Sensor-Data-Cleaning-and-Inference

Paula Ramirez

07/10/2024

Sensor-Data-Cleaning-and-Inference

1. Data Transformation and Preparation

1. Initial Transformation

Basic configurations

```
#Clearing all the plots, the console and the workspace.
#Setting the overall format for numbers.
if(!is.null(dev.list())) dev.off()
```

```
## null device
##           1
```

```
cat("\014")
```

```
rm(list=ls())
options(scipen=9)
```

a. Reading the file and appending my initials to all variables in the data frame

```
#Reading the file and
sensor_tracking_PR <- read.table(here("Sensor-Data-Cleaning-and-Inference", "Data_Sensor.txt"),
                                 header = TRUE, sep = ",")
#Converting it to dataframe.
sensor_tracking_PR <- as.data.frame(sensor_tracking_PR)
#Append PR initials to all variables in the dataframe
colnames(sensor_tracking_PR) <- paste(colnames(sensor_tracking_PR), "PR", sep = "_")
#Showing first results
head(sensor_tracking_PR)
```

```
##   Index_PR Room_PR Ren_PR DT_PR TM_PR S1_L_PR S2_L_PR S3_L_PR S1_T_PR S2_T_PR
## 1        1     111    New    47    21       1       1       1  24.479  21.651
## 2        2     111    New     3     2       1       1       1  23.771  24.167
## 3        3     311    Old    27    10       1       1       1  24.760  21.143
## 4        4     211    New     5    22       1       1       1  22.130  23.925
## 5        5     311    Old    28    18       1       1       1  24.425  24.832
## 6        6     211    New     9    18       1       1       1  22.158  20.393
##   S3_T_PR FN_PR
## 1  22.227  1106
## 2  22.104  1146
## 3  22.881  1109
## 4  19.353  1111
## 5  24.287  1120
## 6  26.612  1078
```

b. Transform character variables to factor variables

```
#Changing to factor
sensor_tracking_PR$Ren_PR <- as.factor(sensor_tracking_PR$Ren_PR)
str(sensor_tracking_PR)
```

```
## 'data.frame':    4323 obs. of  12 variables:
##  $ Index_PR: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Room_PR : int  111 111 311 211 311 211 211 211 211 211 ...
##  $ Ren_PR  : Factor w/ 2 levels "New","Old": 1 1 2 1 2 1 1 1 1 1 ...
##  $ DT_PR   : int  47 3 27 5 28 9 58 28 18 57 ...
##  $ TM_PR   : int  21 2 10 22 18 18 14 13 13 16 ...
##  $ S1_L_PR : int  1 1 1 1 1 1 0 0 1 1 ...
##  $ S2_L_PR : int  1 1 1 1 1 1 0 0 1 1 ...
##  $ S3_L_PR : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ S1_T_PR : num  24.5 23.8 24.8 22.1 24.4 ...
##  $ S2_T_PR : num  21.7 24.2 21.1 23.9 24.8 ...
##  $ S3_T_PR : num  22.2 22.1 22.9 19.4 24.3 ...
##  $ FN_PR   : int  1106 1146 1109 1111 1120 1078 1079 1112 1116 1172 ...
```

2. Outliers

Finding outliers

a. Techniques to identify outliers.

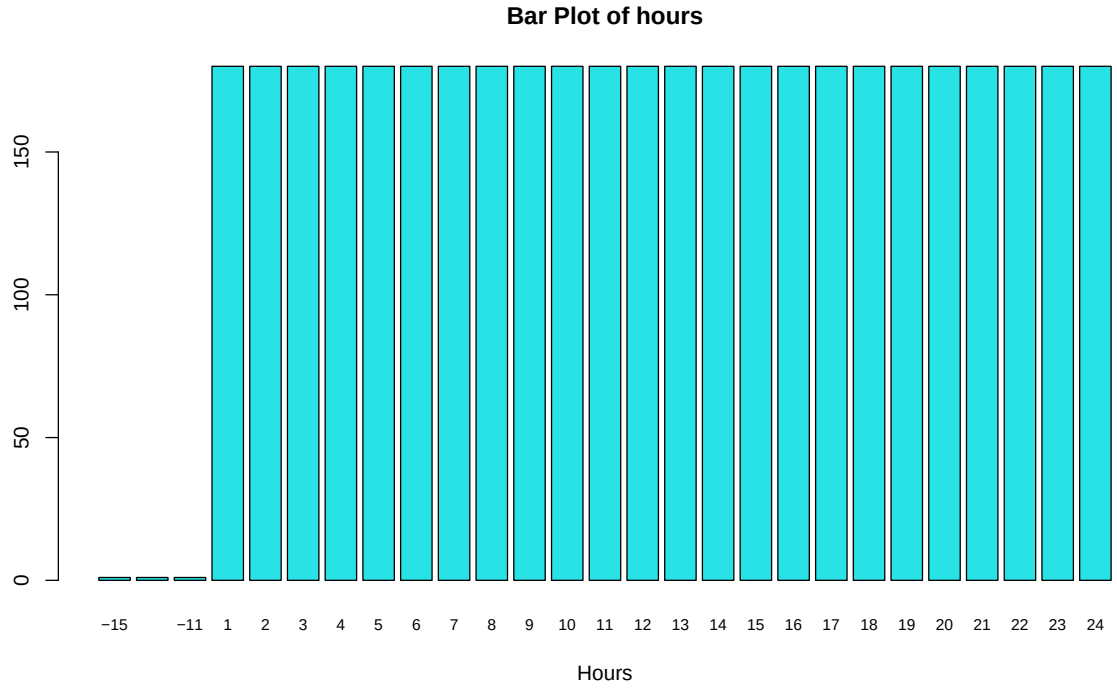
```
# Analyzing dates and times out of range using the range function
range(sensor_tracking_PR$DT_PR)
```

```
## [1]  1 60
```

```
range(sensor_tracking_PR$TM_PR)
```

```
## [1] -15  24
```

```
# Analyzing odd values in the time variable
barplot(table(sensor_tracking_PR$TM_PR),
        col=5,
        main="Bar Plot of hours", cex.names=.75,
        xlab="Hours")
```




```
#Exploring the status of light from sensors with unique function (must be only 1 and 0)
unique(sensor_tracking_PR$S1_L_PR)
```

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

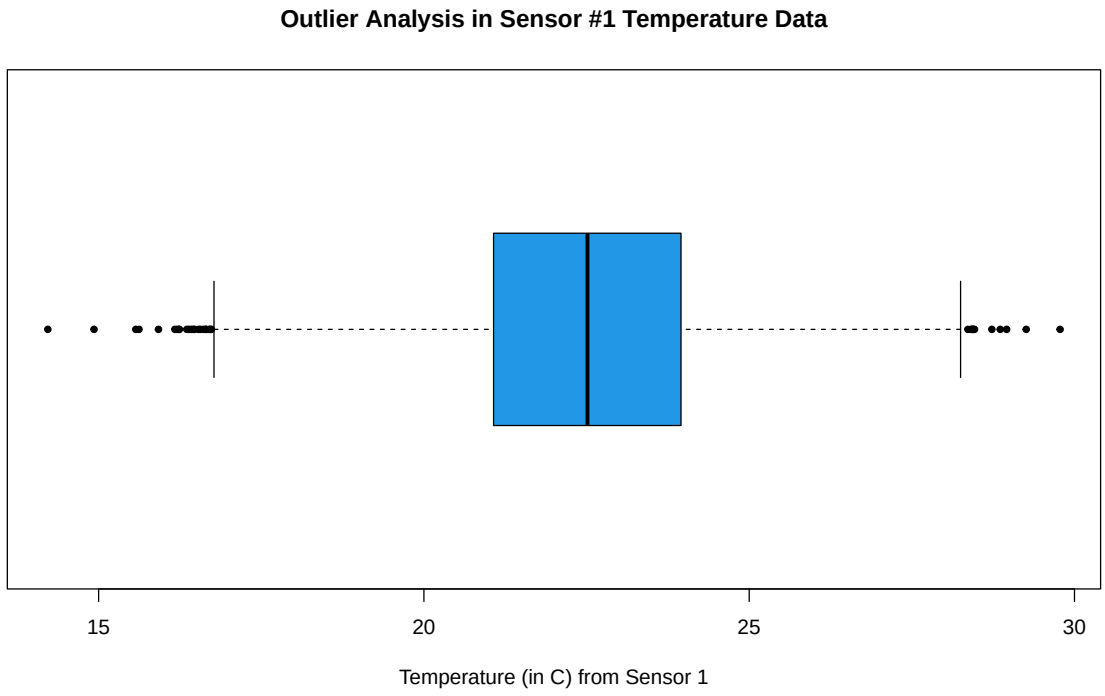
```r
unique(sensor_tracking_PR$S2_L_PR)
```

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

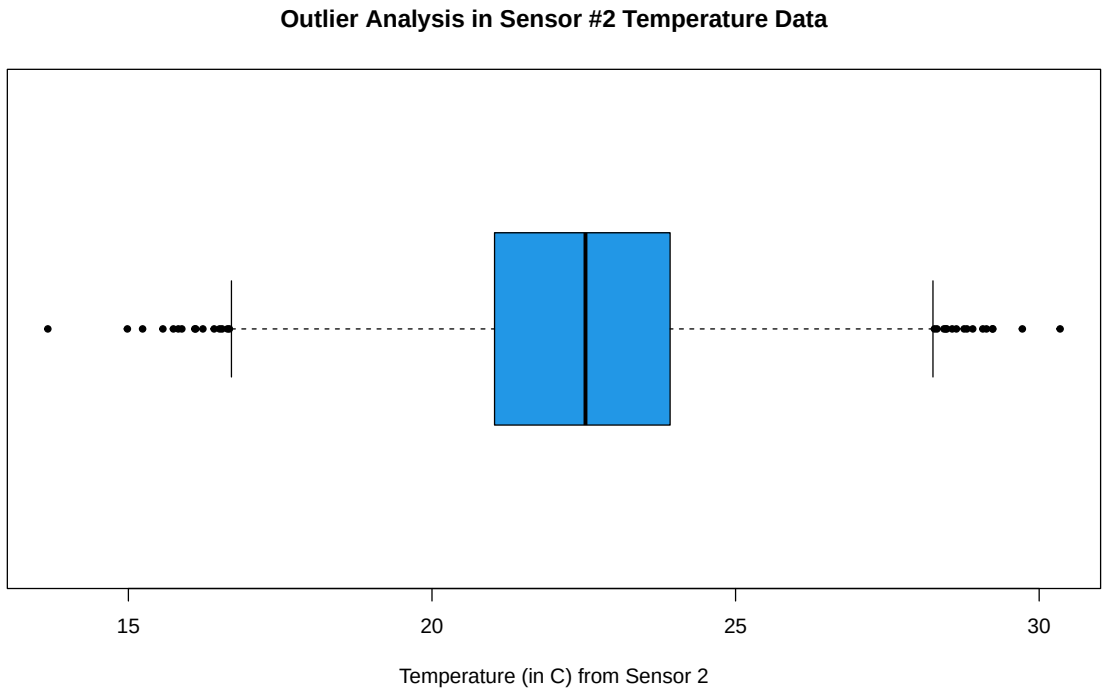
```r
unique(sensor_tracking_PR$S3_L_PR)
```

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

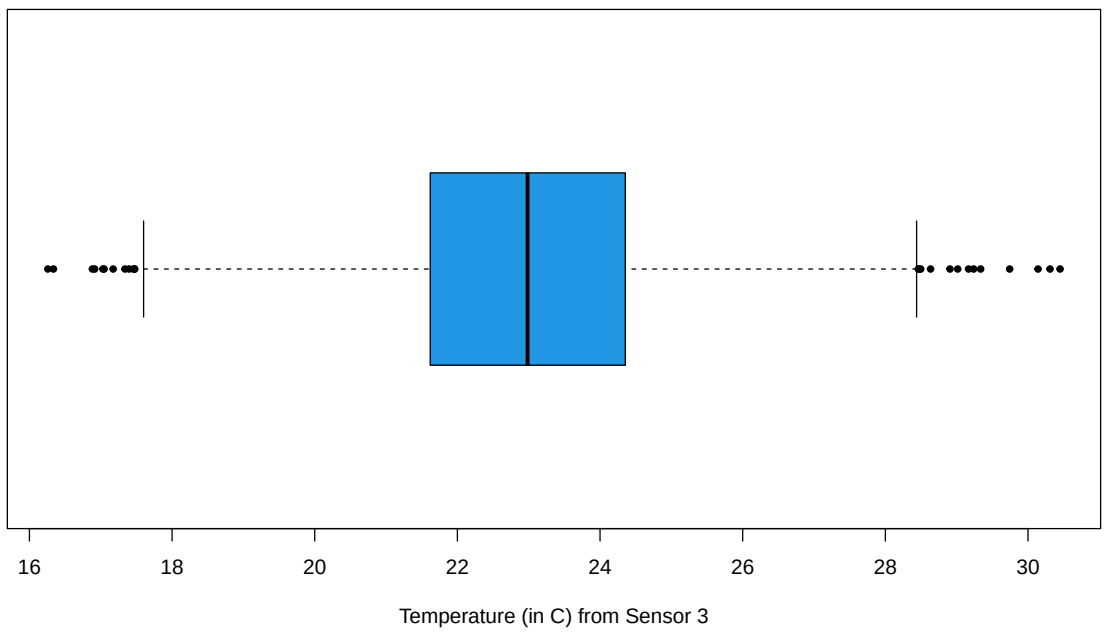
```r
#Box plot to identify outliers
boxplot(sensor_tracking_PR$S1_T_PR, horizontal=TRUE, pch=20,
        main="Outlier Analysis in Sensor #1 Temperature Data",
        xlab="Temperature (in C) from Sensor 1",
        col=4)
```



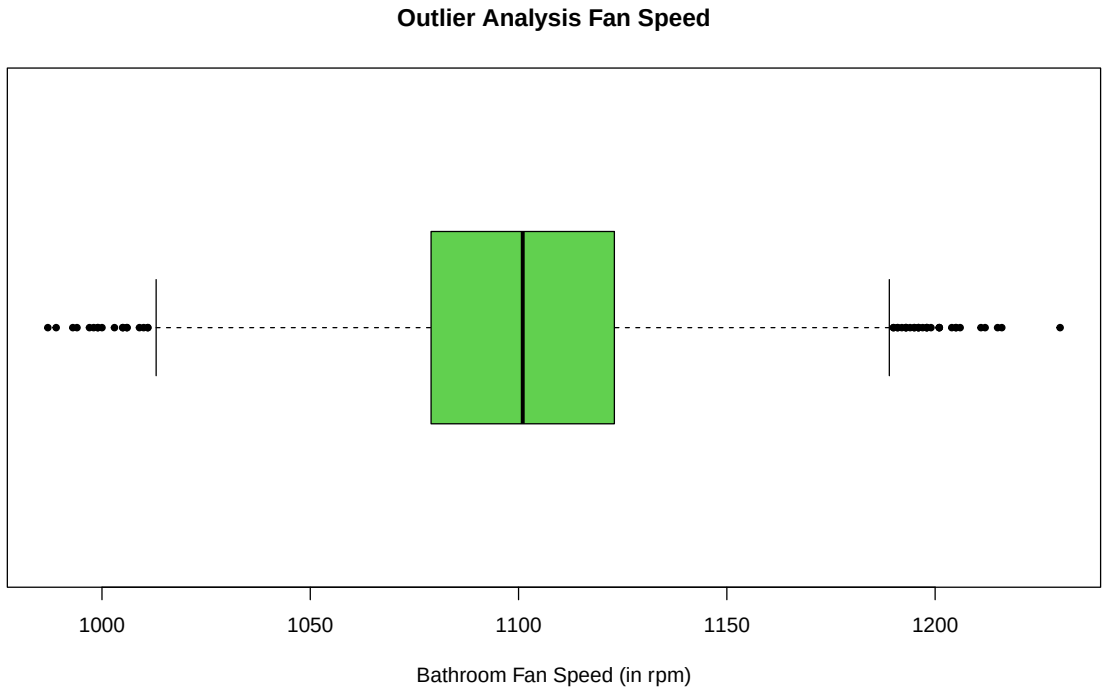

```r
boxplot(sensor_tracking_PR$S2_T_PR, horizontal=TRUE, pch=20,
        main="Outlier Analysis in Sensor #2 Temperature Data",
        xlab="Temperature (in C) from Sensor 2",
        col=4)
```




```
boxplot(sensor_tracking_PR$S3_T_PR, horizontal=TRUE, pch=20,
        main="Outlier Analysis in Sensor #3 Temperature Data",
        xlab="Temperature (in C) from Sensor 3",
        col=4)
```



```r
boxplot(sensor_tracking_PR$FN_PR, horizontal=TRUE, pch=20,
        main="Outlier Analysis Fan Speed",
        xlab="Bathroom Fan Speed (in rpm)",
        col=3)
```

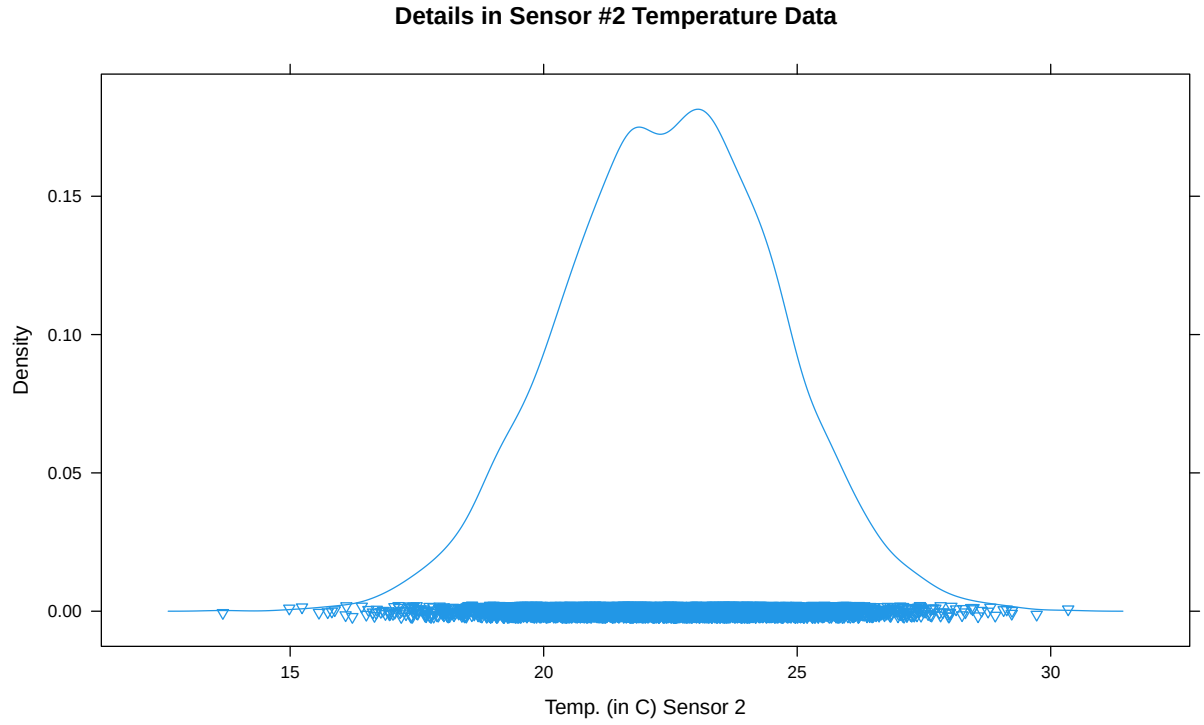


b. Comment on any outliers you see and deal with them appropriately. Make sure you explain why you dealt with them the way you decided to.

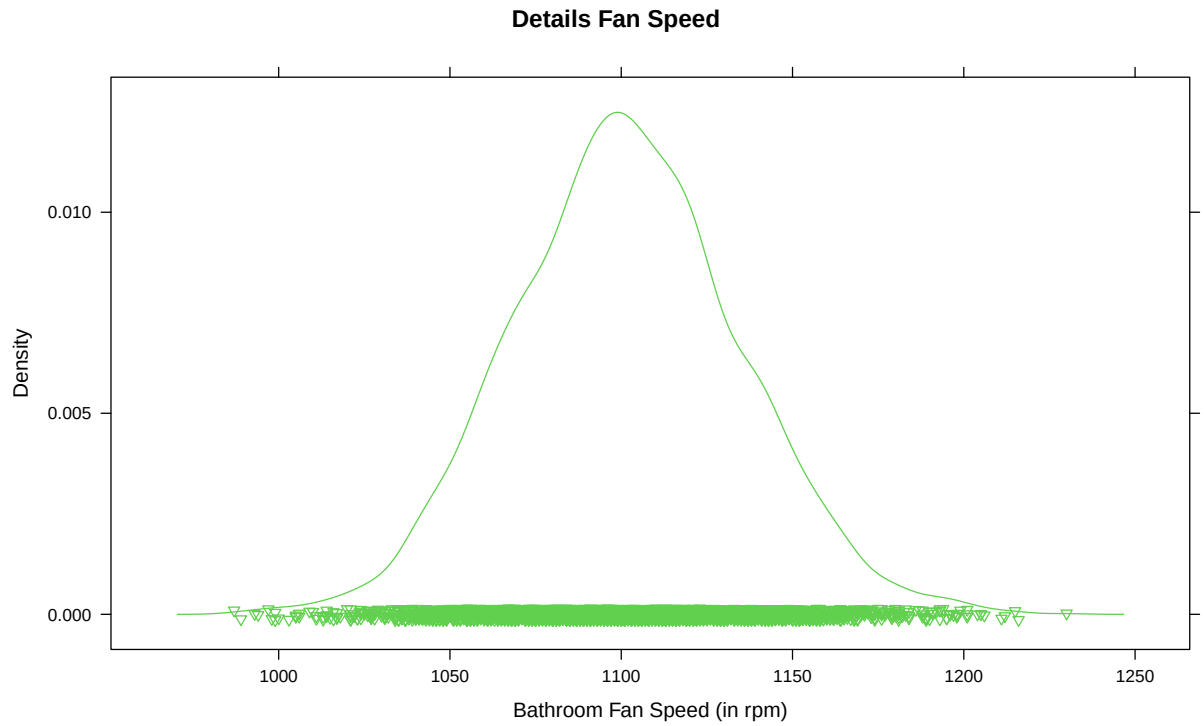
Observations and findings:

- Using the range function and bar plot, I identified some unusual values in the time variable (-15, -12, -11).
- I analyzed the sensor light values with unique function to find odd values but found no anomalies.
- I found some relevant outliers in S1_T_PR (temperature-sensor 1), S2_T_PR (temperature-sensor 2) and FN_PR (fan speed) in the box plot. These values are outside the expected range for fan speed and temperature variables. To get more details, I created a density plot for these variables and analyzed them with some categorical variables like rooms and room conditions.

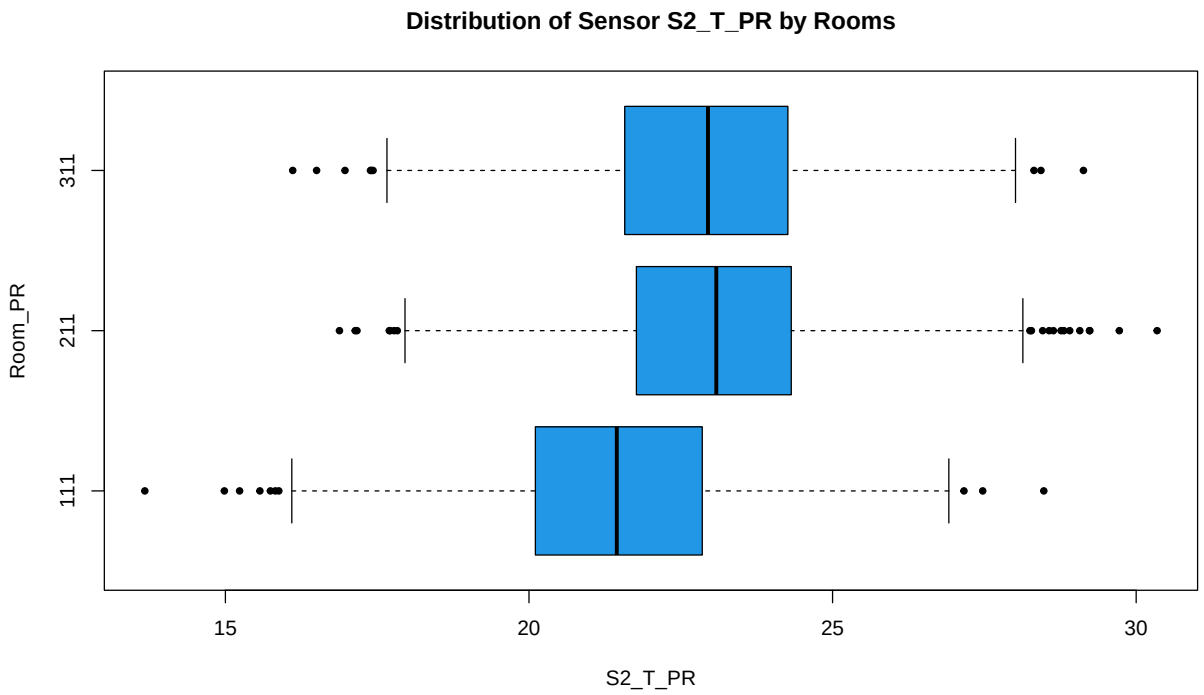
```r
#Density Plot - looking for more details in S2 and fan speed
densityplot( ~ sensor_tracking_PR$S2_T_PR, pch=6,
             main='Details in Sensor #2 Temperature Data',
             xlab="Temp. (in C) Sensor 2",
             col=4)
```



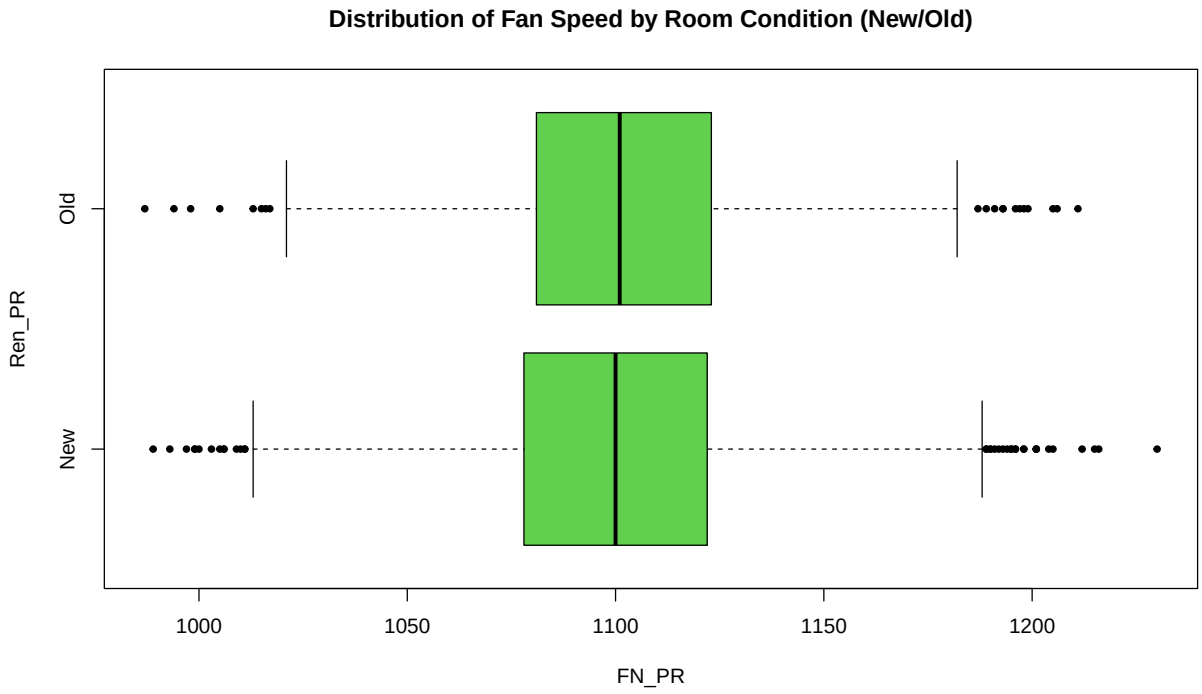

```
densityplot( ~ sensor_tracking_PR$FN_PR, pch=6,
             main='Details Fan Speed',
             xlab="Bathroom Fan Speed (in rpm)",
             col=3)
```



```r
# Identify if outliers in S2_T are specific to a certain room
boxplot(S2_T_PR ~ Room_PR ,
        data=sensor_tracking_PR,
        main="Distribution of Sensor S2_T_PR by Rooms",
        horizontal=TRUE, col=4,pch=20)
```




```r
# Identify if outliers in Fan  Speed are specific to a certain room condition
boxplot(FN_PR ~ Ren_PR ,
        data=sensor_tracking_PR,
        main="Distribution of Fan Speed by Room Condition (New/Old)",
        horizontal=TRUE, col=3,pch=20)
```

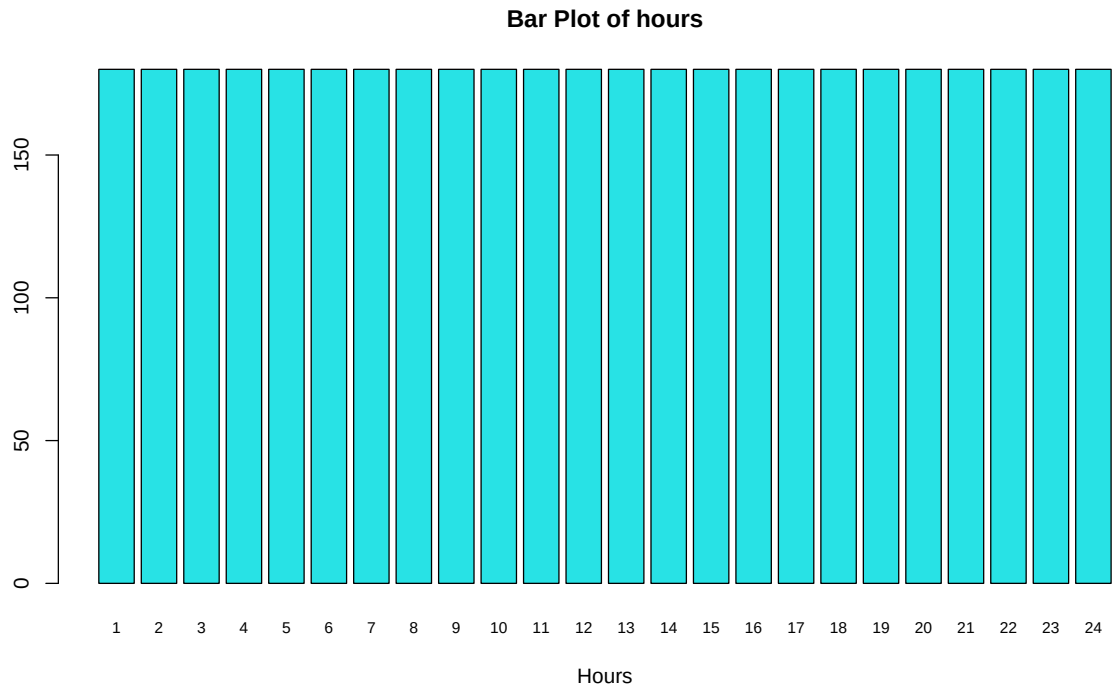


Decisions

- I decided to eliminate rows with unusual times, as they seem to be wrong records.
- In those density plots, I found two data points significantly distant from the other outliers. For S1_T_PR and S2_T_PR minimum and maximum points, for FN_PR maximum point.

Due to this, I decided to remove it from the data set.

```
#TIME OF DAY
#Find row number with odd values (-15, -12, -11)
odds_hours_pr <- which(sensor_tracking_PR$TM_PR %in% c(-15,-12,-11))
#Deleting the data points in hours
sensor_tracking_PR <- sensor_tracking_PR[-c(odds_hours_pr),]
barplot(table(sensor_tracking_PR$TM_PR),
        col=5,
        main="Bar Plot of hours", cex.names=.75,
        xlab="Hours")
```



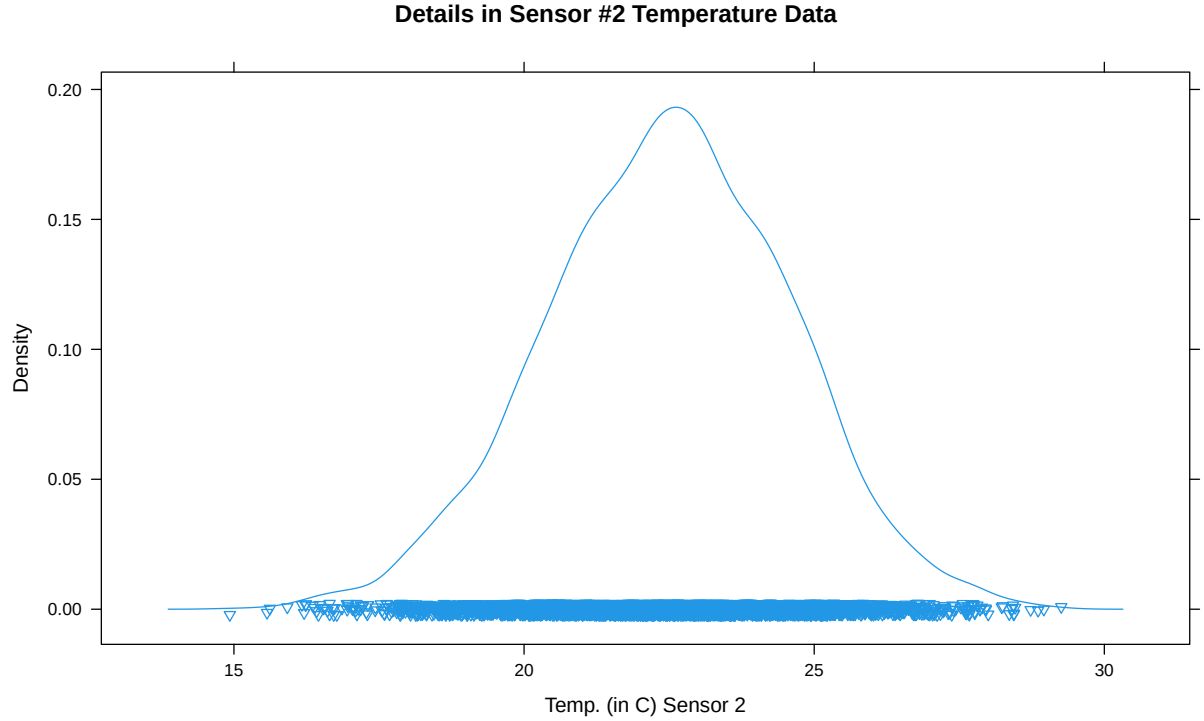

```
#SENSOR NUMBER 1
#Finding record with minimum and maximum value and deleting it
head(sensor_tracking_PR[order(sensor_tracking_PR$S1_T_PR),],1)
```

```
##      Index_PR Room_PR Ren_PR DT_PR TM_PR S1_L_PR S2_L_PR S3_L_PR S1_T_PR
## 1511     1511     111    New    47     3       0       0       1  14.219
##      S2_T_PR S3_T_PR FN_PR
## 1511  24.259  19.047  1133
```

```
head(sensor_tracking_PR[rev(order(sensor_tracking_PR$S1_T_PR)),],1)
```

```
##     Index_PR Room_PR Ren_PR DT_PR TM_PR S1_L_PR S2_L_PR S3_L_PR S1_T_PR S2_T_PR
## 784      784     311    Old    43    23       1       1       1  29.779  24.333
##     S3_T_PR FN_PR
## 784  27.145  1089
```

```
#Row number for max value
max_sensor1_pr <- which(sensor_tracking_PR$S1_T_PR == max(sensor_tracking_PR$S1_T_PR))
#Row number for min value
min_sensor1_pr <- which(sensor_tracking_PR$S1_T_PR == min(sensor_tracking_PR$S1_T_PR))
sensor_tracking_PR <- sensor_tracking_PR[-c(min_sensor1_pr,max_sensor1_pr),]
densityplot( ~ sensor_tracking_PR$S1_T_PR, pch=6,
             main='Details in Sensor #2 Temperature Data',
             xlab="Temp. (in C) Sensor 2",
             col=4)
```



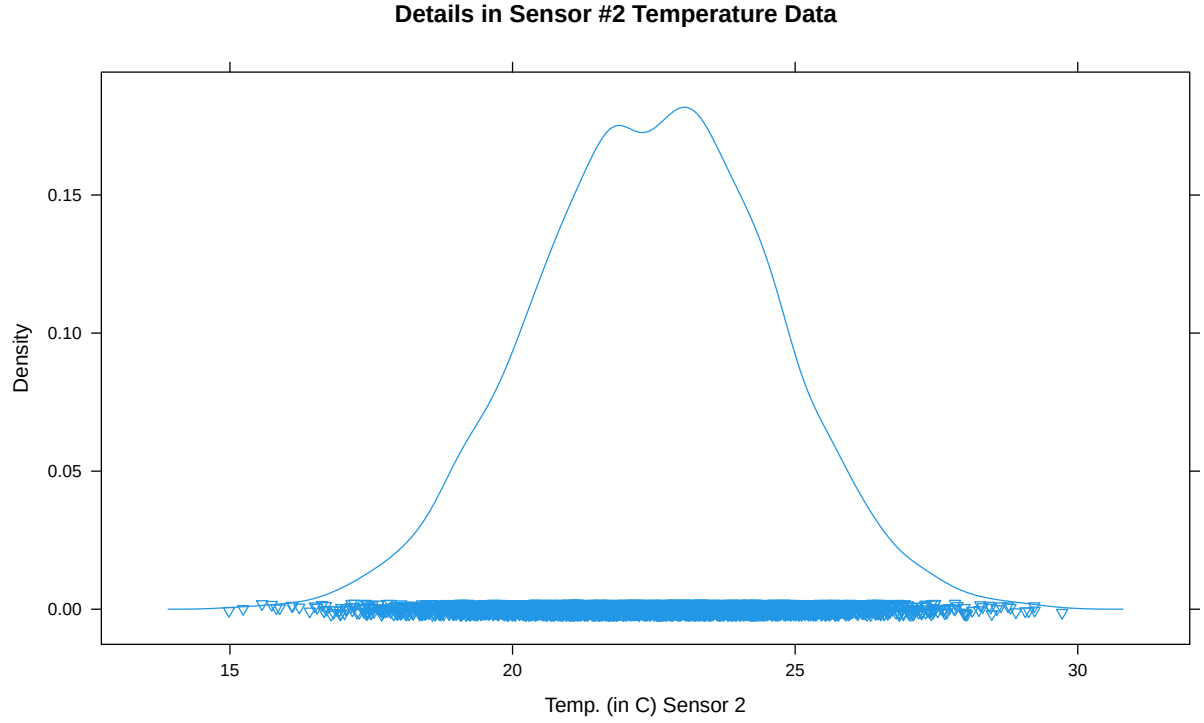

```
#SENSOR NUMBER 2
#Finding record with minimum and maximum value and deleting it
head(sensor_tracking_PR[order(sensor_tracking_PR$S2_T_PR),],1)
```

```
##     Index_PR Room_PR Ren_PR DT_PR TM_PR S1_L_PR S2_L_PR S3_L_PR S1_T_PR S2_T_PR
## 677      677     111    New    12    12       1       1       1  22.894  13.673
##     S3_T_PR FN_PR
## 677  22.563  1177
```

```
head(sensor_tracking_PR[rev(order(sensor_tracking_PR$S2_T_PR)),],1)
```

```
##      Index_PR Room_PR Ren_PR DT_PR TM_PR S1_L_PR S2_L_PR S3_L_PR S1_T_PR
## 3353     3353     211    New    36     4       0       0       1  23.298
##      S2_T_PR S3_T_PR FN_PR
## 3353  30.345   22.28  1142
```

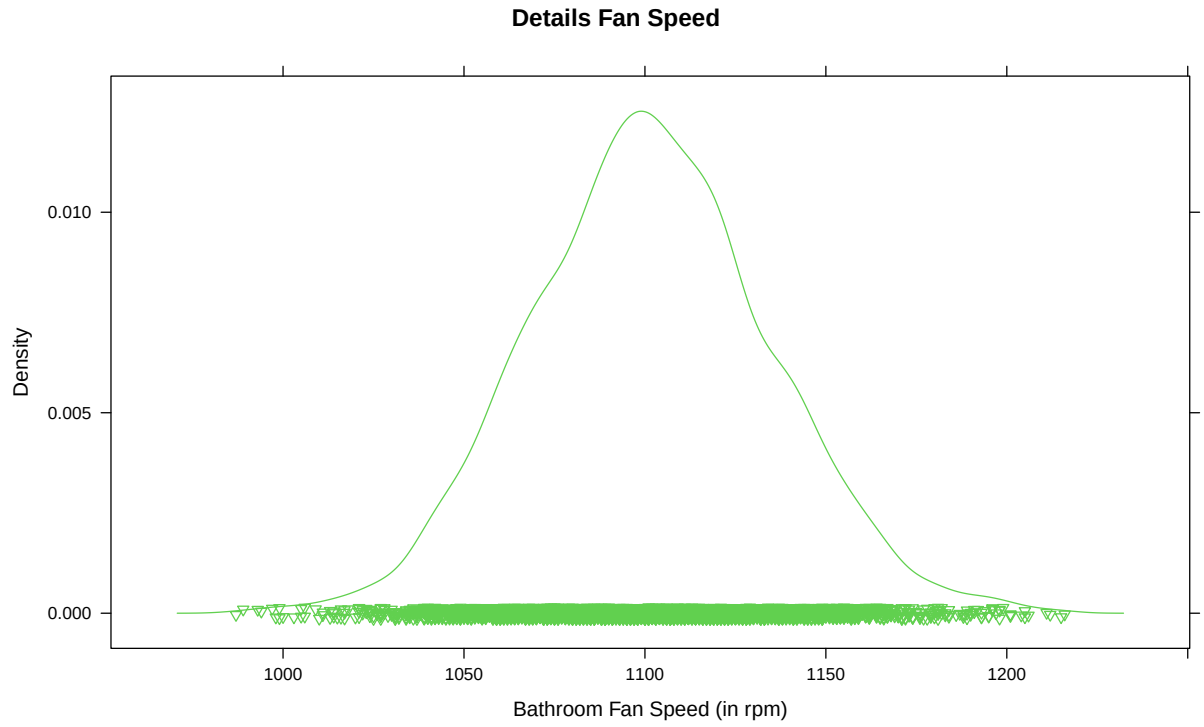
```
#Row number for max value
max_sensor2_pr <- which(sensor_tracking_PR$S2_T_PR == max(sensor_tracking_PR$S2_T_PR))
#Row number for min value
min_sensor2_pr <- which(sensor_tracking_PR$S2_T_PR == min(sensor_tracking_PR$S2_T_PR))
sensor_tracking_PR <- sensor_tracking_PR[-c(min_sensor2_pr,max_sensor2_pr),]
densityplot( ~ sensor_tracking_PR$S2_T_PR, pch=6,
             main='Details in Sensor #2 Temperature Data',
             xlab="Temp. (in C) Sensor 2",
             col=4)
```




```
#FUN SPEED
#Finding record with maximum value and deleting it
head(sensor_tracking_PR[rev(order(sensor_tracking_PR$FN_PR)),],1)
```

```
##      Index_PR Room_PR Ren_PR DT_PR TM_PR S1_L_PR S2_L_PR S3_L_PR S1_T_PR
## 2243     2243     111    New    18    14       1       1       1  22.882
##      S2_T_PR S3_T_PR FN_PR
## 2243  19.352  25.548  1230
```

```
max_fun_speed <- which(sensor_tracking_PR$FN_PR == max(sensor_tracking_PR$FN_PR))
#Row number for max value
sensor_tracking_PR <- sensor_tracking_PR[-c(max_fun_speed),]
densityplot( ~ sensor_tracking_PR$FN_PR, pch=6,
             main='Details Fan Speed',
             xlab="Bathroom Fan Speed (in rpm)",
             col=3)
```



3. Reduce Dimensionality

a. Drop any variables that do not contribute any useful analytical information at all.

```
head(sensor_tracking_PR,3)
```

```
##   Index_PR Room_PR Ren_PR DT_PR TM_PR S1_L_PR S2_L_PR S3_L_PR S1_T_PR S2_T_PR
## 1        1     111    New    47    21       1       1       1  24.479  21.651
## 2        2     111    New     3     2       1       1       1  23.771  24.167
## 3        3     311    Old    27    10       1       1       1  24.760  21.143
##   S3_T_PR FN_PR
## 1  22.227  1106
## 2  22.104  1146
## 3  22.881  1109
```

```
#Deleting index value
sensor_tracking_PR <- sensor_tracking_PR[-c(1)]
head(sensor_tracking_PR)
```

```
##   Room_PR Ren_PR DT_PR TM_PR S1_L_PR S2_L_PR S3_L_PR S1_T_PR S2_T_PR S3_T_PR
## 1     111    New    47    21       1       1       1  24.479  21.651  22.227
## 2     111    New     3     2       1       1       1  23.771  24.167  22.104
## 3     311    Old    27    10       1       1       1  24.760  21.143  22.881
## 4     211    New     5    22       1       1       1  22.130  23.925  19.353
## 5     311    Old    28    18       1       1       1  24.425  24.832  24.287
## 6     211    New     9    18       1       1       1  22.158  20.393  26.612
##   FN_PR
```

```
## 1  1106
## 2  1146
## 3  1109
## 4  1111
## 5  1120
## 6  1078
```

```r
#changing Ren_PR to create correlations
sensor_tracking_PR$Ren_PR <- ifelse(sensor_tracking_PR$Ren_PR == "New", 1, 0)

#execution time before reducing dimensions
start <- Sys.time()
cor(sensor_tracking_PR,method="spearman")
```

```
## Warning in cor(sensor_tracking_PR, method = "spearman"): the standard deviation
## is zero
```

```
##               Room_PR        Ren_PR         DT_PR         TM_PR      S1_L_PR
## Room_PR  1.0000000000 -0.8661257718 -0.0004425989 -0.0007794668 -0.020445214
## Ren_PR  -0.8661257718  1.0000000000  0.0003216643  0.0008997379  0.023925254
## DT_PR   -0.0004425989  0.0003216643  1.0000000000  0.0001578733 -0.011724886
## TM_PR   -0.0007794668  0.0008997379  0.0001578733  1.0000000000  0.015775444
## S1_L_PR -0.0204452135  0.0239252538 -0.0117248863  0.0157754442  1.000000000
## S2_L_PR -0.0204452135  0.0239252538 -0.0117248863  0.0157754442  1.000000000
## S3_L_PR            NA            NA            NA            NA           NA
## S1_T_PR  0.2452349658 -0.1366800812  0.0114076179 -0.0064325048  0.003936901
## S2_T_PR  0.2691304458 -0.1377214979  0.0047683367  0.0012234641 -0.006489933
## S3_T_PR -0.0011930483 -0.0002742646  0.0072274046 -0.0034831846  0.019605620
## FN_PR    0.0213185815 -0.0195912750 -0.0106962000  0.0138004454 -0.023488699
##              S2_L_PR S3_L_PR      S1_T_PR       S2_T_PR       S3_T_PR
## Room_PR -0.020445214      NA  0.245234966  0.2691304458 -0.0011930483
## Ren_PR   0.023925254      NA -0.136680081 -0.1377214979 -0.0002742646
## DT_PR   -0.011724886      NA  0.011407618  0.0047683367  0.0072274046
## TM_PR    0.015775444      NA -0.006432505  0.0012234641 -0.0034831846
## S1_L_PR  1.000000000      NA  0.003936901 -0.0064899335  0.0196056198
## S2_L_PR  1.000000000      NA  0.003936901 -0.0064899335  0.0196056198
## S3_L_PR           NA       1           NA            NA            NA
## S1_T_PR  0.003936901      NA  1.000000000  0.1034588144 -0.0144441353
## S2_T_PR -0.006489933      NA  0.103458814  1.0000000000  0.0007673131
## S3_T_PR  0.019605620      NA -0.014444135  0.0007673131  1.0000000000
## FN_PR   -0.023488699      NA  0.013321480  0.0040890407  0.0021598094
##                FN_PR
## Room_PR  0.021318582
## Ren_PR  -0.019591275
## DT_PR   -0.010696200
## TM_PR    0.013800445
## S1_L_PR -0.023488699
## S2_L_PR -0.023488699
## S3_L_PR           NA
## S1_T_PR  0.013321480
## S2_T_PR  0.004089041
## S3_T_PR  0.002159809
## FN_PR    1.000000000
```

```
end <- Sys.time()
s_time_pr <- end - start
s_time_pr
```

```
## Time difference of 0.02256417 secs
```

I decided to remove the index_PR is used as row identifiers, it does not contribute any useful analitycal information. I converted the room condition variable (Ren_PR) to Boolean type for the purpose of correlation analysis.

b. Apply the Missing Value Filter to remove appropriate columns of data.

```
# Identify columns > 99% missing with summary function
summary(sensor_tracking_PR)
```

```
##     Room_PR        Ren_PR           DT_PR          TM_PR         S1_L_PR
##  Min.   :111   Min.   :0.0000   Min.   : 1.0   Min.   : 1.0   Min.   :0.0000
##  1st Qu.:111   1st Qu.:0.0000   1st Qu.:15.5   1st Qu.: 7.0   1st Qu.:0.0000
##  Median :211   Median :1.0000   Median :30.0   Median :13.0   Median :1.0000
##  Mean   :211   Mean   :0.6665   Mean   :30.5   Mean   :12.5   Mean   :0.5034
##  3rd Qu.:311   3rd Qu.:1.0000   3rd Qu.:45.5   3rd Qu.:18.5   3rd Qu.:1.0000
##  Max.   :311   Max.   :1.0000   Max.   :60.0   Max.   :24.0   Max.   :1.0000
##     S2_L_PR          S3_L_PR     S1_T_PR         S2_T_PR         S3_T_PR
##  Min.   :0.0000   Min.   :1   Min.   :14.93   Min.   :14.98   Min.   :16.26
##  1st Qu.:0.0000   1st Qu.:1   1st Qu.:21.07   1st Qu.:21.03   1st Qu.:21.62
##  Median :1.0000   Median :1   Median :22.51   Median :22.53   Median :22.98
##  Mean   :0.5034   Mean   :1   Mean   :22.48   Mean   :22.47   Mean   :22.99
##  3rd Qu.:1.0000   3rd Qu.:1   3rd Qu.:23.95   3rd Qu.:23.92   3rd Qu.:24.35
##  Max.   :1.0000   Max.   :1   Max.   :29.26   Max.   :29.72   Max.   :30.45
##      FN_PR
##  Min.   : 987
##  1st Qu.:1079
##  Median :1101
##  Mean   :1101
##  3rd Qu.:1122
##  Max.   :1216
```

In the data set there are no nulls values

c. Apply the Low Variance Filter to remove appropriate columns of data.

```
# Identify columns with low variance
stat.desc(sensor_tracking_PR)
```

```
##                     Room_PR         Ren_PR          DT_PR         TM_PR
## nbr.val        4315.0000000 4315.000000000   4315.0000000  4315.0000000
## nbr.null          0.0000000 1439.000000000      0.0000000     0.0000000
## nbr.na            0.0000000    0.000000000      0.0000000     0.0000000
## min             111.0000000    0.000000000      1.0000000     1.0000000
## max             311.0000000    1.000000000     60.0000000    24.0000000
```

```
## range           200.0000000    1.000000000     59.0000000    23.0000000
## sum          910665.0000000 2876.000000000 131604.0000000 53944.0000000
## median          211.0000000    1.000000000     30.0000000    13.0000000
## mean            211.0463499    0.666512167     30.4991889    12.5015064
## SE.mean           1.2429793    0.007178008      0.2637249     0.1053818
## CI.mean.0.95      2.4368784    0.014072586      0.5170365     0.2066026
## var            6666.6645179    0.222325222    300.1119024    47.9195040
## std.dev          81.6496449    0.471513756     17.3237381     6.9223915
## coef.var          0.3868802    0.707434582      0.5680065     0.5537246
##                     S1_L_PR        S2_L_PR S3_L_PR        S1_T_PR
## nbr.val      4315.000000000 4315.000000000    4315  4315.00000000
## nbr.null     2143.000000000 2143.000000000       0     0.00000000
## nbr.na          0.000000000    0.000000000       0     0.00000000
## min             0.000000000    0.000000000       1    14.92900000
## max             1.000000000    1.000000000       1    29.25800000
## range           1.000000000    1.000000000       0    14.32900000
## sum          2172.000000000 2172.000000000    4315 96991.33800000
## median          1.000000000    1.000000000       1    22.51400000
## mean            0.503360371    0.503360371       1    22.47771448
## SE.mean         0.007612374    0.007612374       0     0.03195776
## CI.mean.0.95    0.014924166    0.014924166       0     0.06265364
## var             0.250046656    0.250046656       0     4.40690333
## std.dev         0.500046654    0.500046654       0     2.09926257
## coef.var        0.993416810    0.993416810       0     0.09339306
##                     S2_T_PR        S3_T_PR           FN_PR
## nbr.val       4315.00000000  4315.00000000    4315.0000000
## nbr.null         0.00000000     0.00000000       0.0000000
## nbr.na           0.00000000     0.00000000       0.0000000
## min             14.98300000    16.25800000     987.0000000
## max             29.72300000    30.45000000    1216.0000000
## range           14.74000000    14.19200000     229.0000000
## sum          96950.72900000 99212.38800000 4752068.0000000
## median          22.52600000    22.98400000    1101.0000000
## mean            22.46830336    22.99244218    1101.2903824
## SE.mean          0.03234368     0.03033013       0.5072191
## CI.mean.0.95     0.06341023     0.05946265       0.9944102
## var              4.51397958     3.96944090    1110.1254406
## std.dev          2.12461281     1.99234558      33.3185450
## coef.var         0.09456045     0.08665219       0.0302541
```

```
#Deleting 7 column (S3_L_PR) without variance
sensor_tracking_PR <- sensor_tracking_PR[-c(7)]
head(sensor_tracking_PR)
```

```
##   Room_PR Ren_PR DT_PR TM_PR S1_L_PR S2_L_PR S1_T_PR S2_T_PR S3_T_PR FN_PR
## 1     111      1    47    21       1       1  24.479  21.651  22.227  1106
## 2     111      1     3     2       1       1  23.771  24.167  22.104  1146
## 3     311      0    27    10       1       1  24.760  21.143  22.881  1109
## 4     211      1     5    22       1       1  22.130  23.925  19.353  1111
## 5     311      0    28    18       1       1  24.425  24.832  24.287  1120
## 6     211      1     9    18       1       1  22.158  20.393  26.612  1078
```

The sensor 3 did not detect changes in the light, this dimension is not necessary

d. Apply the High Correlation Filter to remove appropriate columns of data.

```
#isnumeric,ask to profesor unlist someethint
# Identify high correlation filter
cor(sensor_tracking_PR,method="spearman")
```

```
##               Room_PR        Ren_PR         DT_PR         TM_PR      S1_L_PR
## Room_PR  1.0000000000 -0.8661257718 -0.0004425989 -0.0007794668 -0.020445214
## Ren_PR  -0.8661257718  1.0000000000  0.0003216643  0.0008997379  0.023925254
## DT_PR   -0.0004425989  0.0003216643  1.0000000000  0.0001578733 -0.011724886
## TM_PR   -0.0007794668  0.0008997379  0.0001578733  1.0000000000  0.015775444
## S1_L_PR -0.0204452135  0.0239252538 -0.0117248863  0.0157754442  1.000000000
## S2_L_PR -0.0204452135  0.0239252538 -0.0117248863  0.0157754442  1.000000000
## S1_T_PR  0.2452349658 -0.1366800812  0.0114076179 -0.0064325048  0.003936901
## S2_T_PR  0.2691304458 -0.1377214979  0.0047683367  0.0012234641 -0.006489933
## S3_T_PR -0.0011930483 -0.0002742646  0.0072274046 -0.0034831846  0.019605620
## FN_PR    0.0213185815 -0.0195912750 -0.0106962000  0.0138004454 -0.023488699
##              S2_L_PR      S1_T_PR       S2_T_PR       S3_T_PR        FN_PR
## Room_PR -0.020445214  0.245234966  0.2691304458 -0.0011930483  0.021318582
## Ren_PR   0.023925254 -0.136680081 -0.1377214979 -0.0002742646 -0.019591275
## DT_PR   -0.011724886  0.011407618  0.0047683367  0.0072274046 -0.010696200
## TM_PR    0.015775444 -0.006432505  0.0012234641 -0.0034831846  0.013800445
## S1_L_PR  1.000000000  0.003936901 -0.0064899335  0.0196056198 -0.023488699
## S2_L_PR  1.000000000  0.003936901 -0.0064899335  0.0196056198 -0.023488699
## S1_T_PR  0.003936901  1.000000000  0.1034588144 -0.0144441353  0.013321480
## S2_T_PR -0.006489933  0.103458814  1.0000000000  0.0007673131  0.004089041
## S3_T_PR  0.019605620 -0.014444135  0.0007673131  1.0000000000  0.002159809
## FN_PR   -0.023488699  0.013321480  0.0040890407  0.0021598094  1.000000000
```

```
#cor(sensor_tracking_PR,method="pearson")
#Deleting 5 column (S2_L_PR) high relationship with S1_L_PR
sensor_tracking_PR <- sensor_tracking_PR[-c(6)]
head(sensor_tracking_PR)
```

```
##   Room_PR Ren_PR DT_PR TM_PR S1_L_PR S1_T_PR S2_T_PR S3_T_PR FN_PR
## 1     111      1    47    21       1  24.479  21.651  22.227  1106
## 2     111      1     3     2       1  23.771  24.167  22.104  1146
## 3     311      0    27    10       1  24.760  21.143  22.881  1109
## 4     211      1     5    22       1  22.130  23.925  19.353  1111
## 5     311      0    28    18       1  24.425  24.832  24.287  1120
## 6     211      1     9    18       1  22.158  20.393  26.612  1078
```

I have eliminated the dimension *S2_L_PR* due to it is highly correlated with *S1_L_PR*. It is not necessary to keep both.

e. Based on our discussions in class, what are some specific benefits of reducing the dimensionality of this particular dataset? Be specific. For example, if it increases computational efficiency, specify how much of an improvement.

```
#execution time after reducing dimensions
start <- Sys.time()
cor(sensor_tracking_PR,method="spearman")
```

```
##                Room_PR        Ren_PR         DT_PR         TM_PR      S1_L_PR
## Room_PR  1.0000000000 -0.8661257718 -0.0004425989 -0.0007794668 -0.020445214
## Ren_PR  -0.8661257718  1.0000000000  0.0003216643  0.0008997379  0.023925254
## DT_PR   -0.0004425989  0.0003216643  1.0000000000  0.0001578733 -0.011724886
## TM_PR   -0.0007794668  0.0008997379  0.0001578733  1.0000000000  0.015775444
## S1_L_PR -0.0204452135  0.0239252538 -0.0117248863  0.0157754442  1.000000000
## S1_T_PR  0.2452349658 -0.1366800812  0.0114076179 -0.0064325048  0.003936901
## S2_T_PR  0.2691304458 -0.1377214979  0.0047683367  0.0012234641 -0.006489933
## S3_T_PR -0.0011930483 -0.0002742646  0.0072274046 -0.0034831846  0.019605620
## FN_PR    0.0213185815 -0.0195912750 -0.0106962000  0.0138004454 -0.023488699
##              S1_T_PR       S2_T_PR       S3_T_PR        FN_PR
## Room_PR  0.245234966  0.2691304458 -0.0011930483  0.021318582
## Ren_PR  -0.136680081 -0.1377214979 -0.0002742646 -0.019591275
## DT_PR    0.011407618  0.0047683367  0.0072274046 -0.010696200
## TM_PR   -0.006432505  0.0012234641 -0.0034831846  0.013800445
## S1_L_PR  0.003936901 -0.0064899335  0.0196056198 -0.023488699
## S1_T_PR  1.000000000  0.1034588144 -0.0144441353  0.013321480
## S2_T_PR  0.103458814  1.0000000000  0.0007673131  0.004089041
## S3_T_PR -0.014444135  0.0007673131  1.0000000000  0.002159809
## FN_PR    0.013321480  0.0040890407  0.0021598094  1.000000000
```

```
end <- Sys.time()
e_time_pr <- end - start

#Difference executing correlation function before and after reduction
improvement_pr <-- s_time_pr - e_time_pr
improvement_pr
```

```
## Time difference of -0.04209423 secs
```

In this dataset I have eliminated the index column that were not relevant for the analysis. I have applied two effective reductions methods, low variance filter and High Correlation Filter, which means that it also did not provide relevance for the analysis.

Even though there is not significant data, I have compared the correlation processing times before and after the analysis, showing a slight improvement in the times. (see *improvement_pr* variable)

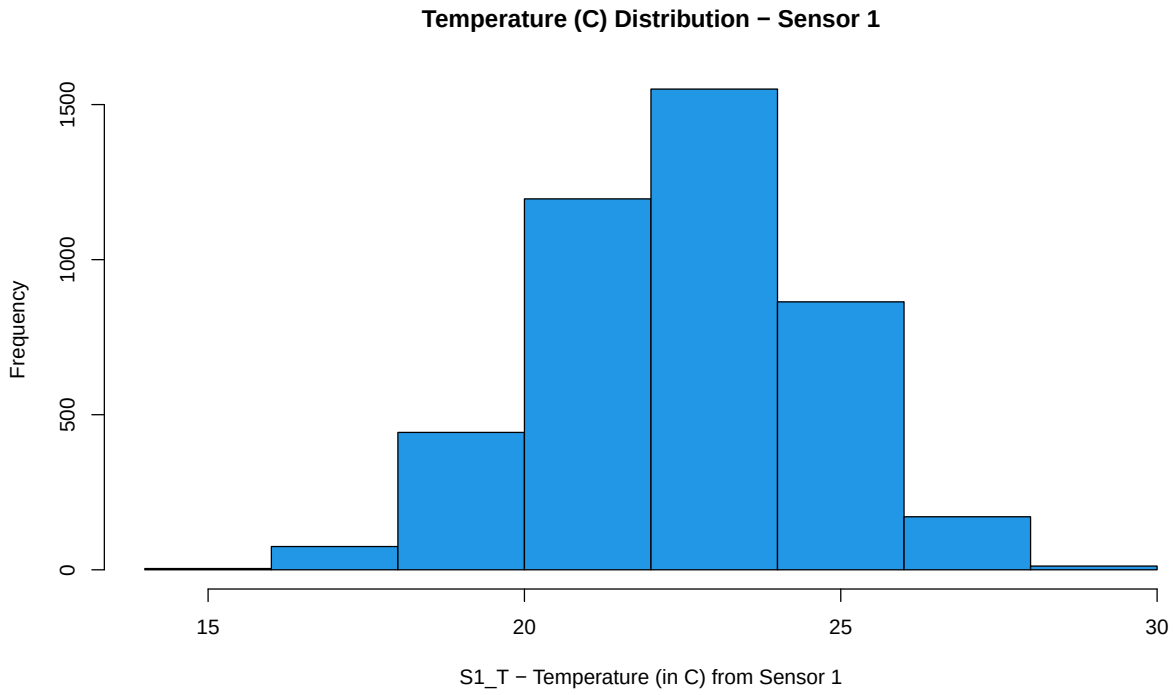
With fewer features or variables, the analysis could improve overall performance. Additionally, it allows for better visualization of the data to create best visualizations analysis.

2. Organizing Data

a. Create a histogram for Temperature from Sensor 1.

```
#Histogram temperature sensor 1
hist(sensor_tracking_PR$S1_T_PR,
     col=4,
     breaks = 10,
     main="Temperature (C) Distribution - Sensor 1",
     xlab="S1_T - Temperature (in C) from Sensor 1",
     ylab="Frequency",
     pch=15
```

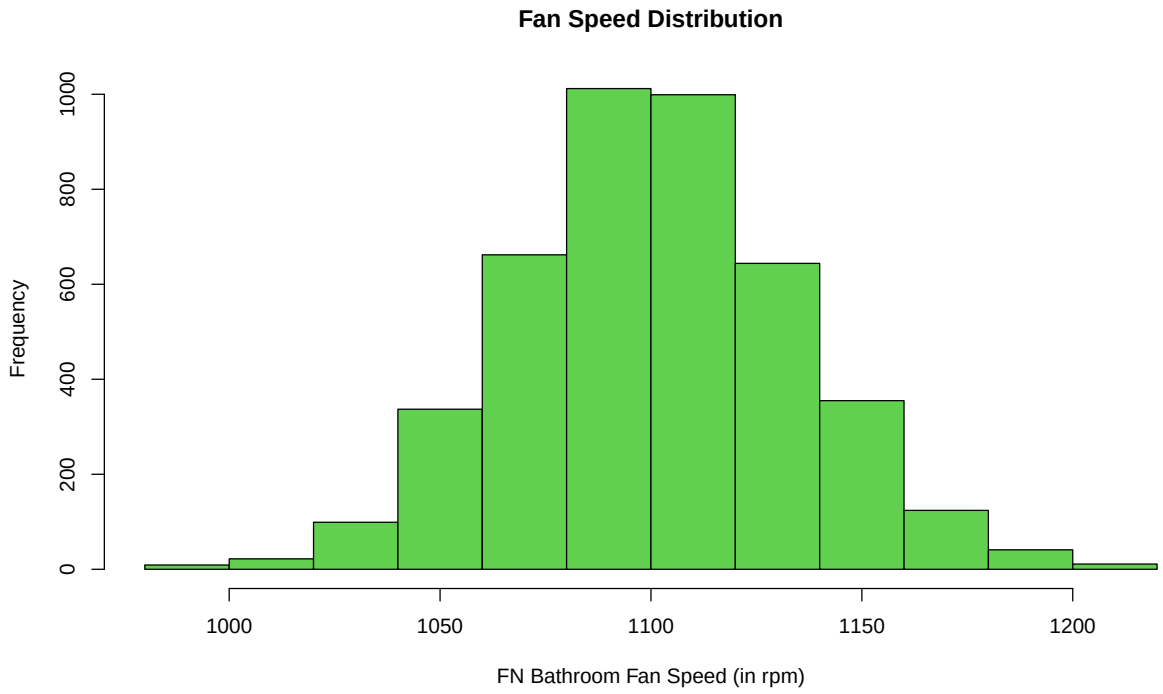
```
    )
```



The histogram shows that the majority of records for the sensor number 1 are concentrated in the temperature range of 20 to 26 C degrees.

b. Create a histogram for Fan Speed.

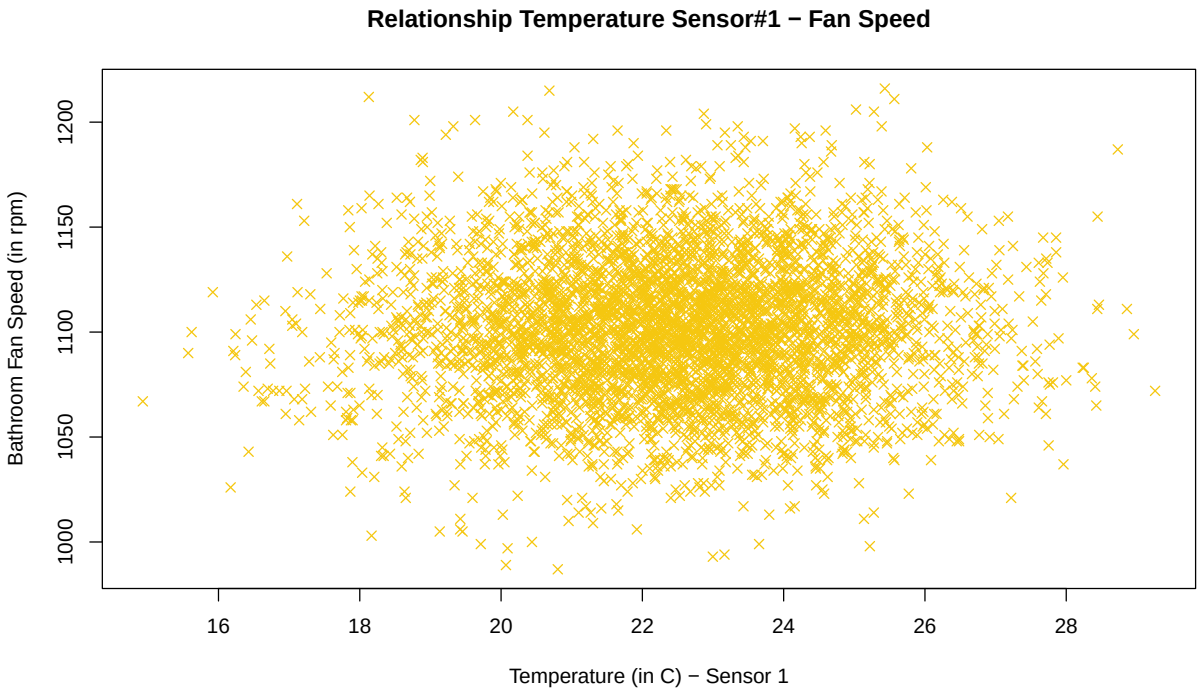
```
#Histogram fan speed in rpm
hist(sensor_tracking_PR$FN_PR,
     col=3,
     main="Fan Speed Distribution",
     xlab="FN Bathroom Fan Speed (in rpm)",
     ylab="Frequency"
    )
```



The histogram revealed that the most common fan speed range is between ~1075rpm and ~1125rpm. There are little extreme values.

c. Create a scatter plot showing the relationship between Temperature Sensor 1 and Fan Speed. (note: Sensor 1 should be on the x-axis, Fan Speed should be the y-axis)

```
#Creating Scatter plot
plot(FN_PR ~ S1_T_PR,
     data=sensor_tracking_PR,
     col=7,
     pch=4,
     main="Relationship Temperature Sensor#1 - Fan Speed",
     xlab="Temperature (in C) - Sensor 1",
     ylab="Bathroom Fan Speed (in rpm)")
```



The markets are dispersed, which means that there are not a clear lineal relationship between those variables.

d. What conclusions, if any, can you draw from the chart?

The fan speed of the bathrooms is not correlated with the temperature (there are not a clear lineal relationship). However the chart shows that both, fan speed and sensor 1's temperature remain within consistent ranges, each with its own range of variation and unit of measurement.

e. Calculate a correlation coefficient between these two variables. Why did you choose the correlation coefficient you did? What conclusion you draw from it?

```
#Calculating correlation coefficient
print("Spearman Correlation")
```

```
## [1] "Spearman Correlation"
```

```
#Pearson defaults, but assumes normality
round(cor(sensor_tracking_PR$FN_PR, sensor_tracking_PR$S1_T_PR, method="spearman"),3)
```

```
## [1] 0.013
```

Based on my notes and the article of 'Correlation Coefficients' (Schober et al., 2018), I could not use the default method Pearson for this specific data set, because it assumes normality and a linear relationship. Meanwhile, Spearman correlation is better due to the significant variability in the observations.

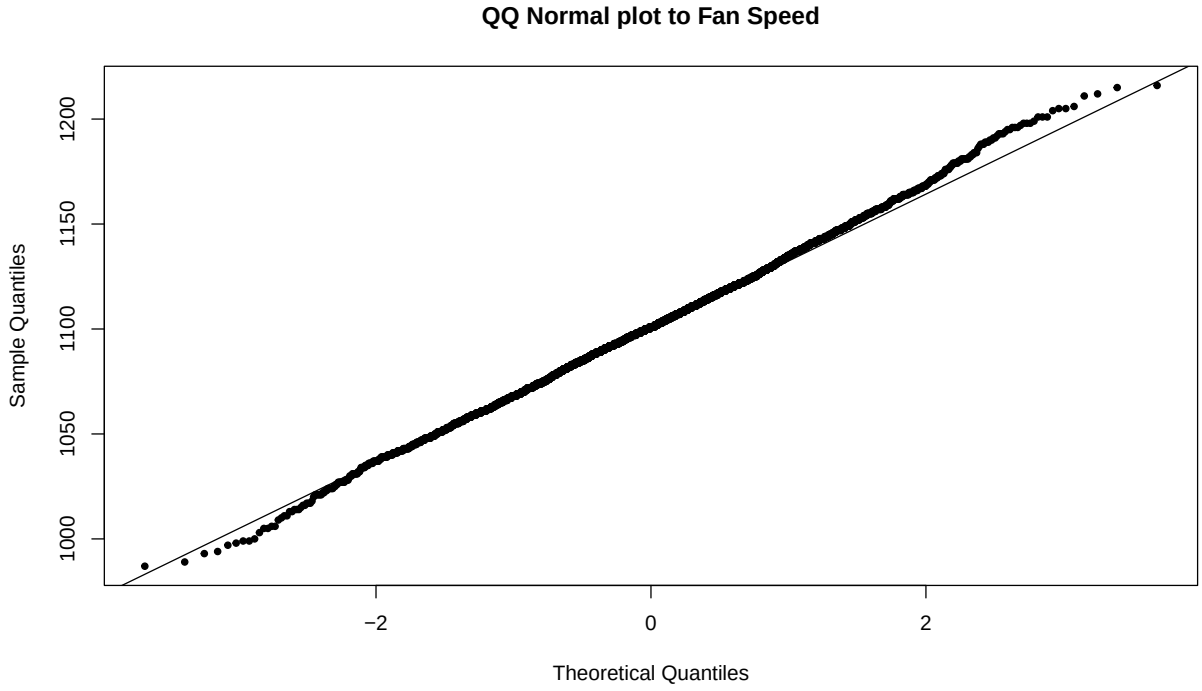
The correlation coefficient was of 0.013, which means that the relationship is significantly weak.

3. Inference

1. Normality

a. Create a QQ Normal plot of for Fan Speed.

```
#Checking normal distribution
qqnorm(sensor_tracking_PR$FN_PR, main="QQ Normal plot to Fan Speed", pch=20)
qqline(sensor_tracking_PR$FN_PR)
```



The majority of the points fall along the diagonal line; however, a deviation from the straight line can be observed at the tails, indicanting that distribution is not normal.

b. Conduct a statistical test for normality on Fan Speed.

```
#Shapiro test
shapiro.test(sensor_tracking_PR$FN_PR)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  sensor_tracking_PR$FN_PR
## W = 0.99901, p-value = 0.01302
```

I did a Shapiro test to validate if the data is normal distributed.

c. Is Fan Speed normally distributed? What led you to this conclusion?

Fan speed is not normally distributed, I have executed the Shapiro Test, and the result was 0.013. The hypothesis that the data is normally distributed must be rejected due to the p-value is less than 0.05.

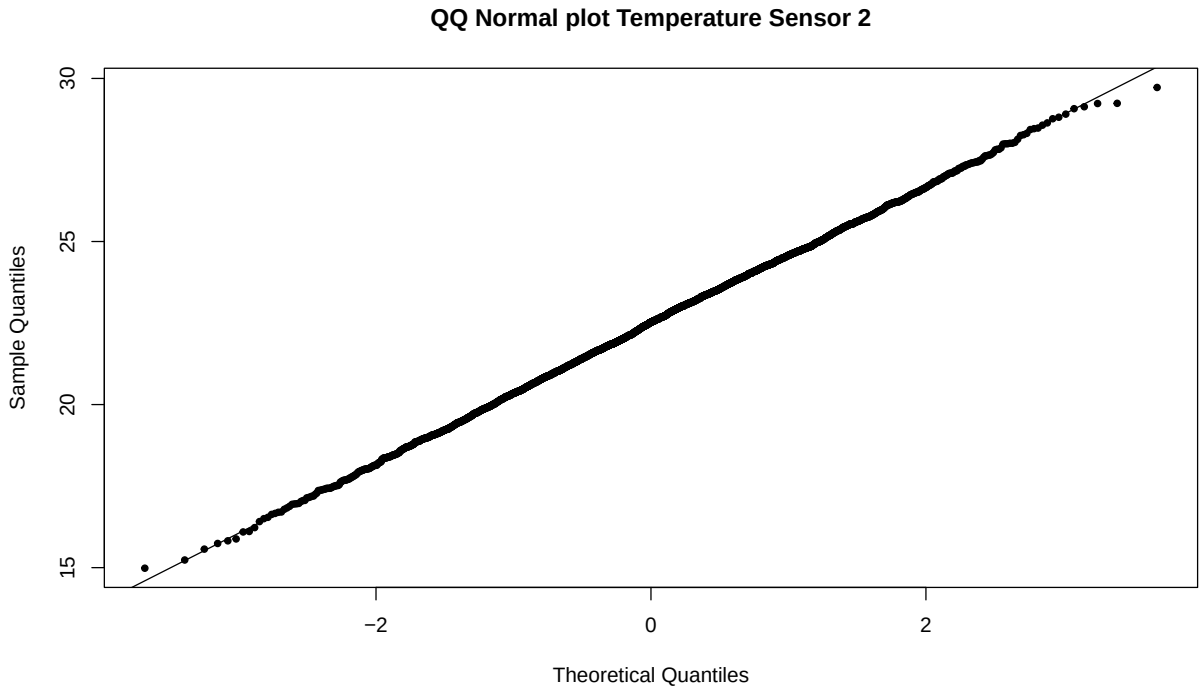
2. Statistically Significant Differences

a. Compare Temperature from Sensor 2 between 'New' and 'Old' rooms in your dataset using a suitable hypothesis test.

```
#Returning Ren_PR to factor
sensor_tracking_PR$Ren_PR <- ifelse(sensor_tracking_PR$Ren_PR == 1, "New", "Old")
sensor_tracking_PR$Ren_PR <- as.factor(sensor_tracking_PR$Ren_PR)
#Shapiro test
shapiro.test(sensor_tracking_PR$S2_T_PR)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  sensor_tracking_PR$S2_T_PR
## W = 0.99963, p-value = 0.6232
```

```
#Checking normal distribution
qqnorm(sensor_tracking_PR$S2_T_PR, main="QQ Normal plot Temperature Sensor 2", pch=20)
qqline(sensor_tracking_PR$S2_T_PR)
```




```
#Comparing Variance F-Test
var.test(S2_T_PR ~ Ren_PR, data=sensor_tracking_PR)
```

```
##
##  F test to compare two variances
##
## data:  S2_T_PR by Ren_PR
```

```
## F = 1.15, num df = 2875, denom df = 1438, p-value = 0.002454
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  1.050741 1.256848
## sample estimates:
## ratio of variances
##           1.149962
```

```
#Comparing Variance F-Test
wilcox.test(S2_T_PR ~ Ren_PR, data=sensor_tracking_PR)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  S2_T_PR by Ren_PR
## W = 1720289, p-value < 2.2e-16
## alternative hypothesis: true location shift is not equal to 0
```

b. Explain why you chose the test you did.

Firstly, I tried with T-Test, my results for each assumption was:

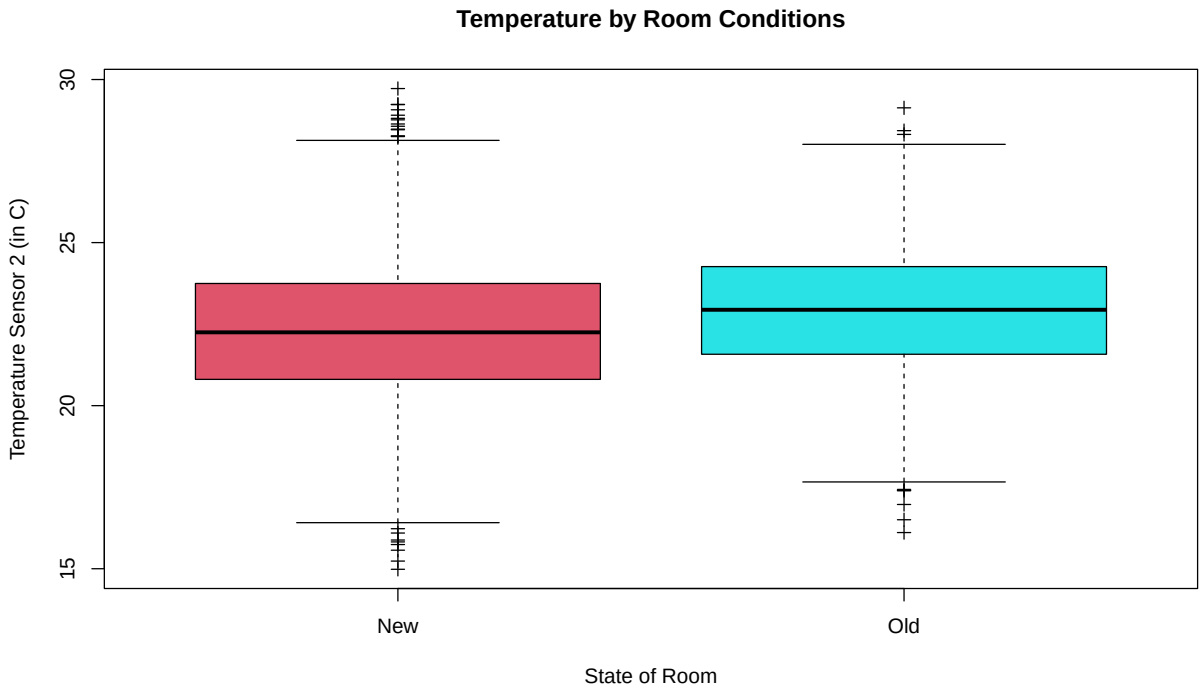
1. Data are independent –> PASS
2. Data is normal distributed. The *S2_T_PR* passed the Shapiro Test with p-value = 0.6232 and QQ Normal plot shows a straight line. PASS
3. F-Test –> FAIL. p-value = 0.002454 < 0.05

I could not use T-Test because Temperature Sensor 2 violates normality assumptions. For that reason my final test was Wilcoxon test.

c. Do you have strong evidence that Temperature from Sensor 2 is different between new and old rooms?

The result for Wilcoxon test was p-value < 2.2e-16, that means that there is significant difference in means between new and old rooms.

```
# Difference means between 'New' and 'Old' Room
boxplot( S2_T_PR ~ Ren_PR, data=sensor_tracking_PR, main="Temperature by Room Conditions",
                xlab="State of Room",color=2, ylab="Temperature Sensor 2 (in C)",pch=3, col=c(2,5))
```

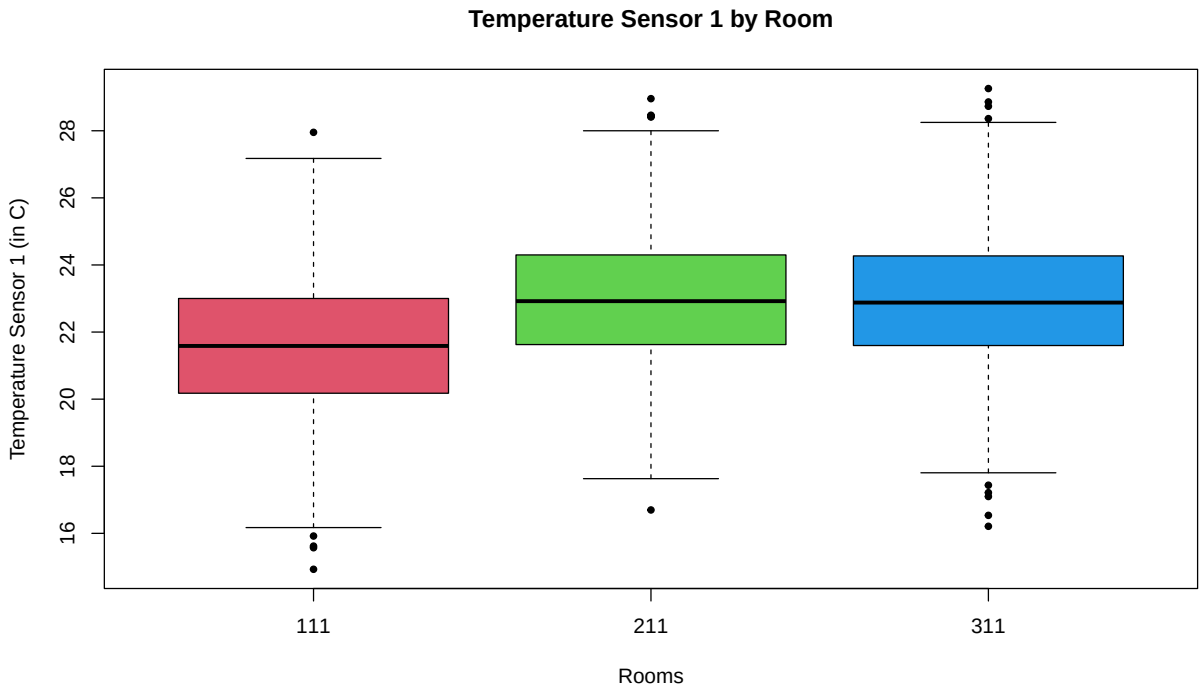


Based onn the box plot and Wilcoxon test, I can conclude that new rooms are slightly colder that old rooms. The median temperature in old rooms is generally higher than in new rooms

3. Multiple Statistical Differences

a. Determine if Temperature from Sensor 1 varies by Room Number using ANOVA (statistical) and a sequence of boxplots (graphical).

```
# Comparing Sensor 1 per Rooms
boxplot( S1_T_PR ~ Room_PR, data=sensor_tracking_PR, main="Temperature Sensor 1 by Room",
                xlab="Rooms",color=2, ylab="Temperature Sensor 1 (in C)",pch=20, col=c(2,3,4)
         )
```



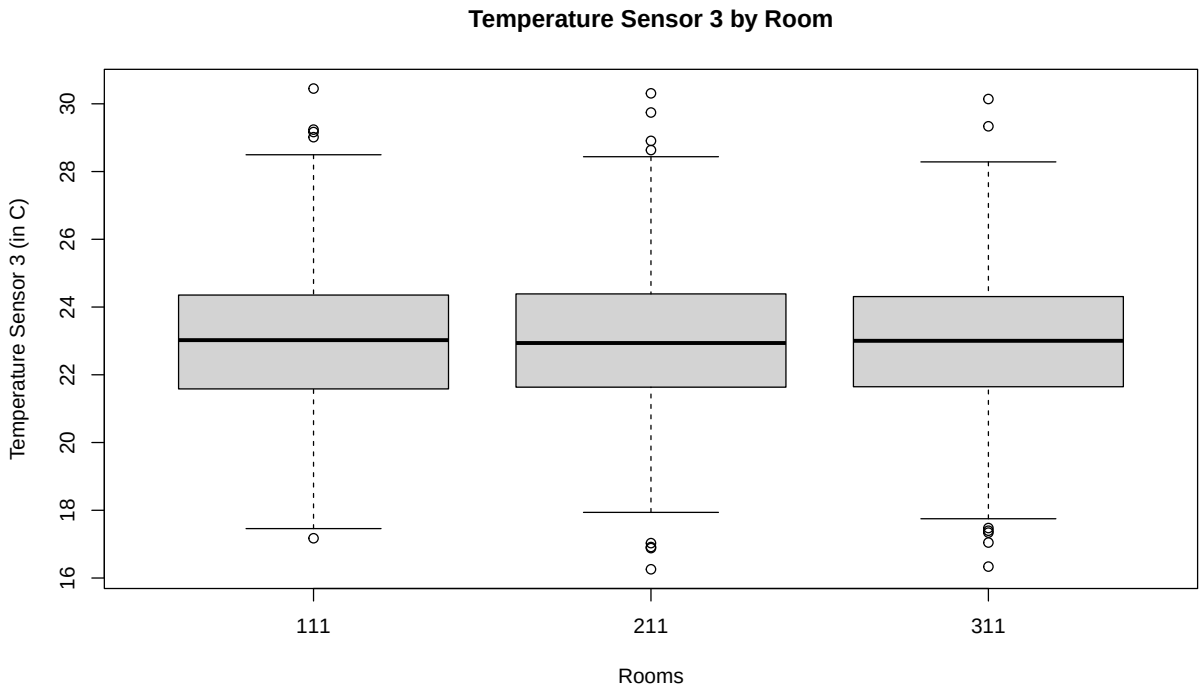

```
summary(aov(S1_T_PR ~ Room_PR, data=sensor_tracking_PR))
```

```
##               Df Sum Sq Mean Sq F value Pr(>F)
## Room_PR        1   1221  1220.8     296 <2e-16 ***
## Residuals   4313  17791     4.1
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Based on box plot, there is a visible variation in the medians in the room 111. However the dispersion between rooms are similar. The p-value is so small (<2e-16) indicates that there are significant temperature differences between rooms.

b. Determine if Temperature from Sensor 3 varies by Room Number using ANOVA (statistical) and a sequence of boxplots (graphical).

```
# Comparing Sensor 3 per Rooms
boxplot( S3_T_PR ~ Room_PR, data=sensor_tracking_PR, main="Temperature Sensor 3 by Room",
                xlab="Rooms",color=2, ylab="Temperature Sensor 3 (in C)"
         )
```




```
summary(aov(S3_T_PR ~ Room_PR, data=sensor_tracking_PR))
```

```
##               Df Sum Sq Mean Sq F value Pr(>F)
## Room_PR        1      0   0.029   0.007  0.932
## Residuals   4313  17124   3.970
```

Based on box plot, there are not visible variations in the medians between rooms, the dispersion between rooms are similar across rooms as well. The p-value is 0.9 confirms that there not significant differences in temperature variations between rooms for Sensor 3, wich means the temperature variance is almost null.

References

unique function - RDocumentation. (n.d.). https://www.rdocumentation.org/packages/base/versions/3.6.2/topics/unique

match function - RDocumentation. (n.d.). https://www.rdocumentation.org/packages/base/versions/3.6.2/topics/match

Schober, P., Boer, C., & Schwarte, L. A. (2018). Correlation Coefficients: appropriate use and interpretation. Anesthesia & Analgesia, 126(5), 1763–1768. https://doi.org/10.1213/ane.0000000000002864