Exercise C

Table of contents

Exercise C

```
## loading all the R workspace data

load("C:/AppliedDataScienceAndStatistics/Applied-Data-Science-and-Statistics/Term1/Applica
load("C:/AppliedDataScienceAndStatistics/Applied-Data-Science-and-Statistics/Term1/Applica
load("C:/AppliedDataScienceAndStatistics/Applied-Data-Science-and-Statistics/Term1/Applica
load("C:/AppliedDataScienceAndStatistics/Applied-Data-Science-and-Statistics/Term1/Applica
dataCharacteristics = read_csv("C:/AppliedDataScienceAndStatistics/Applied-Data-Science-an
```

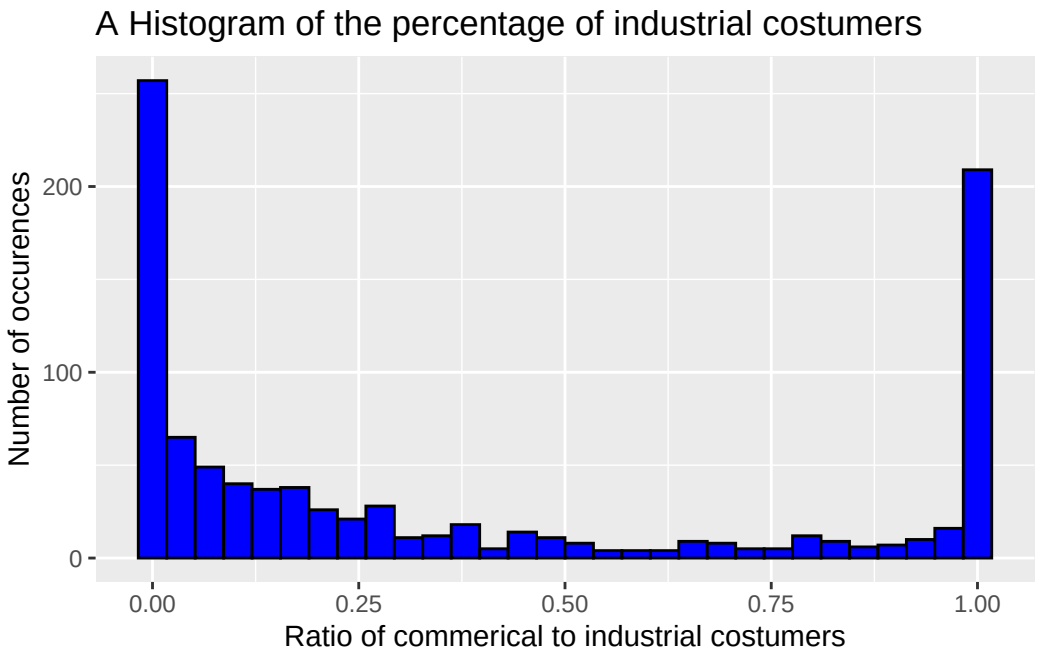
Exercise C 1.i

```
summary(dataCharacteristics)
```

```
SUBSTATION_NUMBER TRANSFORMER_TYPE   TOTAL_CUSTOMERS Transformer_RATING
Min.   :511016    Length:948         Min.   :  0.0   Min.   :   0.0
1st Qu.:521516    Class :character   1st Qu.:  3.0   1st Qu.: 200.0
Median :532653    Mode  :character   Median : 67.5   Median : 315.0
Mean   :534344                       Mean   :104.3   Mean   : 389.1
3rd Qu.:552386                       3rd Qu.:179.2   3rd Qu.: 500.0
Max.   :564512                       Max.   :569.0   Max.   :1000.0
Percentage_IC     LV_FEEDER_COUNT  GRID_REFERENCE
Min.   :0.00000   Min.   : 0.000   Length:948
1st Qu.:0.01048   1st Qu.: 1.000   Class :character
Median :0.17849   Median : 3.000   Mode  :character
Mean   :0.37982   Mean   : 2.762
3rd Qu.:0.90271   3rd Qu.: 4.000
Max.   :1.00000   Max.   :16.000
```

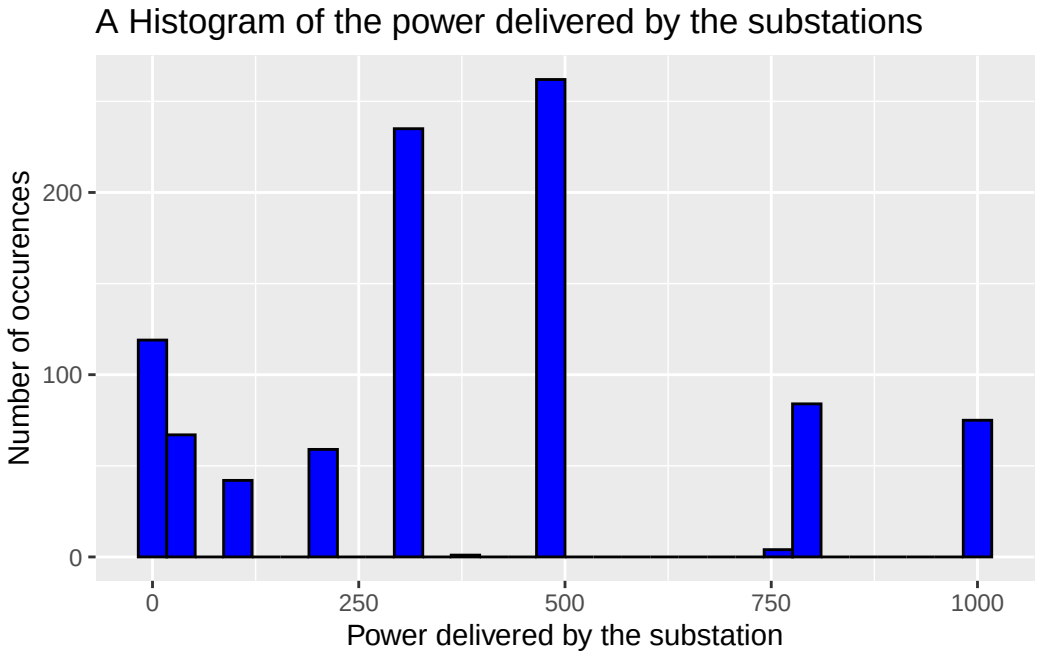
To visualise the distributions of the percentage of industrial and commercial costumer, transformer ratings and transformer types We will plot a histogram of each of these variables

```
ggplot(dataCharacteristics, aes(Percentage_IC)) + geom_histogram(bins = 30,
    fill = "blue", color = "black") + ggtitle("A Histogram of the percentage of industrial
    xlab("Ratio of commerical to industrial costumers") + ylab("Number of occurences")
```



The histogram here the number of occurrences cluster around fully commercial costumers and fully industrial costumers where both these values have a much higher number of occurrences than any other percentage ratio of costumers. As the number of industrial costumers increase, the number of occurrences slowly decreases until it reach it's lowest near the 50 percentile, barely increasing again around the 80 percentile until reaching another high peak on 100% industrial costumers, following what it seems to be a very steep u-shaped distribution also called u-quadratic distribution.

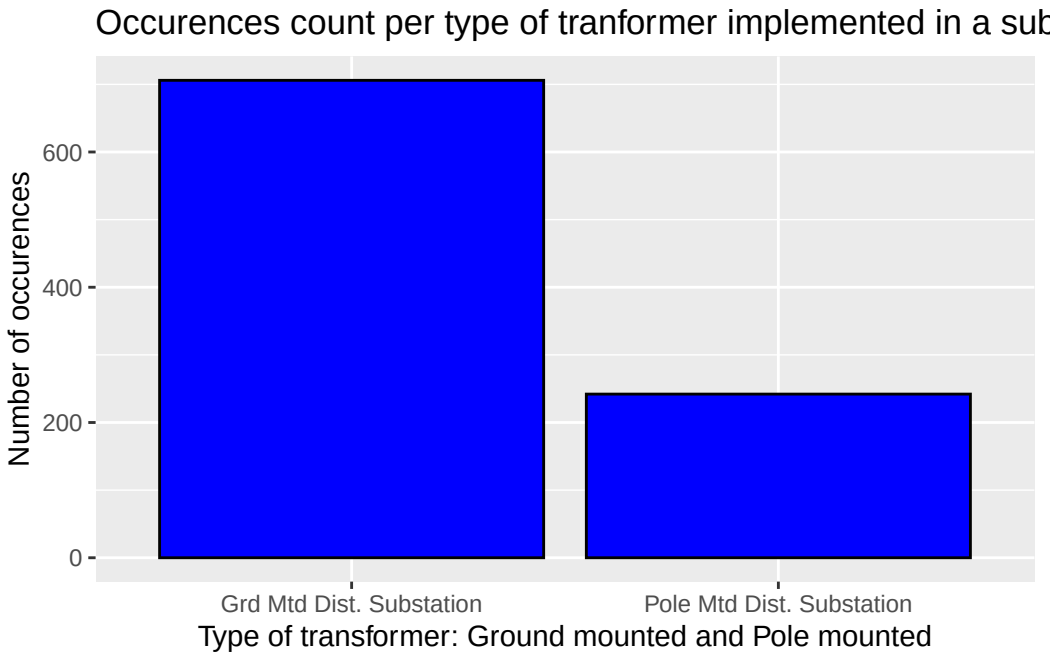
```
ggplot(dataCharacteristics, aes(Transformer_RATING)) + geom_histogram(bins = 30,
    fill = "blue", color = "black") + ggtitle("A Histogram of the power delivered by the s
    xlab("Power delivered by the substation") + ylab("Number of occurences")
```



As we can see in the histogram above the amount of power being delivered to the substations is the highest around 500 and second highest around 300. It is also evident that there is a large gap around amounts of power being delivered to the substation. We can also see that with the exception to 0 or near 0 power delivered and two other values that can possible be outliers the number of occurrences increase as we get closer to 500 power delivered by the substations, following what would resemble a normal distribution.

```
ggplot(dataCharacteristics, aes(TRANSFORMER_TYPE)) + geom_bar(fill = "blue",
    color = "black") + ggtitle("Occurences count per type of tranformer implemented in a s
```

```
    xlab("Type of transformer: Ground mounted and Pole mounted") + ylab("Number of occuren
```



As we can see the number of ground mounted substations is almost 3 times bigger than the pole mounted substations.

Exercise C 1.ii

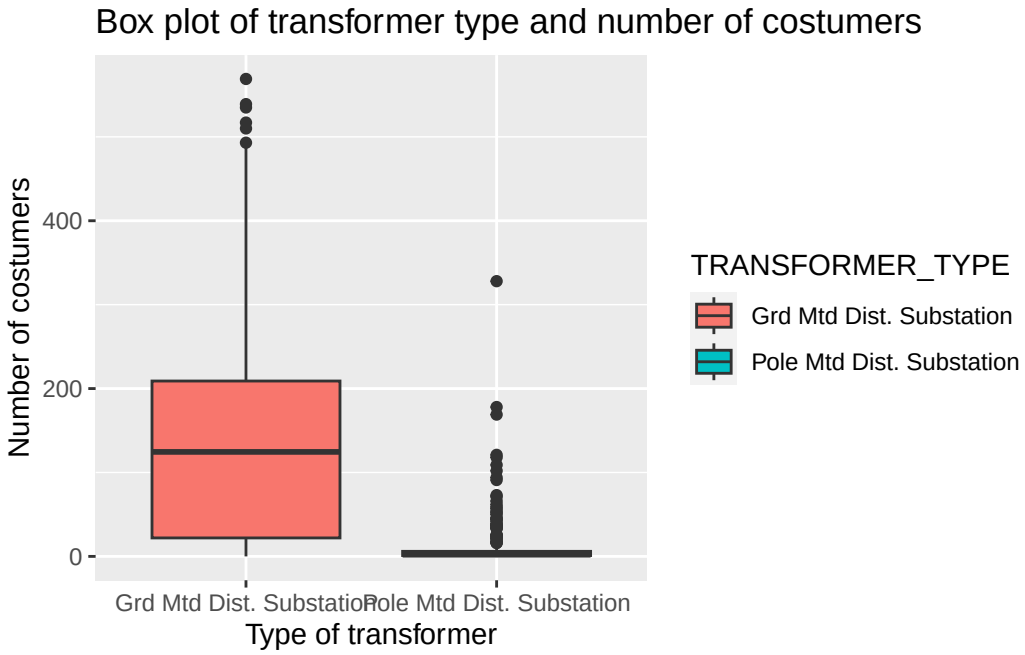
There are several ways to describe the relationships between different characteristics:

- Correlation: This measures the strength and direction of the relationship between two variables. A positive correlation means that as one variable increases, the other variable also increases. A negative correlation means that as one variable increases, the other variable decreases.
- Regression: This is a statistical method used to find the relationship between a dependent variable and one or more independent variables. It can be used to predict the value of the dependent variable based on the values of the independent variables.
- Clustering: This is a method used to group similar observations together based on their characteristics. Clustering can be used to identify patterns or relationships in the data.
- Dimensionality reduction: This is a technique used to reduce the number of variables in a dataset while preserving as much information as possible. This can be useful for visualizing and understanding relationships between variables.

- Visualization: This is a way of representing data graphically, such as with charts, plots, and maps. Visualization can help to identify patterns and relationships in the data.

We have already have resorted to visualization of the distributions on the previous question so now I will be visualising and correlation each of the variables against each other.

```
## variables to compare: transformer type, number of customers, rating,
## percentage of I&C customers and number of feeders these are:
## TRANSFORMER_TYPE, TOTAL_CUSTOMERS , Transformer_RATING ,
## Percentage_IC and LV_FEEDER_COUNT

## lets start with transformer type comparison against all other
## variable the first one will be comparing it with total customers

ggplot(data = dataCharacteristics, aes(x = TRANSFORMER_TYPE, y = TOTAL_CUSTOMERS,
    fill = TRANSFORMER_TYPE)) + geom_boxplot() + ggtitle("Box plot of transformer type and
    xlab("Type of transformer") + ylab("Number of costumers")
```



As we can clearly observe the type of transformer is a strong indicator of the number of costumers as pole mounted substations values from the minimum third quantile values equal or approximate to zero comparing to ground mounted substations, these already have a number of costumers superior to 0 on the first quartile and the entire interquartile range holds values

only present on the top quartile of the pole mounted substations.

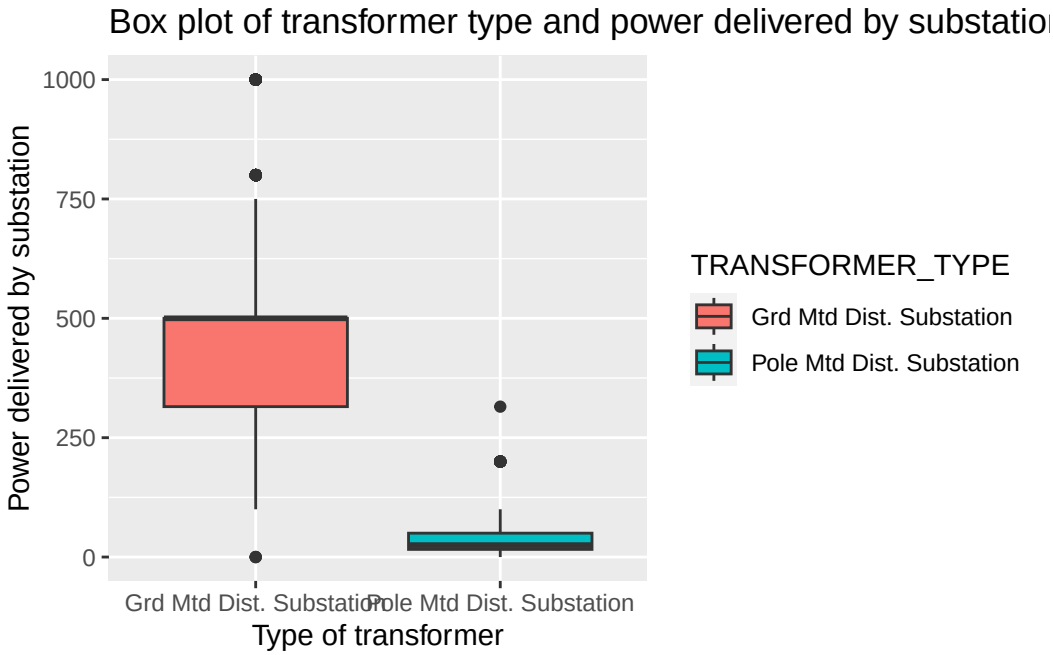
```
modelTransTypeNCostumer = lm(TOTAL_CUSTOMERS ~ TRANSFORMER_TYPE, data = dataCharacteristic

summary(modelTransTypeNCostumer)
```

```
Call:
lm(formula = TOTAL_CUSTOMERS ~ TRANSFORMER_TYPE, data = dataCharacteristics)

Residuals:
    Min      1Q  Median      3Q     Max
-135.78  -72.03  -10.63   49.22  433.22

Coefficients:
                                          Estimate Std. Error t value Pr(>|t|)
(Intercept)                                135.778      3.854   35.23   <2e-16
TRANSFORMER_TYPEPole Mtd Dist. Substation -123.150      7.629  -16.14   <2e-16

(Intercept)                               ***
TRANSFORMER_TYPEPole Mtd Dist. Substation ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 102.4 on 946 degrees of freedom
Multiple R-squared:  0.216, Adjusted R-squared:  0.2151
F-statistic: 260.6 on 1 and 946 DF,  p-value: < 2.2e-16
```

As we can see from the summary of the linear model, a substation being pole mounted largely decreases the number of costumers in such substation. It is also important to notice the the multiple R-squared and adjusted R-squared values as they signify that the type of transformer mounted only explains explains 21% of the variation in the number of costumers.

Now we will proceed to analyse the comparison of transformer type and transformer rating.

```
ggplot(data = dataCharacteristics, aes(x = TRANSFORMER_TYPE, y = Transformer_RATING,
    fill = TRANSFORMER_TYPE)) + geom_boxplot() + ggtitle("Box plot of transformer type and
    xlab("Type of transformer") + ylab("Power delivered by substation")
```



We can observe again that the type of transformer is a strong indicator of the power delivered by the substation, as with exception of 2 outliers the pole mounted entire range of power delivered is smaller then the first quartile of power delivered by ground mounted substations. As such it is clear how impactful are the type of transformer in the power delivered to the substations.

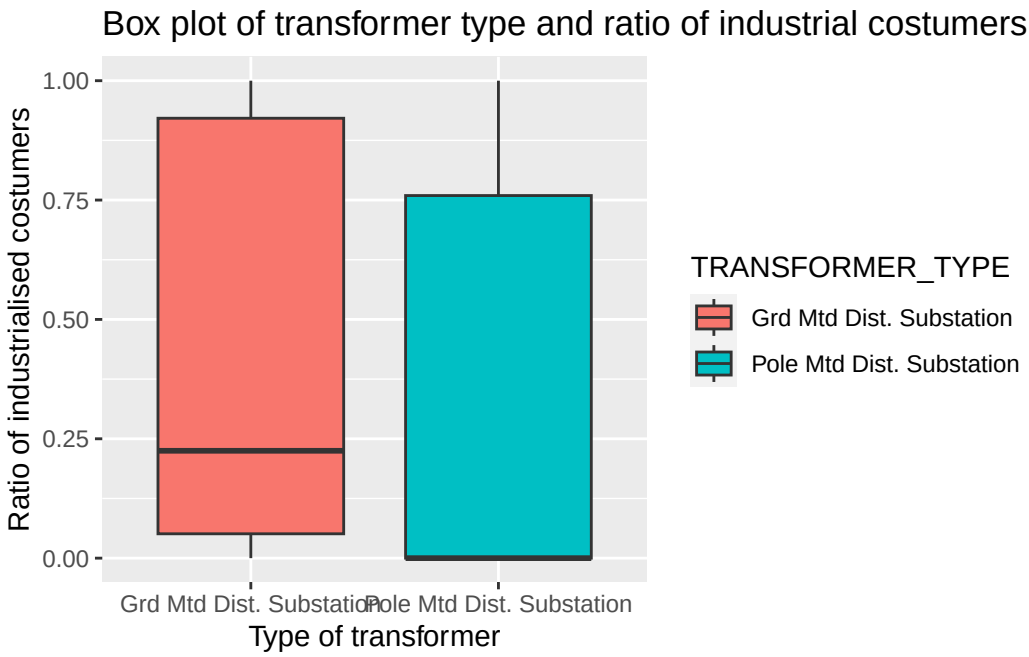
```
modelTransTypePowerDelivery = lm(Transformer_RATING ~ TRANSFORMER_TYPE, data = dataCharact

summary(modelTransTypePowerDelivery)
```

```
Call:
lm(formula = Transformer_RATING ~ TRANSFORMER_TYPE, data = dataCharacteristics)

Residuals:
    Min      1Q  Median      3Q     Max
-505.16 -190.16   -5.16   -0.60  494.84

Coefficients:
                                          Estimate Std. Error t value Pr(>|t|)
(Intercept)                                505.156      7.826   64.55   <2e-16
TRANSFORMER_TYPEPole Mtd Dist. Substation -454.555     15.490  -29.34   <2e-16
```

```
(Intercept)                                  ***
TRANSFORMER_TYPEPole Mtd Dist. Substation ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 208 on 946 degrees of freedom
Multiple R-squared:  0.4765,    Adjusted R-squared:  0.476
F-statistic: 861.1 on 1 and 946 DF,  p-value: < 2.2e-16
```

As we can see from the summary of the model the type of transformer has a massive impact determining the amount of energy supplied to the substation and as observed from the R-squared value the type of mounted substation can explain almost half of the variation in power supplied to the substations

```
ggplot(data = dataCharacteristics, aes(x = TRANSFORMER_TYPE, y = Percentage_IC,
    fill = TRANSFORMER_TYPE)) + geom_boxplot() + ggtitle("Box plot of transformer type and
    xlab("Type of transformer") + ylab("Ratio of industrialised costumers")
```



In the bar graph we can see that half of the pole mounted substation have only business costumers and therefore no industrial costumers. We can also observe that pole mounted substations have in general considerably bigger ratio of business costumers has more pole mounted substations 3rd quartile is approximately 15% lower on industrial costumers ratio than the

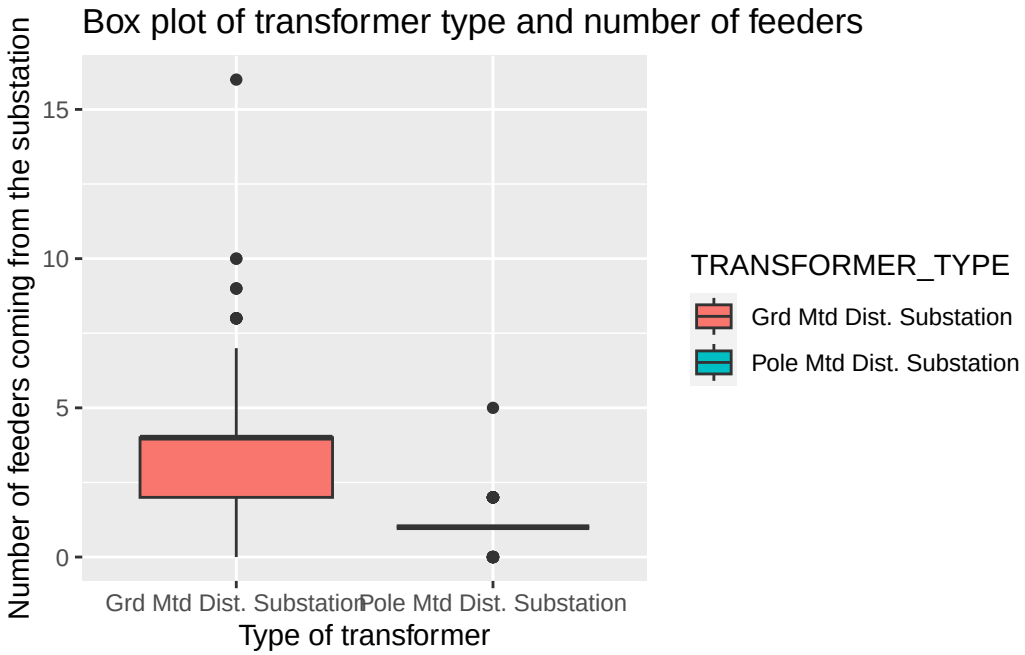
ground mounted 3rd quartile. It should also be noted that the 1st quartile of the ground mounted substations is above 0 meaning the large majority of ground mounted substations have some degree of industrial costumers.

```
modelTransTypeRatioIndustrialCostumers = lm(Percentage_IC ~ TRANSFORMER_TYPE,
    data = dataCharacteristics)

summary(modelTransTypeRatioIndustrialCostumers)
```

```
Call:
lm(formula = Percentage_IC ~ TRANSFORMER_TYPE, data = dataCharacteristics)

Residuals:
    Min      1Q  Median      3Q     Max
-0.4096 -0.3128 -0.2194  0.5057  0.7071

Coefficients:
                                        Estimate Std. Error t value Pr(>|t|)
(Intercept)                              0.40961    0.01521  26.932  < 2e-16
TRANSFORMER_TYPEPole Mtd Dist. Substation -0.11672    0.03010  -3.878 0.000113

(Intercept)                              ***
TRANSFORMER_TYPEPole Mtd Dist. Substation ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4041 on 946 degrees of freedom
Multiple R-squared:  0.01565,   Adjusted R-squared:  0.0146
F-statistic: 15.04 on 1 and 946 DF,  p-value: 0.0001128
```

As we can see from the summary of our model having a pole mounted substation as opposed to a ground mounted substation decreases the number the expected percentage of industrial costumers by approximately 11.7%. As observed by the R-squared value the type of transformer is capable of 40% of the variation of ratio of industrial costumers meaning we can really in our model to some degree but we shouldn't overly rely on it.

```
ggplot(data = dataCharacteristics, aes(x = TRANSFORMER_TYPE, y = LV_FEEDER_COUNT,
    fill = TRANSFORMER_TYPE)) + geom_boxplot() + ggtitle("Box plot of transformer type and
    xlab("Type of transformer") + ylab("Number of feeders coming from the substation")
```



The most apparent difference between ground mounted and pole mounted substations is how the 1st and 3rd quartile coincide on only 1 feeder for the pole mounted substations with no apparent whiskers with only 3 values marked as outliers, meaning almost all pole mounted substations have the same number of feeders. Ground mounted substations not only can have a considerably higher number of feeders, as the first quartile of the box is over the 3rd quartile of the pole mounted substations, the approximatly 75% of ground mounted substations have at least double the amount of feeders than pole mounted substations.

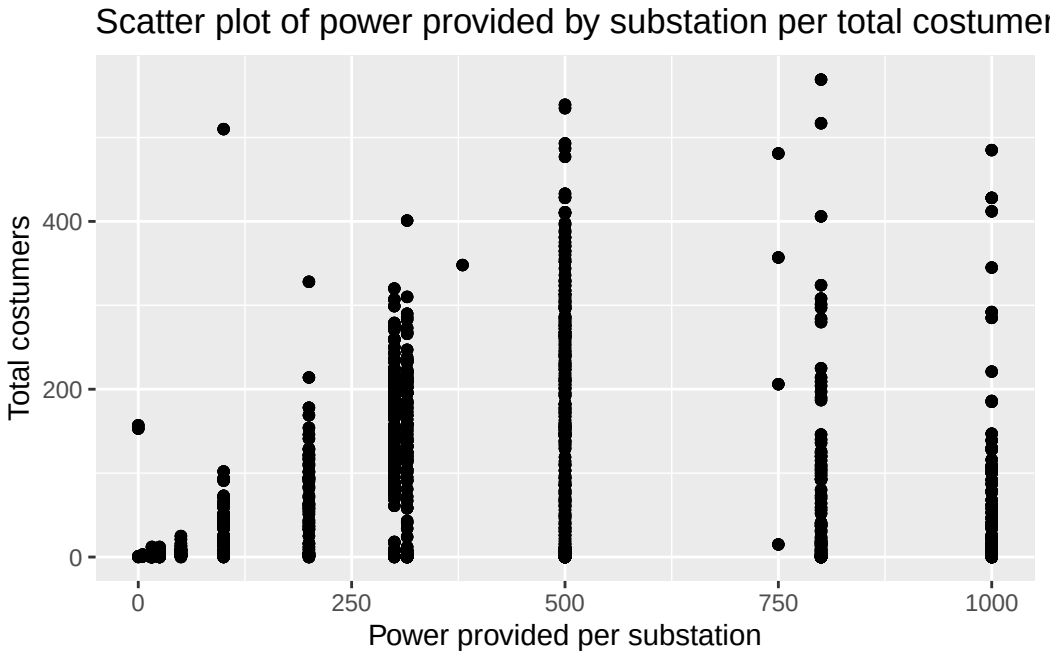
```
modelTransTypeFeederCount = lm(LV_FEEDER_COUNT ~ TRANSFORMER_TYPE, data = dataCharacterist

summary(modelTransTypeFeederCount)
```

```
Call:
lm(formula = LV_FEEDER_COUNT ~ TRANSFORMER_TYPE, data = dataCharacteristics)

Residuals:
    Min      1Q  Median      3Q     Max
-3.3555 -0.3555 -0.0289  0.6445 12.6445

Coefficients:
                                      Estimate Std. Error t value Pr(>|t|)
(Intercept)                            3.35552    0.06035   55.60   <2e-16
```

```
TRANSFORMER_TYPEPole Mtd Dist. Substation -2.32660    0.11945  -19.48   <2e-16

(Intercept)                                  ***
TRANSFORMER_TYPEPole Mtd Dist. Substation ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.604 on 946 degrees of freedom
Multiple R-squared:  0.2862,    Adjusted R-squared:  0.2855
F-statistic: 379.4 on 1 and 946 DF,  p-value: < 2.2e-16
```

Immediately we can see that the type of transformer isn't the best indicator of feeder count as the R-squared value is only 0,286 meaning it will only account for 28,6% of the variation on the number of feeders. It should still be noted that a pole mounted substation estimated value is very approximate to 1.

```
## these are: TRANSFORMER_TYPE, TOTAL_CUSTOMERS , Transformer_RATING ,
## Percentage_IC and LV_FEEDER_COUNT

ggplot(data = dataCharacteristics, aes(y = TOTAL_CUSTOMERS, x = Transformer_RATING)) +
    geom_point() + #geom_smooth(method = 'lm', se = FALSE) + geom_point()
    geom_point() + #geom_smooth(method = 'lm', se = FALSE) + +
    geom_point() + #geom_smooth(method = 'lm', se = FALSE) + #geom_smooth(method
    geom_point() + #geom_smooth(method = 'lm', se = FALSE) + = 'lm', se
    geom_point() + #geom_smooth(method = 'lm', se = FALSE) + = FALSE) +
ggtitle("Scatter plot of power provided by substation per total costumers") +
    xlab("Power provided per substation") + ylab("Total costumers")
```



The relationship between power provided and total costumers seem to follow a normal distribution without noticeable skewness.

```
modelPowerNcostumers = lm(Transformer_RATING ~ TOTAL_CUSTOMERS, data = dataCharacteristics

summary(modelPowerNcostumers)
```

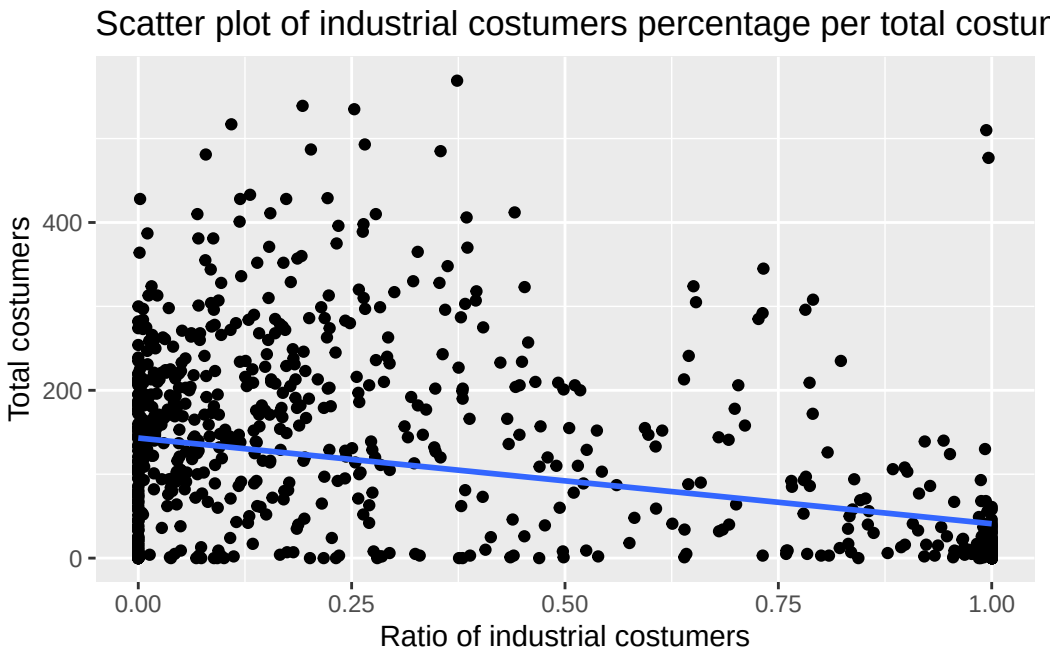
```
Call:
lm(formula = Transformer_RATING ~ TOTAL_CUSTOMERS, data = dataCharacteristics)

Residuals:
    Min      1Q  Median      3Q     Max
-525.15 -207.89  -76.78  141.43  671.59

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)     328.41083   12.22831  26.857  < 2e-16 ***
TOTAL_CUSTOMERS   0.58183    0.07855   7.407 2.85e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 279.4 on 946 degrees of freedom
```

```
Multiple R-squared:  0.05482,   Adjusted R-squared:  0.05382
F-statistic: 54.87 on 1 and 946 DF,  p-value: 2.85e-13
```

As we can see from the R-squared the relationship between total customers and power provided per substation being only approximately 5% means these two variables can barely explain the variation between each other. This could mean that the independent variable in the model is not relevant, or that there are other factors that are not included in the model that are affecting the dependent variable.

```
## these are: TRANSFORMER_TYPE, TOTAL_CUSTOMERS , Transformer_RATING ,
## Percentage_IC and LV_FEEDER_COUNT

ggplot(data = dataCharacteristics, aes(y = TOTAL_CUSTOMERS, x = Percentage_IC)) +
    geom_point() + geom_smooth(method = "lm", se = FALSE) + ggtitle("Scatter plot of indus
    xlab("Ratio of industrial costumers") + ylab("Total costumers")
```

```
`geom_smooth()` using formula = 'y ~ x'
```



As we can see from the scatter plot we can see too values with a very high concentration of occurrences around fully business costumers and fully industrial costumers. Around the 0% industrial costumers marks is where the largest concentration of total costumers per substation

are located and we can see a clear decreasing trend on the number of total costumers as the percentage of industrial costumers increase. However there does not seem to be any other clear trend between the high concentration of occurrences between none and exclusively industrial costumers.

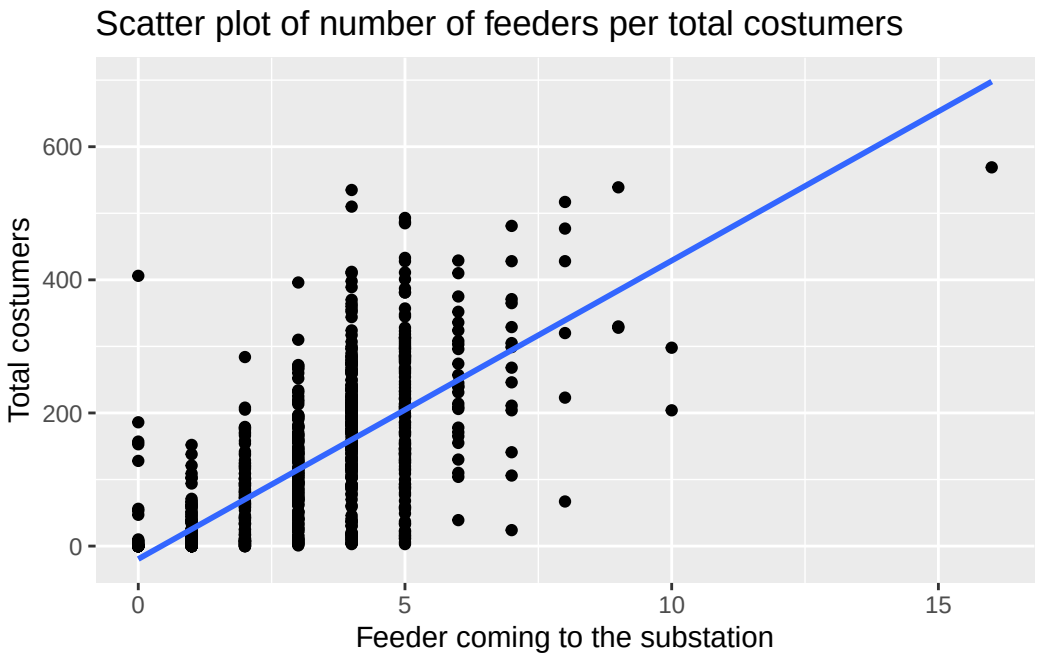
```
modelRatioIndustrialNcostumers = lm(TOTAL_CUSTOMERS ~ Percentage_IC, data = dataCharacteri

summary(modelRatioIndustrialNcostumers)
```

```
Call:
lm(formula = TOTAL_CUSTOMERS ~ Percentage_IC, data = dataCharacteristics)

Residuals:
    Min      1Q  Median      3Q     Max
-143.14  -55.85  -29.98   54.91  468.37

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    143.143      4.795   29.85   <2e-16 ***
Percentage_IC -102.161      8.614  -11.86   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 107.9 on 946 degrees of freedom
Multiple R-squared:  0.1294,    Adjusted R-squared:  0.1285
F-statistic: 140.7 on 1 and 946 DF,  p-value: < 2.2e-16
```

First thing to note is that how big of a difference is the number of total costumers for a fully industrial costumer substation compared to the intersect, so a fully business substation. Second thing to note is our quite low R-squared values of 12,9% meaning our model capability of explaining the number of costumers of one substation through the percentage of industrial costumers is incredibly small as these two variables are not linearly correlated enough to explain a significant proportion of the variation in the dependent variable.

```
## these are: TRANSFORMER_TYPE, TOTAL_CUSTOMERS , Transformer_RATING ,
## Percentage_IC and LV_FEEDER_COUNT

ggplot(data = dataCharacteristics, aes(y = TOTAL_CUSTOMERS, x = LV_FEEDER_COUNT)) +
    geom_point() + geom_smooth(method = "lm", se = FALSE) + ggtitle("Scatter plot of numbe
    xlab("Feeder coming to the substation") + ylab("Total costumers")
```

```
`geom_smooth()` using formula = 'y ~ x'
```



As we can clearly see as the number of feeders increase so does the number of costumers, following a very sharp increase in the first 5 feeders and a gradual increase in the remaining number of feeders.

```
modelNFeederslNcostumers = lm(TOTAL_CUSTOMERS ~ LV_FEEDER_COUNT, data = dataCharacteristic

summary(modelNFeederslNcostumers)
```

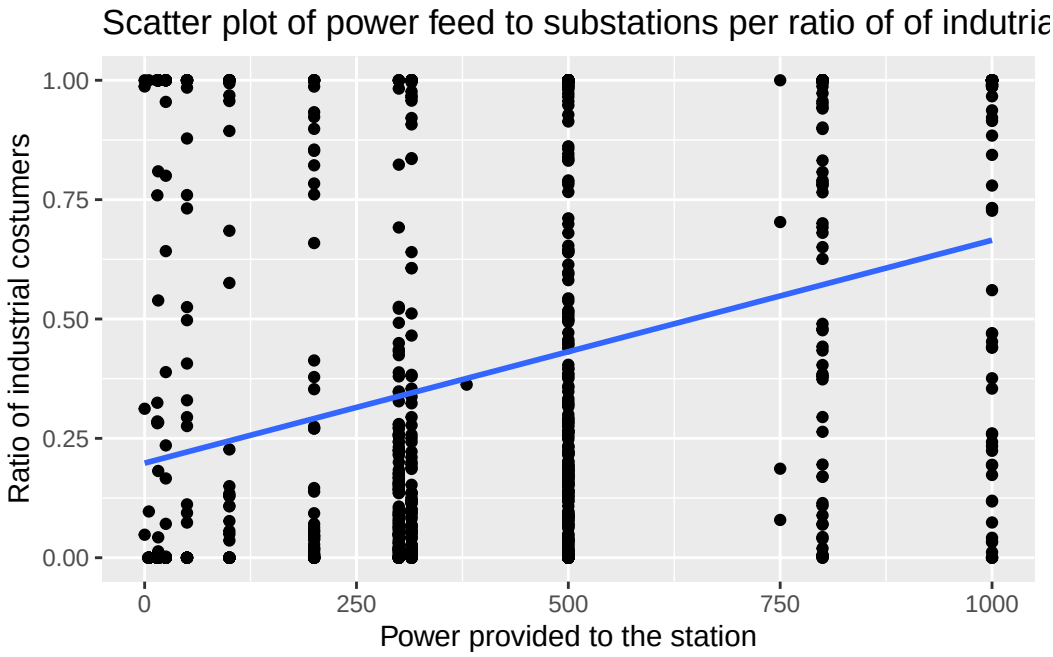
```
Call:
lm(formula = TOTAL_CUSTOMERS ~ LV_FEEDER_COUNT, data = dataCharacteristics)

Residuals:
    Min      1Q  Median      3Q     Max
-272.17  -25.37  -18.77   32.35  425.46

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)     -19.458      4.495  -4.328 1.66e-05 ***
LV_FEEDER_COUNT  44.828      1.342  33.405  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 78.34 on 946 degrees of freedom
Multiple R-squared:  0.5412,    Adjusted R-squared:  0.5407
F-statistic:  1116 on 1 and 946 DF,  p-value: < 2.2e-16
```

While just one singular linear model will not be as precise for the first 5 feeders, this model alone can explain more than half of the variation of the number of costumers of each substation.

```
## these are: TRANSFORMER_TYPE, TOTAL_CUSTOMERS , Transformer_RATING ,
## Percentage_IC and LV_FEEDER_COUNT

ggplot(data = dataCharacteristics, aes(x = Transformer_RATING, y = Percentage_IC)) +
    geom_point() + geom_smooth(method = "lm", se = FALSE) + ggtitle("Scatter plot of power
    xlab("Power provided to the station") + ylab("Ratio of industrial costumers")
```

```
`geom_smooth()` using formula = 'y ~ x'
```



From this graph we can see that substation with a bigger percentage of business costumers are more likely to provide less power than substations with a bigger industrial costumers base.

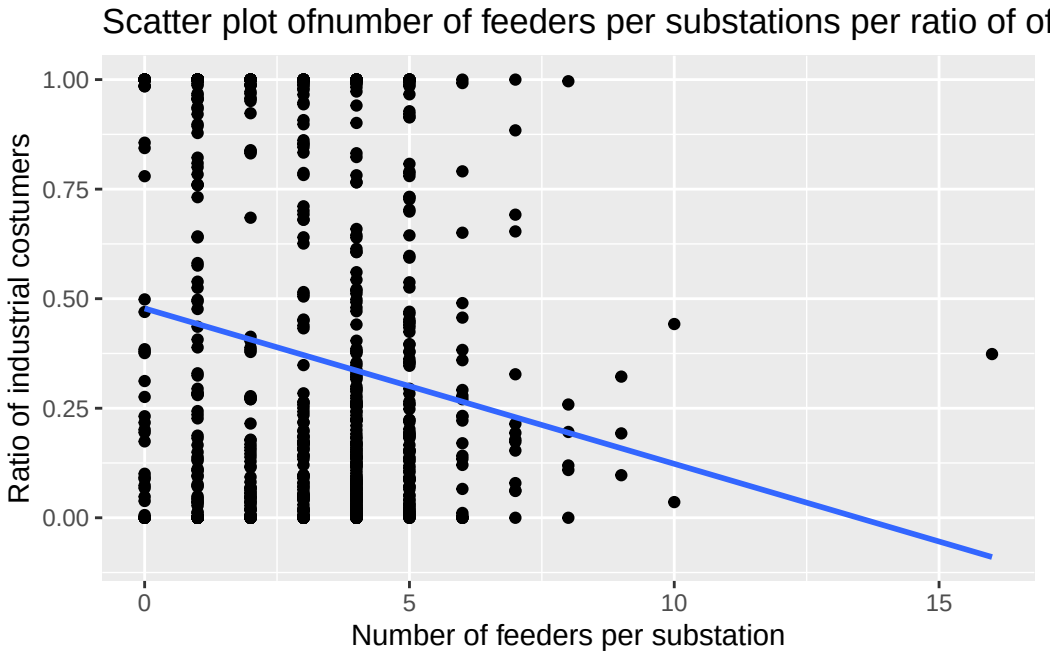
```
modelNFeederslNcostumers = lm(Percentage_IC ~ Transformer_RATING, data = dataCharacteristi

summary(modelNFeederslNcostumers)
```

```
Call:
lm(formula = Percentage_IC ~ Transformer_RATING, data = dataCharacteristics)

Residuals:
    Min      1Q  Median      3Q     Max
-0.6650 -0.2875 -0.1988  0.3350  0.8019

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)        0.1981319  0.0210372   9.418   <2e-16 ***
Transformer_RATING 0.0004669  0.0000435  10.733   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3846 on 946 degrees of freedom
Multiple R-squared:  0.1086,    Adjusted R-squared:  0.1076
F-statistic: 115.2 on 1 and 946 DF,  p-value: < 2.2e-16
```

As we can see from the model, the power provided to the substations does explain approximately 11% of the changes in percentage of industrial costumer per substation which is a extremely low level of explanatory power. Therefore we should not really on the power provided to substation to predict the ratio industrial costumers.

```
## these are: TRANSFORMER_TYPE, TOTAL_CUSTOMERS , Transformer_RATING ,
## Percentage_IC and LV_FEEDER_COUNT

ggplot(data = dataCharacteristics, aes(x = LV_FEEDER_COUNT, y = Percentage_IC)) +
    geom_point() + geom_smooth(method = "lm", se = FALSE) + ggtitle("Scatter plot ofnumber
    xlab("Number of feeders per substation") + ylab("Ratio of industrial costumers")
```

```
`geom_smooth()` using formula = 'y ~ x'
```



From the plot we can see that there is a tendency for the ratio of industrials costumers to decrease as the number of feeder increases per substation. It appears that the jump from when there are no feeders to the fist feeder might be an actual increase on the ratio industrial costumers but it is not going to be properly represented with a singular linear model.

```
modelNFeederslNcostumers = lm(Percentage_IC ~ LV_FEEDER_COUNT, data = dataCharacteristics)

summary(modelNFeederslNcostumers)
```

```
Call:
lm(formula = Percentage_IC ~ LV_FEEDER_COUNT, data = dataCharacteristics)

Residuals:
    Min      1Q  Median      3Q     Max
-0.4777 -0.3359 -0.1623  0.5073  0.8022

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)      0.477744   0.023051  20.726  < 2e-16 ***
LV_FEEDER_COUNT -0.035460   0.006881  -5.153 3.12e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 0.4017 on 946 degrees of freedom
Multiple R-squared:  0.02731,   Adjusted R-squared:  0.02628
F-statistic: 26.56 on 1 and 946 DF,  p-value: 3.117e-07
```

As we can see form the minuscule Multiple R-squared value this model simply is not an adequate and cannot bring any meaningful correlation between these two variables.

Exercise C 2.i

```
## as data from 0:00 to 7:50

tenMinuteMaximum = January_2013[, -c(1:2)]

## first we need to the maximum power output of the day i

maxVal = 0
January_2013$max = 0


for (i in 1:nrow(tenMinuteMaximum)) {

    ## the apply() function is used with the MARGIN parameter set to 1,
    ## which applies the function to all rows of the dataframe. Inside
    ## the function, we are checking if the row name is not equal to
    ## substation as the substation number is also bigger than the
    ## power measured in the interval of 10 mins

    maxVal = apply(tenMinuteMaximum[i, ], 1, max)

    # tenMinuteMaximum

    January_2013$max[i] = maxVal


}

January_2013[, 3:146] = January_2013[, 3:146]/January_2013$max


January_2013
```

```
# A tibble: 15,095 x 147
   Date       Substation `00:00` `00:10` `00:20` `00:30` `00:40` `00:50` `01:00`
   <date>          <int>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
 1 2013-01-03     511016   0.597   0.603   0.614   0.611   0.576   0.555   0.565
 2 2013-01-03     511029   0.624   0.723   0.754   0.644   0.802   0.835   0.843
 3 2013-01-03     511030   0.701   0.654   0.645   0.627   0.603   0.636   0.633
 4 2013-01-03     511033   0.297   0.285   0.288   0.293   0.295   0.298   0.299
 5 2013-01-03     511034   0.557   0.549   0.557   0.534   0.536   0.530   0.517
 6 2013-01-03     511035   0.524   0.591   0.586   0.547   0.523   0.490   0.479
 7 2013-01-03     511079   0.427   0.361   0.364   0.321   0.330   0.307   0.299
 8 2013-01-03     511150   0.137   0.140   0.132   0.133   0.136   0.137   0.141
 9 2013-01-03     511151   0.223   0.181   0.194   0.236   0.228   0.214   0.183
10 2013-01-03     511188   0.257   0.244   0.243   0.233   0.235   0.220   0.249
# ... with 15,085 more rows, and 138 more variables: `01:10` <dbl>,
#   `01:20` <dbl>, `01:30` <dbl>, `01:40` <dbl>, `01:50` <dbl>, `02:00` <dbl>,
#   `02:10` <dbl>, `02:20` <dbl>, `02:30` <dbl>, `02:40` <dbl>, `02:50` <dbl>,
#   `03:00` <dbl>, `03:10` <dbl>, `03:20` <dbl>, `03:30` <dbl>, `03:40` <dbl>,
#   `03:50` <dbl>, `04:00` <dbl>, `04:10` <dbl>, `04:20` <dbl>, `04:30` <dbl>,
#   `04:40` <dbl>, `04:50` <dbl>, `05:00` <dbl>, `05:10` <dbl>, `05:20` <dbl>,
#   `05:30` <dbl>, `05:40` <dbl>, `05:50` <dbl>, `06:00` <dbl>, ...
```

Exercise C 2.ii

```
## group by substation and get the mean of each time interval 0:00,
## 0:10, 0:20 etc.

January_2013Grouped = January_2013 %>%
    group_by(Substation) %>%
    summarise_all(mean)

January_2013Grouped = January_2013Grouped[, -c(2)]

January_2013Grouped
```

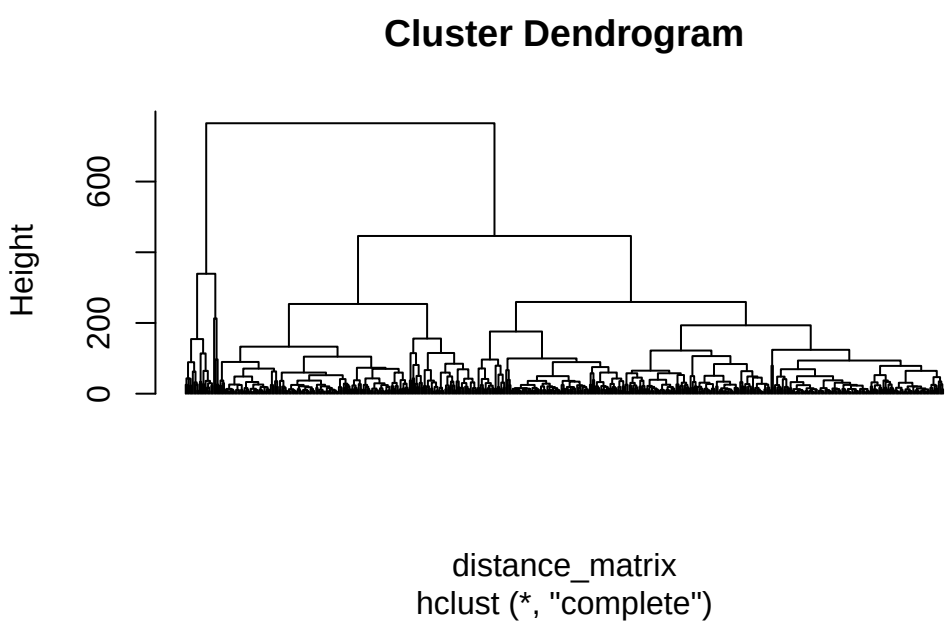
```
# A tibble: 535 x 146
   Substation `00:00` `00:10` `00:20` `00:30` `00:40` `00:50` `01:00` `01:10`
        <int>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
 1     511016   0.622   0.625   0.610   0.599   0.590   0.578   0.566   0.553
 2     511029   0.568   0.642   0.675   0.702   0.762   0.749   0.734   0.716
 3     511030   0.607   0.588   0.566   0.546   0.531   0.523   0.515   0.503
 4     511033   0.309   0.298   0.309   0.306   0.302   0.302   0.306   0.300
```

```
 5     511034   0.526   0.521   0.510   0.509   0.501   0.492   0.484   0.484
 6     511035   0.527   0.548   0.531   0.509   0.503   0.492   0.476   0.470
 7     511079   0.419   0.396   0.380   0.363   0.351   0.339   0.333   0.333
 8     511150   0.208   0.203   0.200   0.199   0.208   0.201   0.203   0.203
 9     511151   0.306   0.305   0.305   0.298   0.293   0.309   0.303   0.302
10     511188   0.340   0.338   0.341   0.341   0.336   0.340   0.345   0.346
# ... with 525 more rows, and 137 more variables: `01:20` <dbl>, `01:30` <dbl>,
#   `01:40` <dbl>, `01:50` <dbl>, `02:00` <dbl>, `02:10` <dbl>, `02:20` <dbl>,
#   `02:30` <dbl>, `02:40` <dbl>, `02:50` <dbl>, `03:00` <dbl>, `03:10` <dbl>,
#   `03:20` <dbl>, `03:30` <dbl>, `03:40` <dbl>, `03:50` <dbl>, `04:00` <dbl>,
#   `04:10` <dbl>, `04:20` <dbl>, `04:30` <dbl>, `04:40` <dbl>, `04:50` <dbl>,
#   `05:00` <dbl>, `05:10` <dbl>, `05:20` <dbl>, `05:30` <dbl>, `05:40` <dbl>,
#   `05:50` <dbl>, `06:00` <dbl>, `06:10` <dbl>, `06:20` <dbl>, ...
```

Exercise C 2.iii

Manhattan distance (also known as L1 distance) is a measure of the absolute differences between the coordinates of two points in a multi-dimensional space.

```r
January_2013Grouped = January_2013Grouped[, -c(1)]

distance_matrix <- dist(January_2013Grouped, method = "manhattan")

cluster = hclust(distance_matrix)

# plot(cluster, xlab='', sub='')

plot(cluster, labels = FALSE, hang = -1)
```



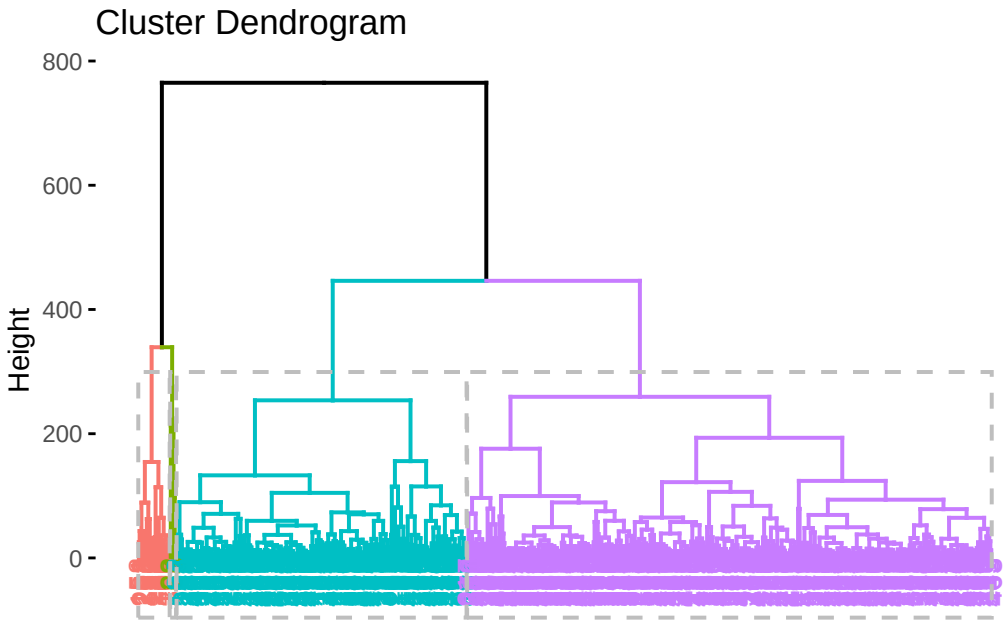
Exercise C 2.iv

Using the visualization method to decide the number of clusters we can observe from the dendrogram the natural breaks, where it branches into multiple clusters and we will be deciding if they should be a new cluster by comparing the height difference of the breaks. The initial data set is first divided into 2 with one of the subdivision containing most of the occurrences. Comparing the height between the second division of leafs and any possible further down divisions we can see that there is still a significant height difference. So we can go down a level and reaching the 3 divisions, into 3 clusters. If we further analyse any other branches with the same heigh as the 3 previous clusters we can see there is still 1 more with similar height on the very left side, making it possible to have 4 clusters without decreasing the height difference between branches. However if we want to make any further divisions we can see that there will be significantly smaller heights for any other possible clusters, meaning we would be sacrificing the similarity or distance between the clusters that are joined at this point.

```
## here we can see the the dendogram with the proper clusters already
## cut has mentioned above, using the factoextra library
fviz_dend(cluster, k = 4, rect = TRUE, show_labels = TRUE)
```

```
Warning: The `<scale>` argument of `guides()` cannot be `FALSE`. Use "none" instead as
of ggplot2 3.3.4.
```

```
i The deprecated feature was likely used in the factoextra package.
  Please report the issue at <https://github.com/kassambara/factoextra/issues>.
```



Exercise C 2.v

```
# cut the dendrogram into 4 clusters
groups <- cutree(cluster, k = 4)

# find number of observations in each cluster
table(groups)
```

```
groups
  1   2   3   4
 20 182   4 329
```

```
## now I will have to get an average of the entire cluster to plot

January_2013GroupedCluster = cbind(January_2013Grouped, cluster = groups)

## we will have to convert it to factor
```

```r
January_2013GroupedCluster$cluster = as.factor(January_2013GroupedCluster$cluster)
```

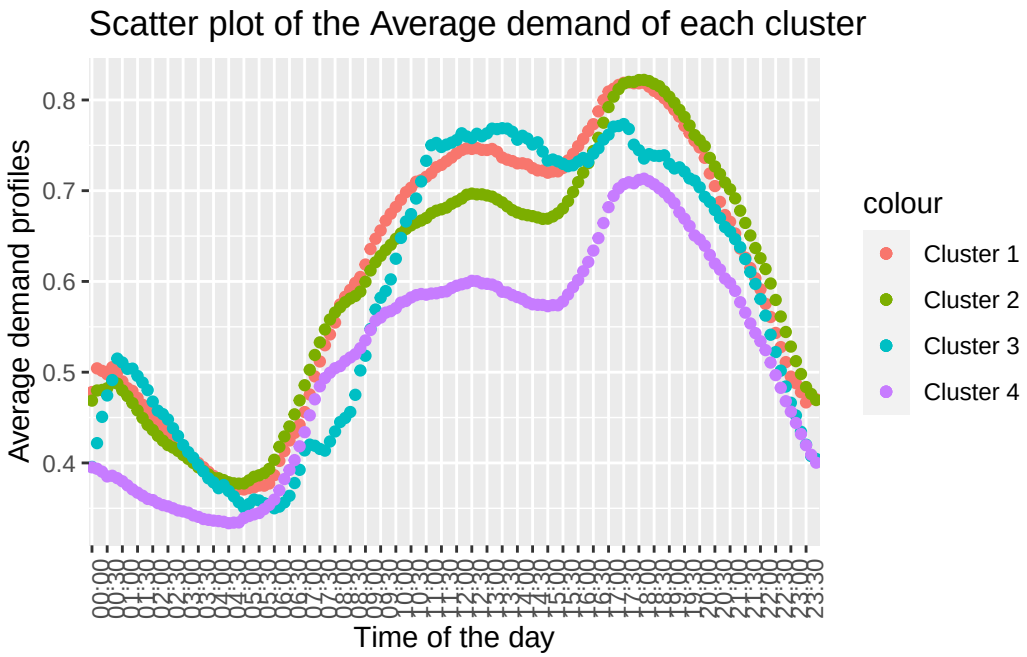
```r
## now we will be grouping by cluster to be able to have the average of
## each time interval

January_2013GroupedClusterAverage = January_2013GroupedCluster %>%
    group_by(cluster) %>%
    summarise_all(mean)


## we got to transform this into vertical data so we can have a colum
## of time and the power percent to plot

January_2013GroupedClusterAverageVertical = transpose(January_2013GroupedClusterAverage)


# Change colnames of all columns
colnames(January_2013GroupedClusterAverageVertical) <- c("Cluster1", "Cluster2",
    "Cluster3", "Cluster4")

## we will also be dropping the first row as we no longer need the
## cluster variables and these will interfere with the plotting
January_2013GroupedClusterAverageVertical = January_2013GroupedClusterAverageVertical[-1,
    ]

## we will also be dropping the last row as we no longer need the max
## values and these will interfere with the plotting as well

January_2013GroupedClusterAverageVertical = January_2013GroupedClusterAverageVertical[-nro
    ]
```

```r
time_interval <- seq(from = as.numeric(as.POSIXlt("0000", format = "%H%M")),
    to = as.numeric(as.POSIXlt("2350", format = "%H%M")), by = 10 * 60)
time_interval <- format(as.POSIXlt(time_interval, origin = "1970-01-01"),
    format = "%H:%M")

January_2013GroupedClusterAverageVertical$timeIntervals = time_interval

str(January_2013GroupedClusterAverageVertical)
```

```
'data.frame':   144 obs. of  5 variables:
 $ Cluster1     : chr  "0.478124511966287" "0.504229973125133" "0.50143789507594" "0.4976846
 $ Cluster2     : chr  "0.468656498878136" "0.479812493904961" "0.480405945508163" "0.482260
 $ Cluster3     : chr  "0.395641381958471" "0.42181157694028" "0.450751323868799" "0.4743491
 $ Cluster4     : chr  "0.395191738014495" "0.393601699730004" "0.390332347150401" "0.385122
 $ timeIntervals: chr  "00:00" "00:10" "00:20" "00:30" ...
```

```r
## it seems we will have to convert the values of the clusters to float

January_2013GroupedClusterAverageVertical$Cluster1 = as.double(January_2013GroupedClusterA
January_2013GroupedClusterAverageVertical$Cluster2 = as.double(January_2013GroupedClusterA
January_2013GroupedClusterAverageVertical$Cluster3 = as.double(January_2013GroupedClusterA
January_2013GroupedClusterAverageVertical$Cluster4 = as.double(January_2013GroupedClusterA
```

```r
ggplot() + geom_point(data = January_2013GroupedClusterAverageVertical, aes(x = timeInterv
    y = Cluster1, color = "Cluster 1")) + geom_point(data = January_2013GroupedClusterAver
    aes(x = timeIntervals, y = Cluster2, color = "Cluster 2")) + geom_point(data = January
    aes(x = timeIntervals, y = Cluster3, color = "Cluster 3")) + geom_point(data = January
    aes(x = timeIntervals, y = Cluster4, color = "Cluster 4")) + ggtitle("Scatter plot of
    xlab("Time of the day") + ylab("Average demand profiles") + theme(axis.text.x = elemen
    scale_x_discrete(breaks = c("00:00", "01:00", "02:00", "03:00", "04:00",
        "05:00", "06:00", "07:00", "08:00", "09:00", "10:00", "11:00", "12:00",
        "13:00", "14:00", "15:00", "16:00", "17:00", "18:00", "19:00", "20:00",
        "21:00", "22:00", "23:00", "00:30", "01:30", "02:30", "03:30", "04:30",
        "05:30", "06:30", "07:30", "08:30", "09:30", "10:30", "11:30", "12:30",
        "13:30", "14:30", "15:30", "16:30", "17:30", "18:30", "19:30", "20:30",
        "21:30", "22:30", "23:30"))
```



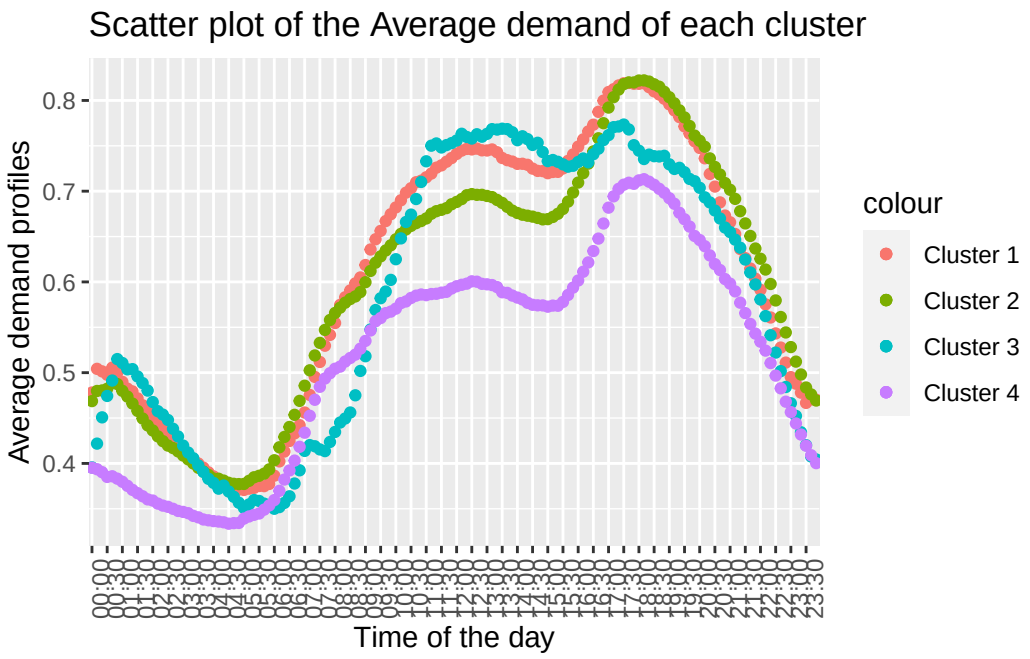
Exercise C 2.vi

```
January_2013Weekdays = January_2013 %>%
    group_by(Substation) %>%
    summarise_all(mean)
January_2013Weekdays = cbind(January_2013Weekdays, cluster = groups)

## Now we got to check which days are week days and which days are
## weekends

January_2013Weekdays$isWeekDay = 1

for (i in 1:nrow(January_2013Weekdays)) {
    if (weekdays(January_2013Weekdays$Date[i]) == "Saturday") {
        January_2013Weekdays$isWeekDay[i] = 0
    }
    if (weekdays(January_2013Weekdays$Date[i]) == "Sunday") {
        January_2013Weekdays$isWeekDay[i] = 0
    }
}
```

```
## then I will group by weekday and substation? TODO figure out
```

Exercise C 2.vii

```
ggplot() + geom_point(data = January_2013GroupedClusterAverageVertical, aes(x = timeInterv
    y = Cluster1, color = "Cluster 1")) + geom_point(data = January_2013GroupedClusterAver
    aes(x = timeIntervals, y = Cluster2, color = "Cluster 2")) + geom_point(data = January
    aes(x = timeIntervals, y = Cluster3, color = "Cluster 3")) + geom_point(data = January
    aes(x = timeIntervals, y = Cluster4, color = "Cluster 4")) + ggtitle("Scatter plot of
    xlab("Time of the day") + ylab("Average demand profiles") + theme(axis.text.x = elemen
    scale_x_discrete(breaks = c("00:00", "01:00", "02:00", "03:00", "04:00",
        "05:00", "06:00", "07:00", "08:00", "09:00", "10:00", "11:00", "12:00",
        "13:00", "14:00", "15:00", "16:00", "17:00", "18:00", "19:00", "20:00",
        "21:00", "22:00", "23:00", "00:30", "01:30", "02:30", "03:30", "04:30",
        "05:30", "06:30", "07:30", "08:30", "09:30", "10:30", "11:30", "12:30",
        "13:30", "14:30", "15:30", "16:30", "17:30", "18:30", "19:30", "20:30",
        "21:30", "22:30", "23:30"))
```




```
table(groups)
```

```
groups
  1   2   3   4
 20 182   4 329
```

As we can see from the number of substations in each cluster, cluster 4 has the highest number of substations. This means that these 349 substations are all relatively similar. We can also infer from the previously displayed dendrogram as that the height on any other potential new leaf would be significantly smaller than the already cut cluster leaves.

As the cluster 4 is the most numerous by far it can be considered the baseline for daily power demand of the "common" substation seeing as it represents approximately 65% of all substations.

There are several ways to compare the characteristics data for each of the clusters in R. We can use the tapply() function to calculate summary statistics (e.g mean, median, standard deviation) for each cluster.

```
## we will have to use a alternative version of the first grouping made
## on the clusters so we can group by cluster again

January_2013Analyses = January_2013 %>%
    group_by(Substation) %>%
    summarise_all(mean)
January_2013Analyses = cbind(January_2013Analyses, cluster = groups)

# with this we now have the corresponding cluster, the date and the
# substation

# cluster_summary <- January_2013GroupedClusterAverageVertical %>%
# group_by(cluster) %>% summarize(mean = mean(characteristic), median =
# median(characteristic), sd = sd(characteristic))
```

```
# fviz_dend(cluster, k = 4, rect = TRUE, show_labels = TRUE)
```