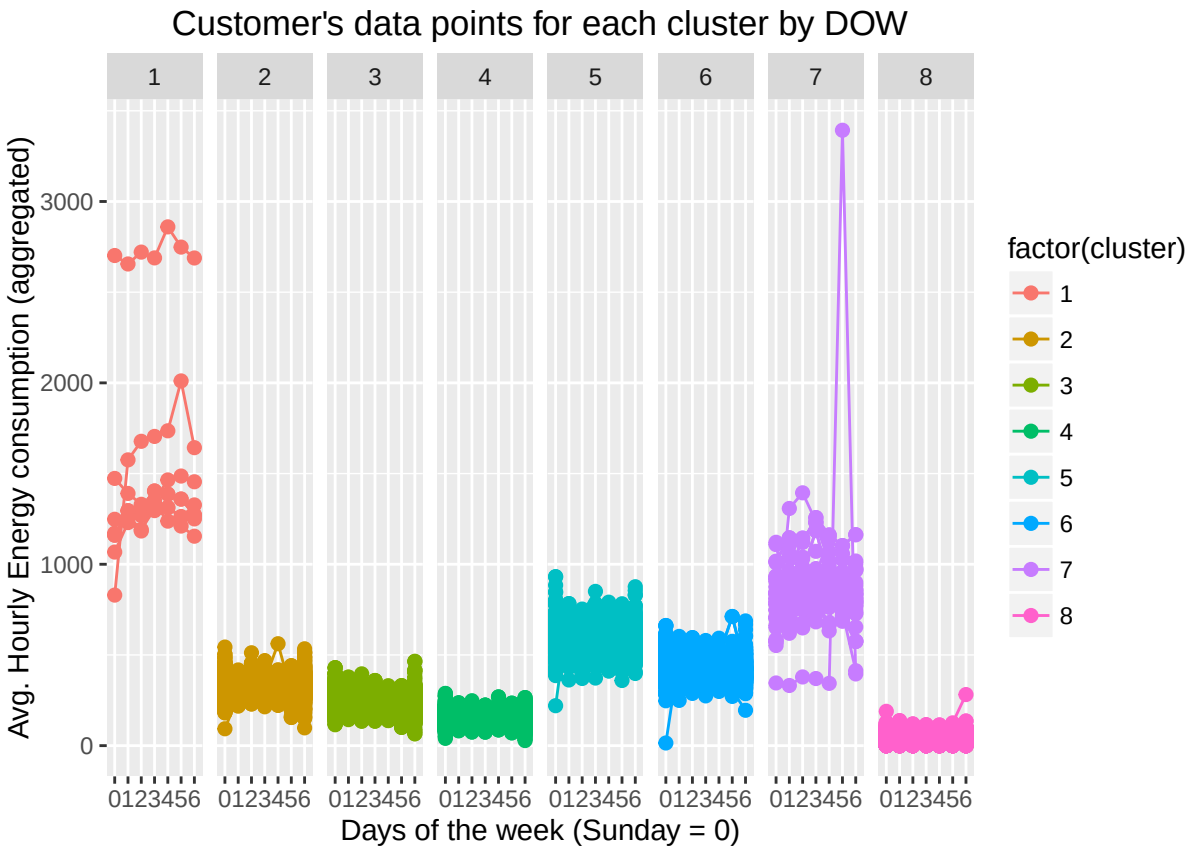
```
#Append cluster column to our original dataset
clean_madrid_data_cluster$cluster <- fit.km_2$cluster

#Aggregate by clusters and day of the week
clean_madrid_data_results <- clean_madrid_data_cluster %>%
  gather(key = "DOW", value = "avg_hourly_season",2:8) %>%
  group_by(DOW, cluster) %>%
  summarise(dow_avg_consumption = mean(avg_hourly_season), standDev = sd(avg_hourly_season))

#Visualizing the average hourly consumption per day of the week for each of the clusters
clean_madrid_data_results %>%
  ggplot(., aes(x = DOW, y = dow_avg_consumption, group = cluster, shape=factor(cluster), colour=factor
```

```
  ggtitle("Clusters") + scale_linetype_discrete(name="x") + facet_grid( . ~ cluster) +
  geom_errorbar(aes(x = DOW, ymin = dow_avg_consumption-standDev, ymax = dow_avg_consumption+standDev, 
                width=.1, position=position_dodge(width = .1)) +
  ggtitle("Avg. Hourly energy consumption for each cluster by DOW")
```



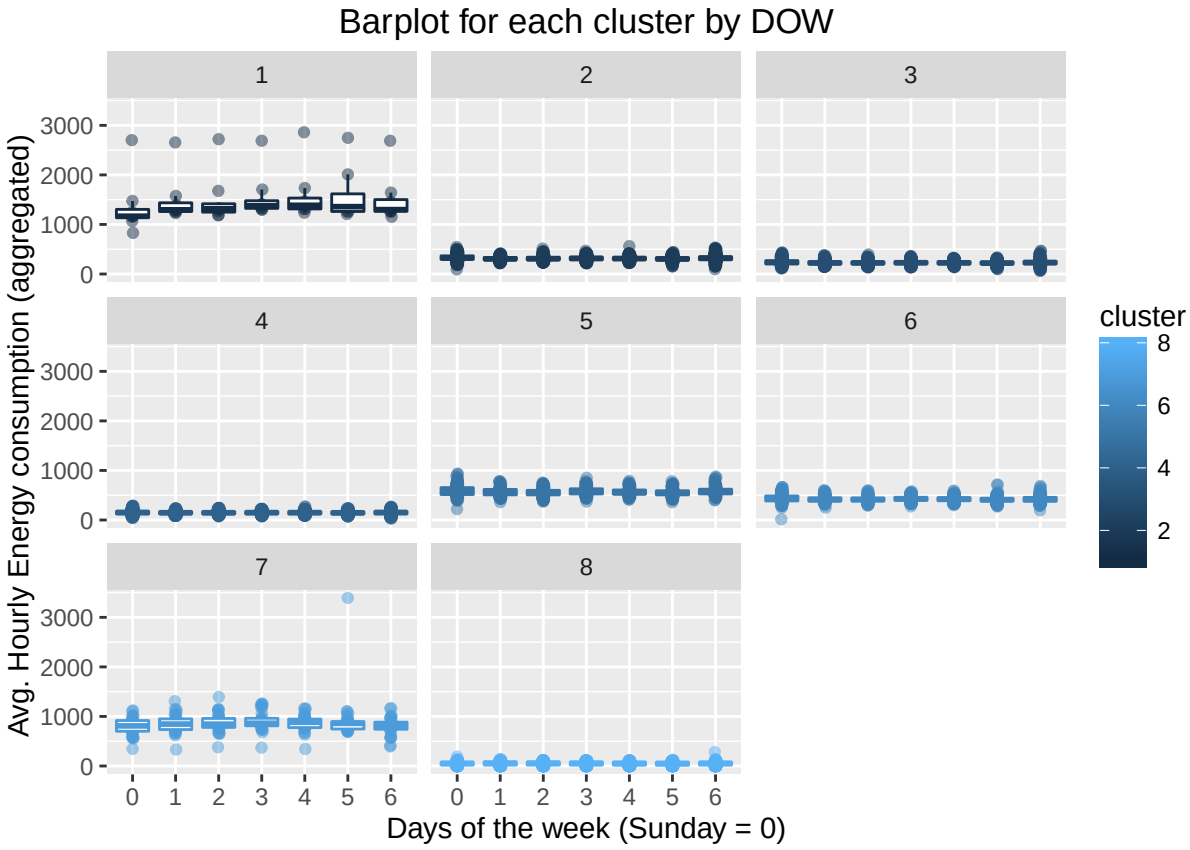
Shown below is the data from each customer for each clusters and day of the week (not aggregated):

```
clean_madrid_data_cluster %>%
  gather(key = "DOW", value = "avg_hourly_season",2:8) %>%
  ggplot(., aes(x = DOW, y = avg_hourly_season, group = CLIENT_ID, colour=factor(cluster))) + geom_line
```



Shown below is the data from each customer for each clusters and day of the week using a barplot to better understand the distribution of values per cluster

```
clean_madrid_data_cluster %>%
  gather(key = "DOW", value = "avg_hourly_season",2:8) %>%
  ggplot(.,aes(x=DOW , y=avg_hourly_season)) +
  scale_x_discrete() +
  geom_jitter(aes(colour = cluster, x = DOW),
              position = position_jitter(width = .05), alpha = 0.5) +
  geom_boxplot(aes(colour = cluster), outlier.colour = NA, position = "dodge") +
  facet_wrap(~ cluster) + labs(title = "Barplot for each cluster by DOW", x = "Days of the week (Sunday
```



Conclusion from Clustering analysis

- For most of the clusters, the Std. Dev. per DOW (error bars in the first plot) look significantly smaller than in the previous clustering analysis, which indicates that in this case customers within clusters share a higher similarity in terms of the usage pattern and level.
- There are couple of clusters that have very high Std. Devs per DOW. In the second plot we see that there are very fewer samples in these clusters and that customers within them exhibit a very different behavior which implies high Std. Dev.
- Based on both this clustering analysis and the previous one, it seems that most customer show a very consistent and flat pattern in terms of energy usage throughout the week. In other words, most of the customers spend the same level of energy every day of the week. However, on the next clustering analysis we will try to prove this hypothesis approaching the clustering from a different approach

K-means clustering 3: Clustering based on % energy consumption per day of the week

Dataset details:

- CITY: Madrid
- Timeframe: Jan-Mar 2015
- Products: P30
- Total Number of customers: 3316 (after removing outlier)

The main goal of this clustering model is to group together customers that exhibit very similar spending pattern and determine if there are groups of customers that spend energy during the week in a non-regular

pattern. For example we would like to be able to group together customers that spend significant amount of energy on the weekends but low during the week. This cluster would include then customers that tend to spend a lot of time at home during the weekend. With this kind of information available, Utilities companies can offer products specifically adapted to the spending pattern of each of their individual customers

Now we will run k-means on the same dataset but with a slight transformation: it will just contain the percentage of their total weekly energy consumption that is spend for each day of the week (e.g. customer A spends on Tuesday 20% of their weekly energy). This approach ensures that our clustering model is grouping customers just based on the weekly spending pattern (e.g customer spend most of the energy on the weekends) instead of also taking into consideration the quantity of energy use (e.g customer spends XX kWh on Tuesday)

We will be using the same dataset as in the previous section (customers in P30) but will need to do a bit of manipulation to calculate for each customer and DOW (day of the week) the percentage usage of energy out of the total week (averaged across the 3 months of data)

Lets group customers by week of the year (e.g. Jan 1st through 7th will be week 1 of the year)

```
n_distinct(daily_energy_madrid_clus2$CLIENT_ID)
```

```
## [1] 5627
```

```
by_week <- daily_energy_madrid_clus2 %>%
  filter(SEASON == "winter") %>%
  filter (PRODUCT == "P30") %>%
  group_by(CLIENT_ID, week_in_year) %>%
  summarise(energy_week = sum(TOTAL_DAY_CONS))

n_distinct(by_week$CLIENT_ID)
```

```
## [1] 3563
```

There are some customers that have certain weeks with 0 total energy consumption.These can be considered as outliers, therefore, since we want to model the distribution of energy consumption across a week, I believe it is safer to get rid of these customers. They might add unnecesary noise to the clustering algorithm.

```
customer_no_week_consump <-by_week %>%
  filter(energy_week <= 0)

by_week <- by_week %>%
  filter(!CLIENT_ID %in% c(customer_no_week_consump$CLIENT_ID))

n_distinct(by_week$CLIENT_ID)
```

```
## [1] 3466
```

Now lets calculate for each combination of DAY and customer the percentage energy usage out of the total correponding week

```
by_dow <- daily_energy_madrid_clus2 %>%
  filter(SEASON == "winter") %>%
  filter (PRODUCT == "P30") %>%
  select(CLIENT_ID, week_in_year, DOW,TOTAL_DAY_CONS)
```

```
by_dow <- by_dow %>%
  filter(!CLIENT_ID %in% c(customer_no_week_consump$CLIENT_ID))

by_dow_joined <- left_join(by_dow, by_week, by = c("CLIENT_ID" = "CLIENT_ID", "week_in_year" = "week_in

by_dow_joined_perc <- by_dow_joined %>%
  mutate(usage_perc = TOTAL_DAY_CONS/energy_week)

by_dow_joined_perc <- by_dow_joined_perc %>%
  select(CLIENT_ID, DOW, week_in_year, usage_perc)
```

For each customer and each day of the week (Sunday through Saturday), lets calcualte the average across the three months of data of the percentage of energy usage out of the whole week. Therefore, the resulting dataset will contain the following info:

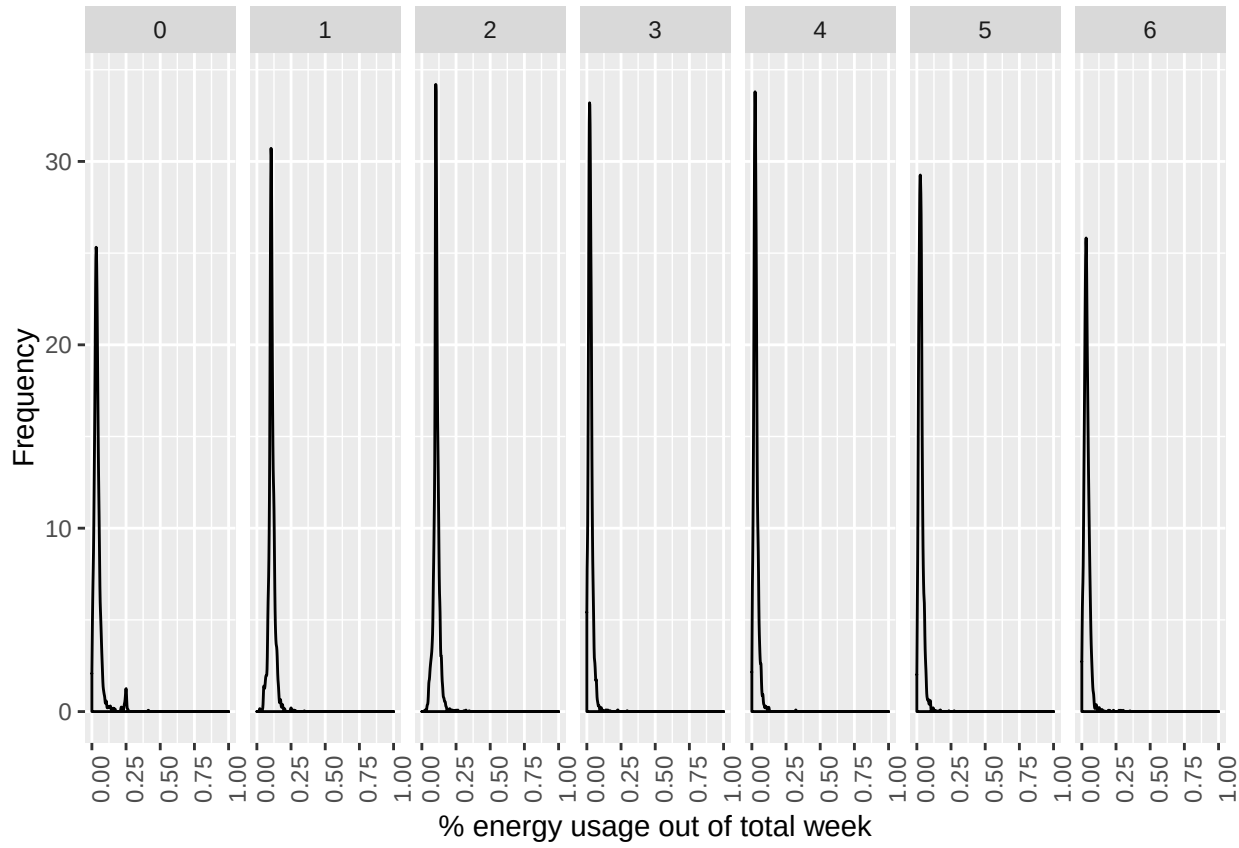
- Customer A - Monday - average of 20% energy spent out of total week
- Customer A - Tuesday - average of 15% energy spent out of total week (here the average for a given customer is calculated across the three months of data)

```
cluster_data <- by_dow_joined_perc %>%
  group_by(CLIENT_ID, DOW) %>%
  summarise(avg_usage_perc = mean(usage_perc), std = sd(usage_perc))
head(cluster_data)
```

```
## Source: local data frame [6 x 4]
## Groups: CLIENT_ID [1]
##
##   CLIENT_ID   DOW avg_usage_perc        std
##       (int) (int)          (dbl)      (dbl)
## 1     21457     0      0.1725288 0.04475844
## 2     21457     1      0.2149253 0.12528139
## 3     21457     2      0.1592245 0.08146150
## 4     21457     3      0.1469266 0.03327402
## 5     21457     4      0.1150656 0.02402825
## 6     21457     5      0.1003592 0.02619437
```

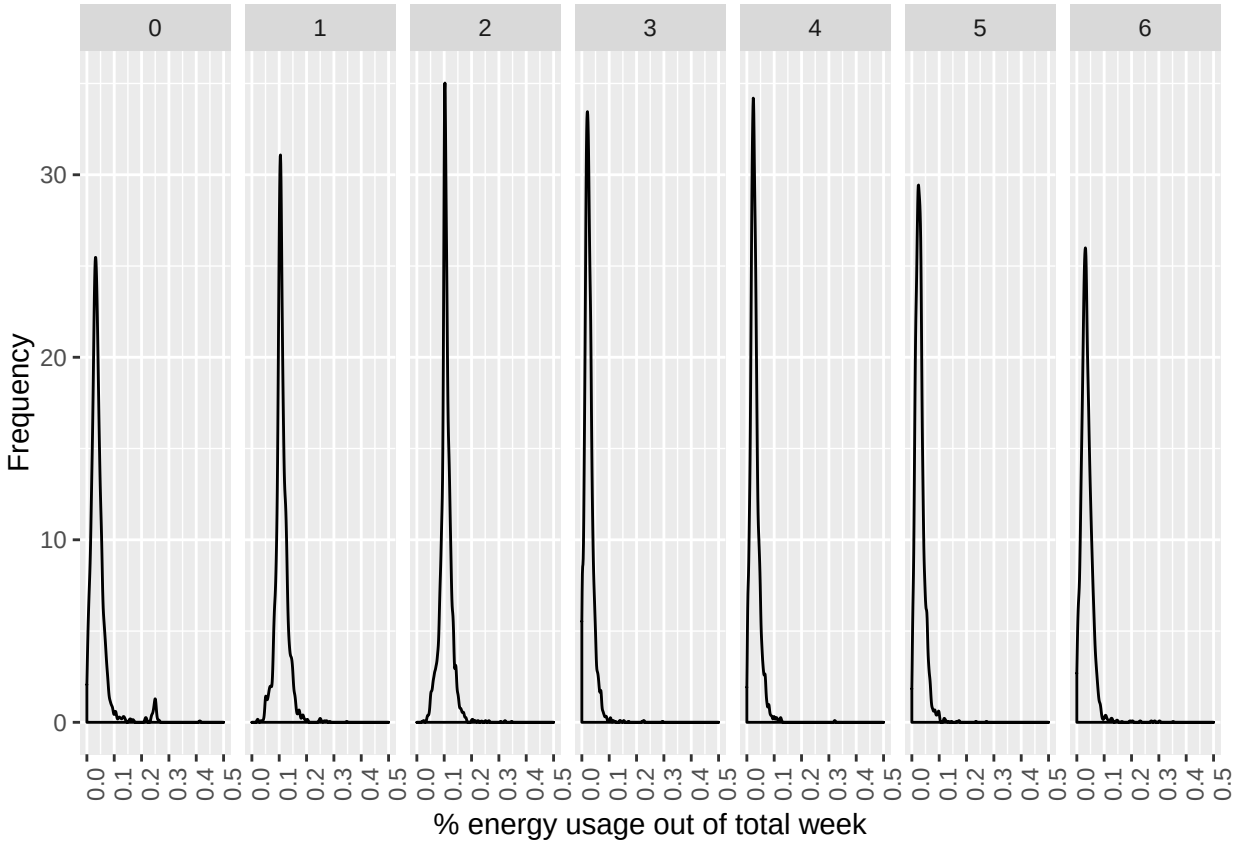
Now for each client, lets check what is the average percentage consumption by day of the week. We might need to remove some outliers that exhibit high standard deviations

```
ggplot(cluster_data, aes(x=std)) + geom_density(alpha=.3) + xlim(c(0,1)) + facet_grid (.~DOW) +
  xlab ("% energy usage out of total week") + ylab("Frequency") + theme(axis.text.x = element_text(angl
```



Zooming in to determine a reasanable cut-off

```
ggplot(cluster_data, aes(x=std)) + geom_density(alpha=.3) + xlim(c(0,0.5)) + facet_grid (.~DOW) +
    xlab ("% energy usage out of total week") + ylab("Frequency") + theme(axis.text.x = element_text(ang
```



By looking at the frequency plot above, it seems that most of the customers have less that 0.1 of Std. Dev. Therefore, to avoid having too many outliers to be input into our clustering algorithm, lets get rid of all those clients that have more than 0.15 of standard deviation per day of the week.

Lets identify which customers have at least one of their DOWs with Std. Dev. higher than 0.15.

```
outliers <- cluster_data %>%
  filter(std > 0.17)
n_distinct(outliers$CLIENT_ID)
```

```
## [1] 150
```

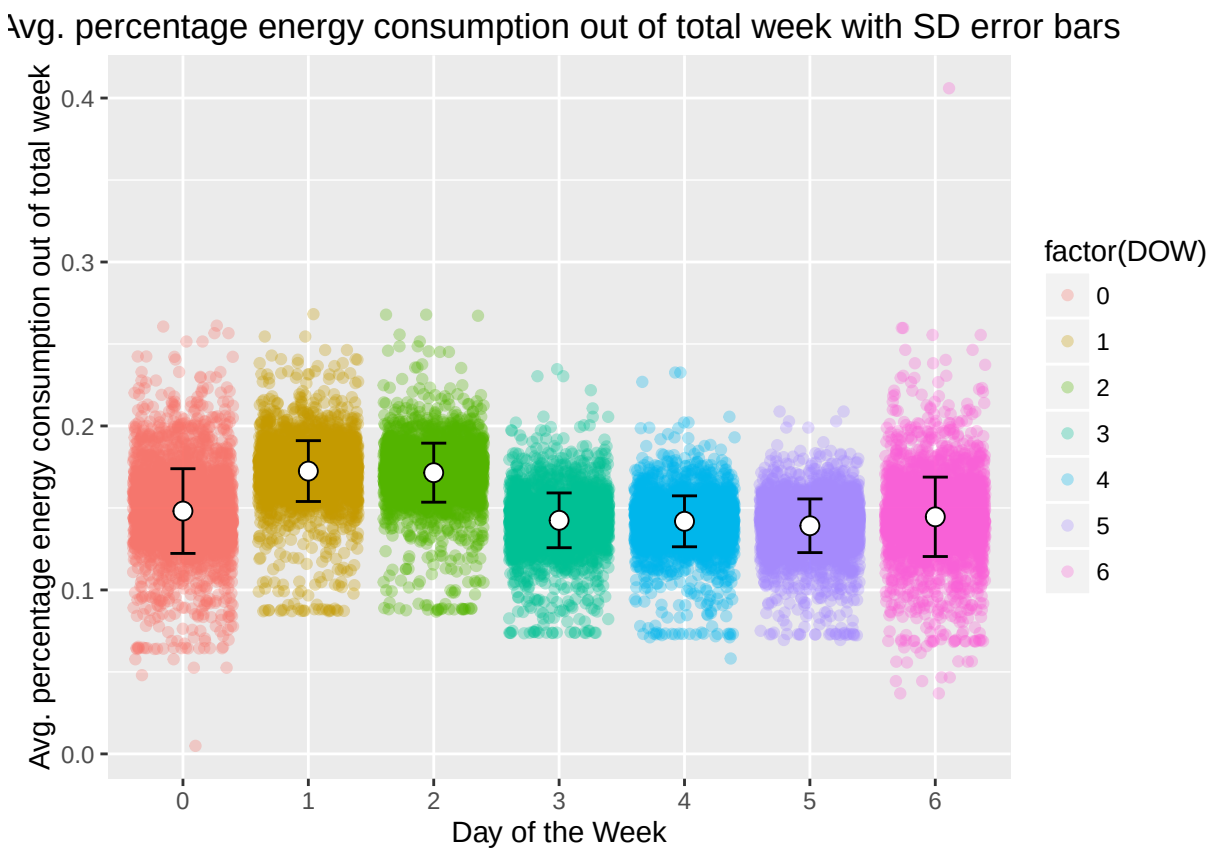
We are getting rid of 150 customers with at least the average of one of the days of the weeks with higher than 0.15 Std. Dev. So, lets remove these customers from our dataset and our resulting dataset will then contain 3316 customers.

```
cluster_data_clean <- cluster_data %>%
  filter (!CLIENT_ID %in% c(outliers$CLIENT_ID))
n_distinct(cluster_data_clean$CLIENT_ID)
```

```
## [1] 3316
```

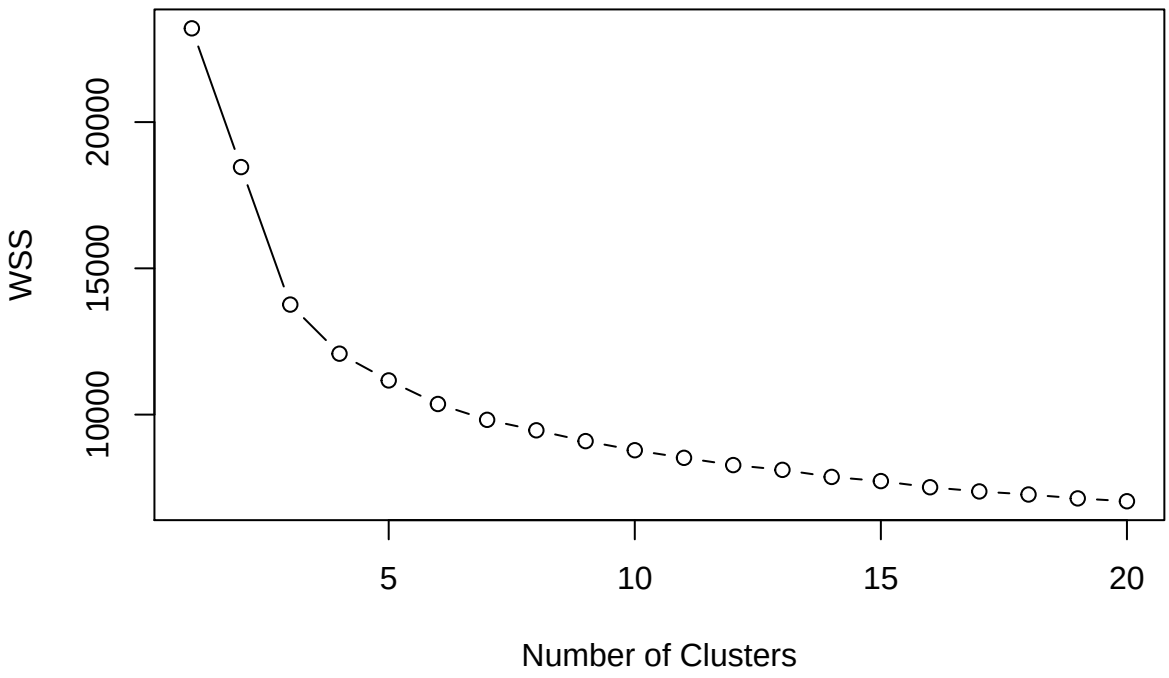
Before moving to the actual clustering algorithm, lets visually inspect an aggregated view of how all customers in our dataset on average behave for each day of the week. We see that customers in Madrid (for product 30) tend to spend more energy on Mondays and Tuesdays. On these two days respectively, customers on average consume ~17% of the total energy spent in the week.

```
cluster_data_clean%>%
  group_by(DOW) %>%
  summarise(perc_electr_usage = mean(avg_usage_perc), std = sd(avg_usage_perc)) %>%
  ggplot(data=., aes(x = factor(DOW), y = perc_electr_usage)) +
  geom_jitter(data = cluster_data_clean, mapping=aes(x = factor(DOW), y = avg_usage_perc, colour = facto
  geom_errorbar(aes(ymin=perc_electr_usage-std, ymax=perc_electr_usage+std),
                width = .2,                    # Width of the error bars
                position = position_dodge(.9)) +
  geom_point(shape = 21, size = 3, fill = "white") +
  scale_fill_discrete(guide=guide_legend(title = "Day of the Week")) +
  xlab("Day of the Week") +
  ylab("Avg. percentage energy consumption out of total week") +
  ggtitle("Avg. percentage energy consumption out of total week with SD error bars")
```



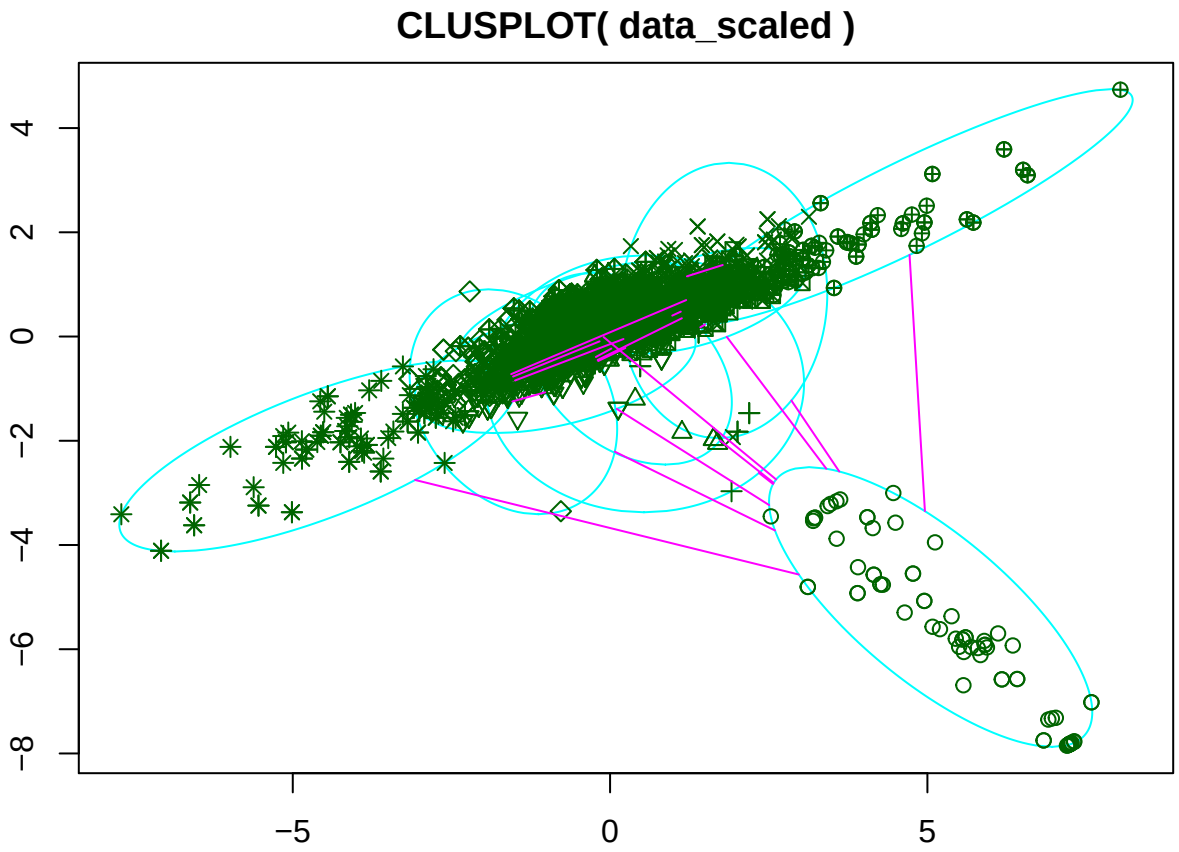
Lets scale our dataset and run the WSS function as we did previously.

```
clean_madrid_data_cluster <- cluster_data_clean %>%
  select(-std) %>%
  spread(DOW, avg_usage_perc)
data_scaled <- clean_madrid_data_cluster[,2:8]
data_scaled <- scale(data_scaled, center = TRUE)
#Run the WSS function
wssplot(data_scaled, 20)
```



Based on the WSS plot it seems that k~10 is the optimal number of clusters

```
fit.km_3 <- kmeans(data_scaled, centers = 10)
par(mar = rep(2, 4))
clusplot(data_scaled,fit.km_3$cluster)
```




```
#WSS (measure of compactness of clusters -> minimize)
fit.km_3$tot.withinss
```

```
## [1] 8781.755
```

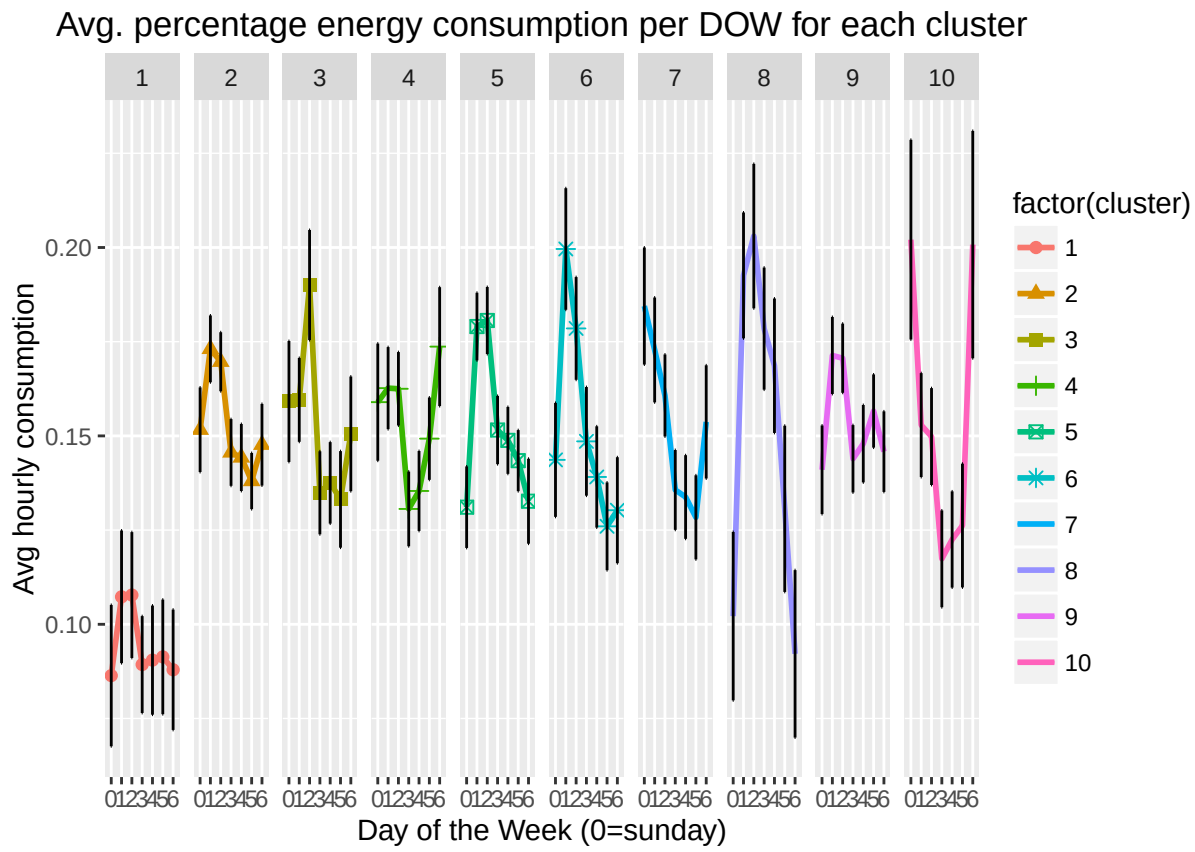
```
#BSS (measure of the distance between clusters -> maximize)
fit.km_3$betweenss
```

```
## [1] 14423.24
```

Lets visualize our clusters in the same manner as we did in the previous section:

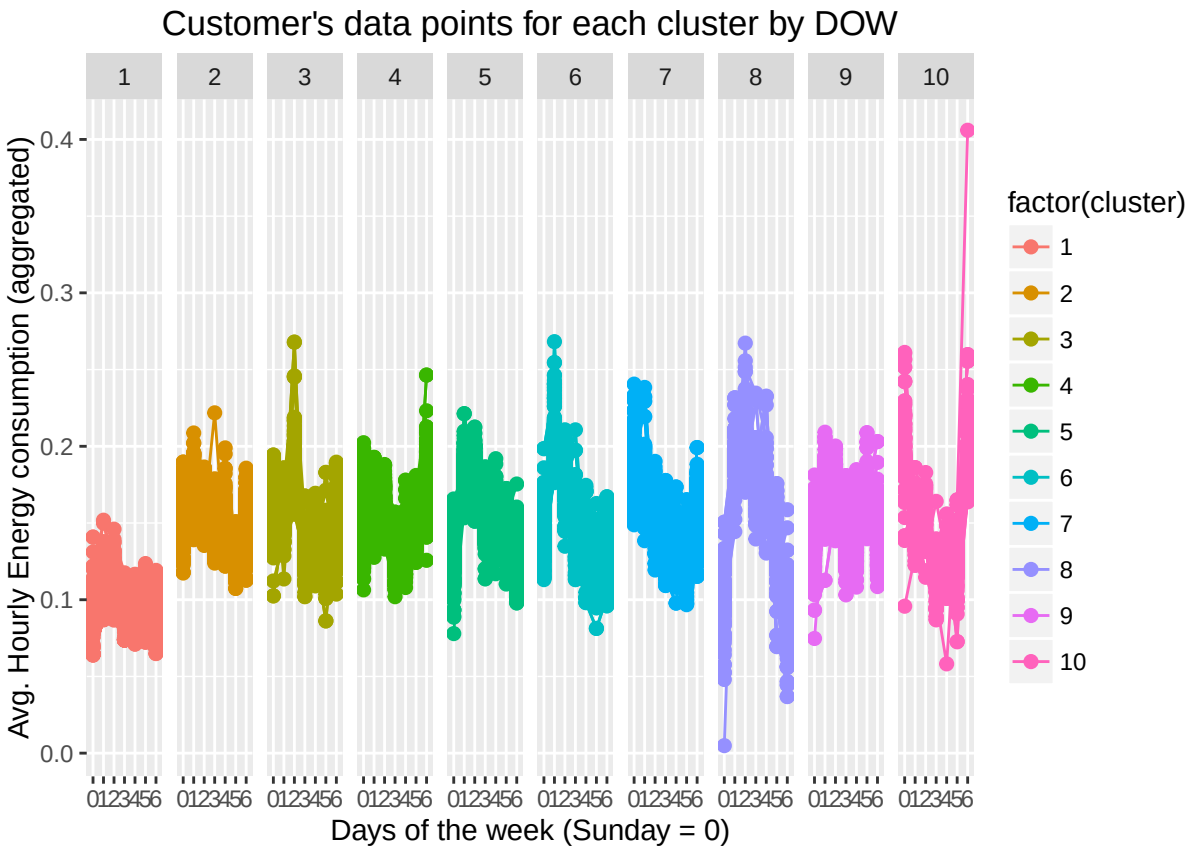
```
#Append cluster column to our original dataset
clean_madrid_data_cluster$cluster <- fit.km_3$cluster

#Aggregate by clusters and day of the week
results <- clean_madrid_data_cluster %>%
  gather(key = "DOW", value = "avg_weekday_perc_season",2:8) %>%
  group_by(cluster, DOW ) %>%
  summarise(avg_weekday = mean(avg_weekday_perc_season), standDev = sd(avg_weekday_perc_season))

#Visualizing the average hourly consumption per day of the week for each of the clusters
results %>%
  ggplot(., aes(x = DOW, y = avg_weekday, group = cluster, shape=factor(cluster), colour=factor(cluster)
```

```
  ggtitle("Clusters") +  scale_linetype_discrete(name="x") + facet_grid( . ~ cluster) +
  geom_errorbar(aes(x = DOW, ymin = avg_weekday-standDev, ymax = avg_weekday+standDev, color = NULL, li
```



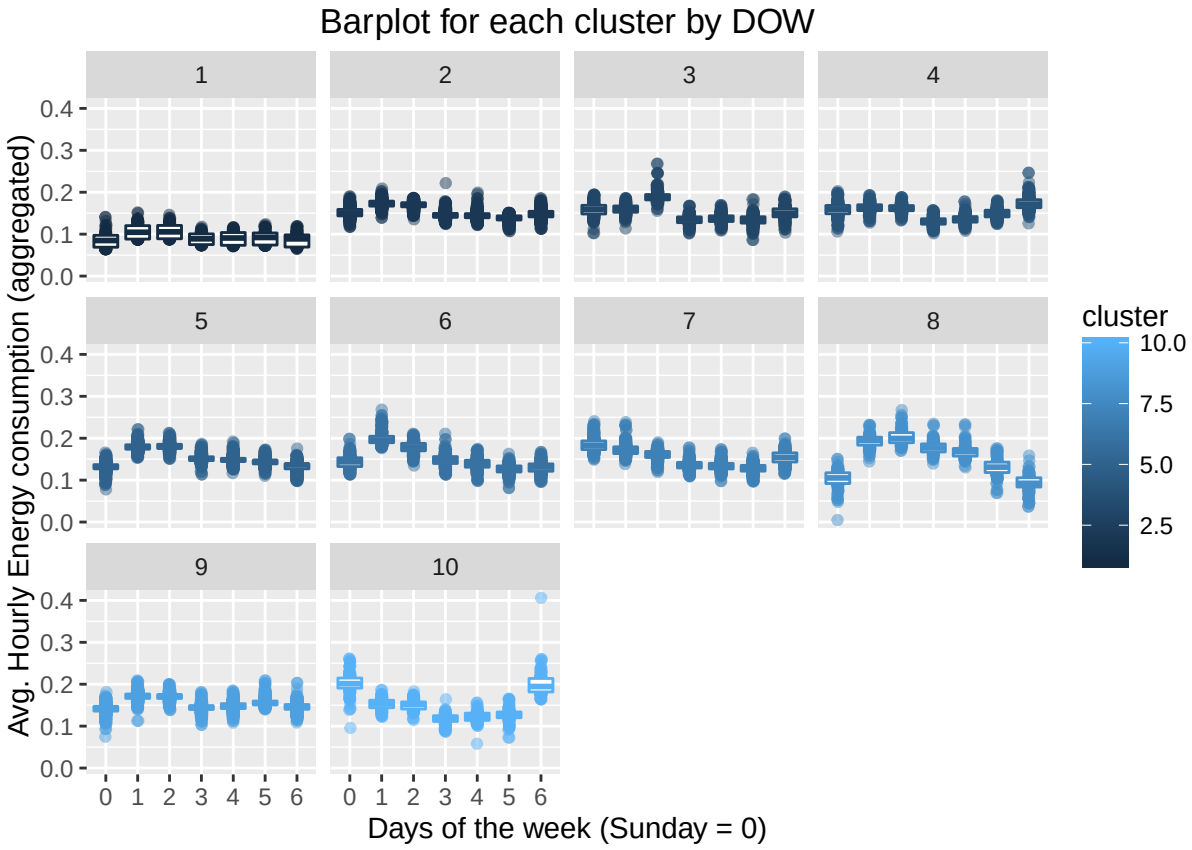
Shown below is the data from each customer for each clusters and day of the week (not aggregated):

```
clean_madrid_data_cluster %>%
  gather(key = "DOW", value = "avg_weekday_perc_season",2:8) %>%
  ggplot(., aes(x = DOW, y = avg_weekday_perc_season, group = CLIENT_ID, colour=factor(cluster))) + geo
```



Shown below is the data from each customer for each clusters and day of the week using a barplot to better understand the distribution of values per cluster

```
clean_madrid_data_cluster %>%
  gather(key = "DOW", value = "avg_weekday_perc_season",2:8) %>%
  ggplot(.,aes(x=DOW , y=avg_weekday_perc_season)) +
  scale_x_discrete() +
  geom_jitter(aes(colour = cluster, x = DOW),
              position = position_jitter(width = .05), alpha = 0.5) +
  geom_boxplot(aes(colour = cluster), outlier.colour = NA, position = "dodge") +
  facet_wrap(~ cluster) +
  labs(title = "Barplot for each cluster by DOW", x = "Days of the week (Sunday = 0)", y = "Avg. Hourly
```



Conclusion from Clustering analysis Using the average percentage energy consumption per DOW (instead of abs. value of energy consumed) has helped us achieve some results that a priori look promising. The k-means clustering algorithm run on this transformed dataset seems to have better grouped together customers that on average exhibit a similar energy usage pattern across the week.

- For more than half of the clusters obtained, the error bars (std dev per DOW for each cluster) look moderately low. This means we have a reasonable low variability of % energy consumption per DOW within each cluster and therefore we could safely assume that most of the clusters include customers that exhibit a similar weekly routine.
- From the three plots above, we can **easily differentiate the weekly energy usage patterns** for each clusters: we see several clusters that tend to spend more on Mondays and Tuesdays and less during the weekends. In a couple clusters (with a bit more inter-cluster variability), customers seems to spend most of their energy during the weekend.
- One interesting point to note that, on average very few customers tend to spend on a single day no more than 20% of their total weekly energy consumption

Next steps: The results of this last clustering analysis seem to confirm that using k-means along with a properly prepared dataset, we are able to group together customers that exhibit similar weekly routines. The next-steps (but out of the scope of this project) would be to use the full data set (~3 years worth of data) and feed it to k-means and possibly other clusterings algorithm (i.e. cobweb cluster, etc.) to get a more accurate understanding of usage patterns across the whole customer base.

Deep-dive Madrid: 5. Customers away from home

Dataset details:

- CITY: Madrid
- Timeframe: Jan-Mar 2015
- Products: P30
- Total Number of customers: 5627

In this section, we will leverage a simple but effective way to **identify for which days during the given period a customer was not at home**. We will do so by comparing the energy consumption of a particular day with their usual usage level. We will calculate for each customer, their average energy consumption per day of the week for the period selected (Jan, Feb, March) and then run through the whole dataset (day by day) to determine for which days the energy consumption is significantly lower than their average.

For this analysis, we will use the same dataset that we created for the last clustering algorithm. As we can see above, this dataset includes customers from Madrid that have contracted product P30. After some data cleansing performed in the previous section, we end up with 5627 customers. The aggregated data frame below shows the average (and std. dev.) total day energy consumption per customer for each day of the week during the selected period.

```
n_distinct(daily_energy_madrid_clus2$CLIENT_ID)
```

```
## [1] 5627
```

```
clean_madrid_data_p30 <- daily_energy_madrid_clus2 %>%
  filter (SEASON == "winter") %>%
  filter (PRODUCT == "P30") %>%
  group_by(CLIENT_ID, DOW) %>%
  summarise(avg_DOW_consumption = mean(TOTAL_DAY_CONS), std = sd(TOTAL_DAY_CONS))
n_distinct(clean_madrid_data_p30$CLIENT_ID)
```

```
## [1] 3563
```

Lets create another dataframe with all the detailed daily data for each customer. We are including an additional column ("check-home") that we will use later to flag the days when customers seem to be away from home.

```
customer_in_scope <- daily_energy_madrid %>%
  filter(CLIENT_ID %in% c(unique(clean_madrid_data_p30$CLIENT_ID))) %>%
  filter(SEASON == "winter") %>%
  filter (PRODUCT == "P30") %>%
  mutate(check_home = "Y")
```

To our full daily dataset for the customers in scope ("customers_in_scope") lets append the columns that indicate the average and standard deviation energy consumption for the day of the week (Mon through Sun) in the winter season (Jan, Feb, Mar).

```
customer_in_scope_DOW <- left_join(customer_in_scope, clean_madrid_data_p30, by = c("CLIENT_ID" = "CLIE
```

We are going to consider that a customer is "away-from-home" if the energy consumed for a particular day is below 2 std.dev from the average energy consumed for that given day of the week across the 3 months. For example, if the energy that a particular customer spent on day 02-04-2015 (a Monday) is lower than 2 Std. Dev below his/her average consumption on Mondays for the last 3 months, then we will conclude that the customer was either away from home or at least spent a good portion of the day away. We will capture this info on the "check-home" flag column.

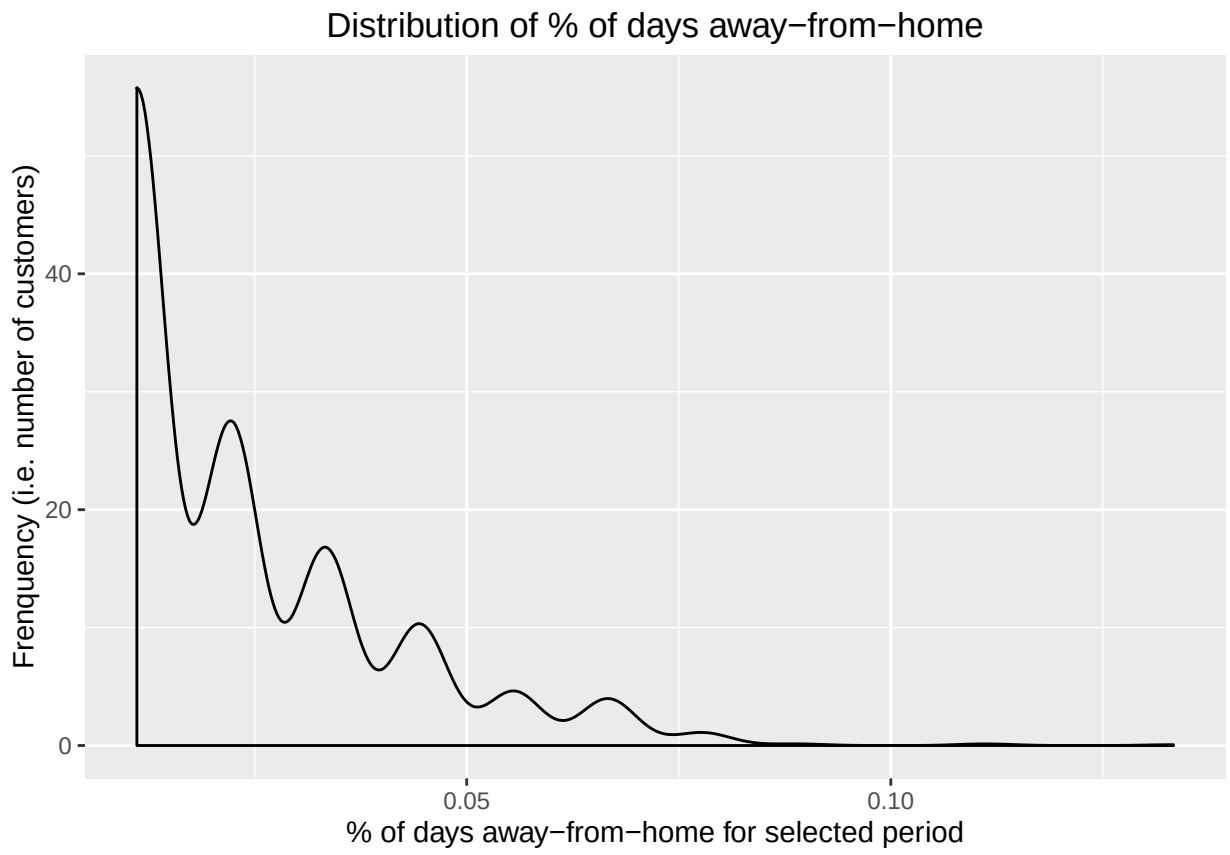
```r
#Determining which days are customers away from home
customer_in_scope_DOW <- customer_in_scope_DOW %>%
  mutate (check_home = ifelse(TOTAL_DAY_CONS < avg_DOW_consumption - std*2,"N", "Y"))
```

Using a frequency plot lets get a sense of the usual total number of days that customers were away from home during this period of time.

```r
customer_in_scope_DOW %>%
  group_by(check_home) %>%
  summarise(client_count= n_distinct(CLIENT_ID), day_count = n_distinct(DAY))
```

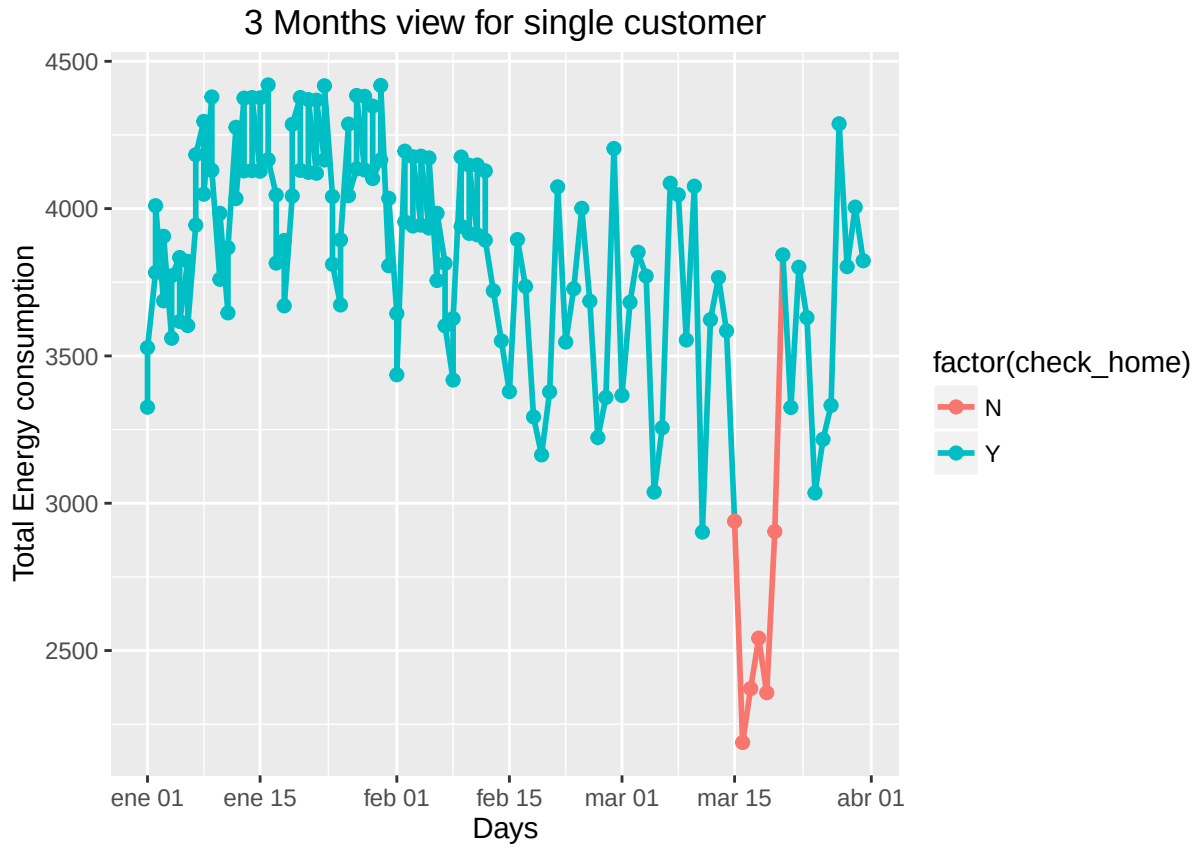
```
## Source: local data frame [2 x 3]
##
##   check_home client_count day_count
##        (chr)        (int)     (int)
## 1          N         1754        90
## 2          Y         3563        90
```

```r
summary_days_out <- customer_in_scope_DOW %>%
  group_by(CLIENT_ID, check_home) %>%
  summarise(day_count = n_distinct(DAY)) %>%
  filter(check_home == "N") %>%
  mutate (avg_days_out = day_count/90)

ggplot(summary_days_out, aes(x=avg_days_out)) + geom_density(alpha=.3) + xlab("% of days away-from-home
```



Below is it shown the daily energy consumption for a given customers. We can clearly see that the points in red (points that we flagged as away from home) show clearly that the customer those days was either away from home or away for a good portion of the day.

```
customer_in_scope_DOW %>%
  filter(CLIENT_ID == 24903) %>%
  ggplot(., aes(x = DAY, y = TOTAL_DAY_CONS, colour = factor(check_home)))+ geom_line(size=1, aes(group
```



Now, with the plot below we want to determine the variation of the number of customers away from home along the period in time that we are analyzing. The question here is whether we can identify specific periods of time when a significant number of customers were away from their homes due to, for example, vacations.

From the plot we can clearly see that indeed there is a period of time in these 3 months when a significant amount of customers seemed to be away from their homes. There is an unusual spike around the 27th of March of 2015 that keeps increasing for several days until the end of the dataset. Between the 27/03/2015 and 03/04/2015 in Spain it was the country-wide "Easter Break", which is a period of time when a significant number of people in Spain are off work due to public holidays.

- **Conclusion**: This simple but elegant method is able to accurately flag when customers are away from their home and the plot below is a clear proof that the results obtained are realistic.

```
customer_in_scope_DOW %>%
  group_by(DAY) %>%
  filter(check_home == "N") %>%
  summarise(client_count= n()) %>%
  mutate (perc_out = client_count/3574) %>%
```

```
ggplot(., aes(x = DAY, y = perc_out)) + geom_line(size=1, colour = I("skyblue2") ,aes(group = 1)) +
annotate("rect", xmin=as.Date("03/27/2015", "%m/%d/%Y"), xmax=as.Date("04/01/2015", "%m/%d/%Y"), ymin=
```