Are there people in the office?

A work by F. Cabras, A. Donvito

June 19, 2019

To make the world a better place, mankind should not only care about the way energy is produced but also, and especially, about how it is consumed. In the path to sustainability, the priority is to avoid any type of waste. If fifty years ago such optimization was somehow difficult to be obtained, now the affordability and the capabilities of sensors are making it within reach. The aim of the authors is to demonstrate how easy it is to predict the presence of individuals in a room, specifically an office. This can help in a variety of ways: for instance the System could decide to automatically turn heaters and lights off. The features of the dataset, downloadable at the webpage `archive.ics.uci.edu/ml/datasets/Occupancy+Detection+`, are the following:

- *Temperature*, measured in Celsius;
- *Relative humidity*, in percentage terms;
- *Light*, measured in lux;
- *CO2*, measured in ppm;
- *Humidity ratio*, a quantity derived from temperature and relative humidity and measured in kgwater-vapor/kg-air;
- *Room Occupancy*, 0 for not occupied, 1 for occupied status.

Observations are taken at the rate of one per minute and for three periods of time. The variables are the same, and units of measurement do not change. The occupancy-non occupancy label is always provided. The first time-span goes from February, the 2nd at 14:19, to February, the 4th at 10:43. The second time-span goes from the latter day at 17:51 to February, the 10th at 9:33. Once the datasets are downloaded, we see how they are already given a name, and those first two are labeled as test and training, respectively. The third time-span goes from February, the 11th at 14:48 to February, the 18th, at 9:19. Also this last dataset is labeled as test, so we are going to use it together with the first for validating our results. The underlying reasons for such a timing repartition are not specified. To not work with two separate validation datasets, we have simply decided to merge them. The resulting training and validation dataframes gather a total of 8143 and 12417 observations, respectively.

```
> train <- read.csv("datatraining.txt", header=T)
> test <- rbind(read.csv("datatest.txt", header=T), read.csv("datatest2.txt"))
```

1 Extracting Information

Let us now concentrate on the variables at our disposal. All of them are numeric, with the notable exception of the date, which is classified as a factor. To better make use of such a feature what we do is a class transformation into a POSIXct type, a class for time representation in the R environment. Also, the column occupancy is classified as numeric but we are interested in using it as a factor. Of course, such transformations are executed on both the train and the test set (the complete code shall be found in the .Rnw file).

```
> train$Occupancy <- as.factor(train$Occupancy)
> train$date <- as.POSIXct(train$date, format="%Y-%m-%d %H:%M:%S")
> str(train)

'data.frame':        8143 obs. of  7 variables:
 $ date         : POSIXct, format: "2015-02-04 17:51:00" ...
 $ Temperature  : num  23.2 23.1 23.1 23.1 23.1 ...
 $ Humidity     : num  27.3 27.3 27.2 27.2 27.2 ...
 $ Light        : num  426 430 426 426 426 ...
 $ CO2          : num  721 714 714 708 704 ...
 $ HumidityRatio: num  0.00479 0.00478 0.00478 0.00477 0.00476 ...
 $ Occupancy    : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
```

A question comes up to our minds: is it really interesting to consider in our work both the variable *Humidity* and the variable *HumidityRatio*? After all, from the description of the dataset, it seems that the latter is simply derived from the temperature and the humidity level. Hence, we test the hypothesis that humidity ratio is linearly derived from those two.

```
> summary(lm(HumidityRatio ~ Humidity + Temperature, train))$adj.r.squared

[1] 0.9965603
```

The variability of the humidity ratio is explained at the 99.7% by the behaviour of the humidity and temperature, which means that the linear relationship is evident. To avoid dealing with collinear variables, the most convenient way is to not consider the humidity ratio when building models. Another useful perspective on our data is given by the command *summary.*

```
> summary(train)

      date                         Temperature   
 Min.   :2015-02-04 17:51:00   Min.   :19.00  
 1st Qu.:2015-02-06 03:46:30   1st Qu.:19.70  
 Median :2015-02-07 13:41:59   Median :20.39  
 Mean   :2015-02-07 13:41:59   Mean   :20.62  
 3rd Qu.:2015-02-08 23:37:30   3rd Qu.:21.39  
 Max.   :2015-02-10 09:33:00   Max.   :23.18  
    Humidity         Light             CO2        
 Min.   :16.75   Min.   :   0.0   Min.   : 412.8  
 1st Qu.:20.20   1st Qu.:   0.0   1st Qu.: 439.0  
 Median :26.22   Median :   0.0   Median : 453.5  
 Mean   :25.73   Mean   : 119.5   Mean   : 606.5  
 3rd Qu.:30.53   3rd Qu.: 256.4   3rd Qu.: 638.8  
 Max.   :39.12   Max.   :1546.3   Max.   :2028.5  
 HumidityRatio       Occupancy
 Min.   :0.002674   0:6414   
 1st Qu.:0.003078   1:1729   
```

```
Median :0.003801
Mean   :0.003863
3rd Qu.:0.004352
Max.   :0.006476
```

What we find out is that our data is quite clean already: there are no NAs or absurd measurements for the variables. Only some light records are atypical: we find out that on Saturday morning, hence when the office is empty,surprisingly high values are reached, over 1000! How to explain such a behaviour? Probably, natural light had an effect: it is possible that a sun glare hit the sensor. We opt for a substitution of these outliers with the mean value of light at 9 when the room is empty.

```
> empty <- subset(train, Occupancy==0)
> train[c(3832:3834),]$Light <- mean(empty[as.POSIXlt(empty$date)$h==9,]$Light)
```

Focusing instead on our response variable, we notice that there is an imbalance between the occupancy-non occupancy classes: 6414 against 1729 observations. Let us now focus on the relationship between *Occupancy* and the environmental conditions.

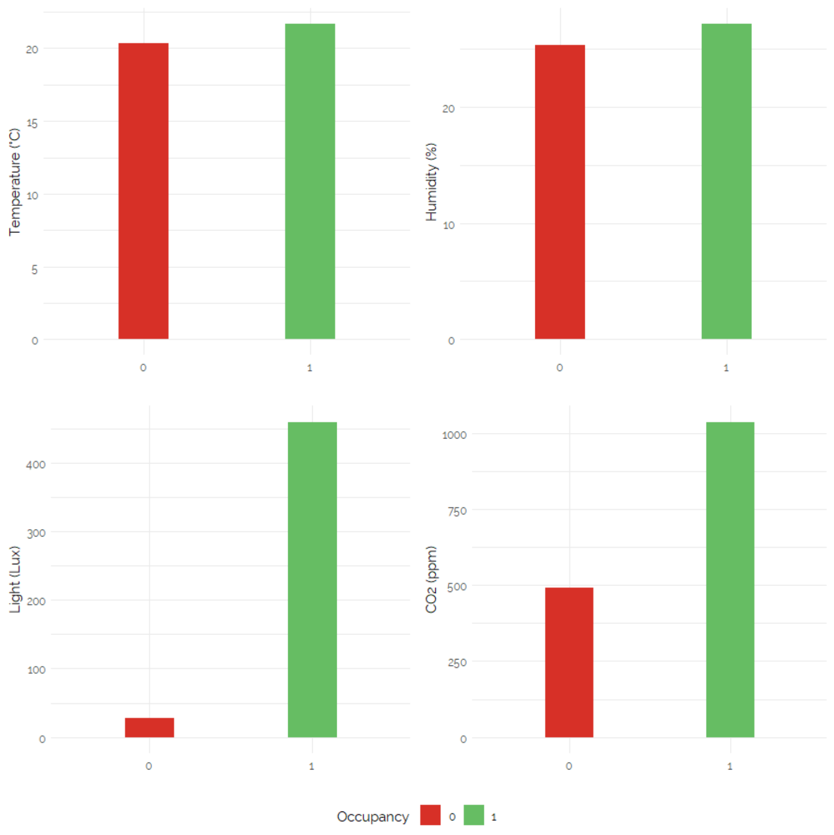

Figure 1: Mean values for environmental conditions.

When people are in the office, indicators of temperature, humidity, light and CO2 tend to be higher than when the office is empty. **Figure 2** shows the behaviour of the environmental variables during the five days considered.

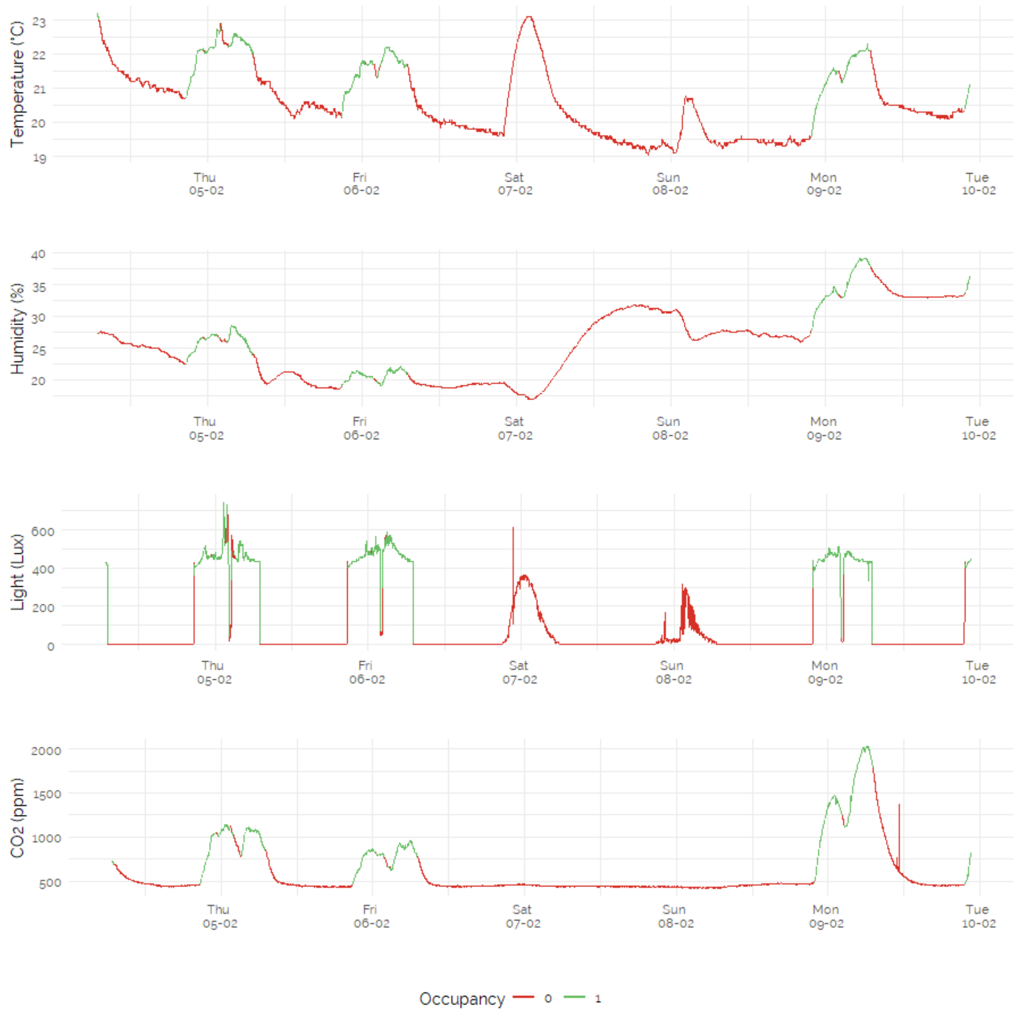

Figure 2: Environmental conditions over time.

After the adjustments executed on those unusual light observations, all curves seem to be quite continuous: no significant jumps are present.
In all these reasonings we have temporarily forgotten about the datetime value. If untouched this feature is quite pointless, but through feature extraction it can become useful for a variety of reasons. One interesting idea is to extract a logical variable *Day*, that assigns **True** to observations recorded between 8 and 20, and **False** otherwise. This can be easily achieved with R through a loop function that first extracts the hour and then outputs the right classification.

```
> len <- dim(train)[1]
> Day <- c()
> for (i in 1:dim(train)[1]){
   H <- hour(train$date[i])
   ifelse(between(H, 8, 20), Day <- append(Day, TRUE), Day <- append(Day, FALSE))
 }
> train$Day <- Day
```

Another way we shall make use of the temporal variable in a smart way is by calculating the change in value for our features between one observation and the precedent one. It could be, for instance, that the temperature record alone is not significant for predicting the presence of people, but the temperature difference between one minute and the one before does. The coding for the extraction of the **CO2_change**, **temp_change**, **hum_change** and **light_change** variables is presented next. This time we have reported only the lines of code for the test set because we have to take care of one situation: since two temporally distinct datasets have been joined, the line in which there is a time jump should not report the difference between the values recorded at that time and the values recorded in the line above. Instead, we are inserting neutral values corresponding to zero, since we do not really know the actual change at that moment in time.

Let us now plot the behaviour of the new variables over time.

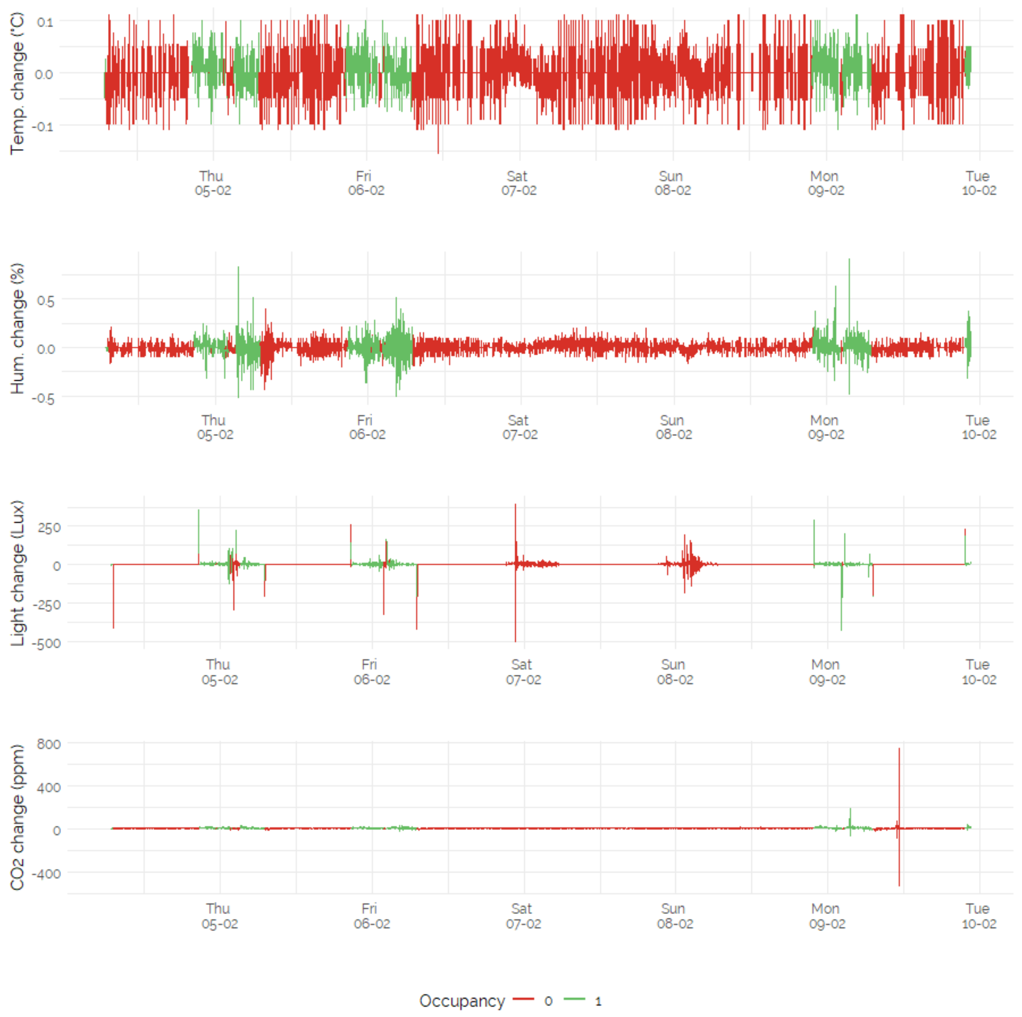

Figure 3: Variation over time in environmental conditions.

Although the third and the fourth plots suffer from the presence of odd observations that make the overall understanding rather hard, some interesting insights can be found. Those features that appear to be more dependent on the occupancy of the room are the second and the third, namely the change in humidity and the change in light. To visualize the interactions between the complete set of features, **Figure 4** shows correlations.

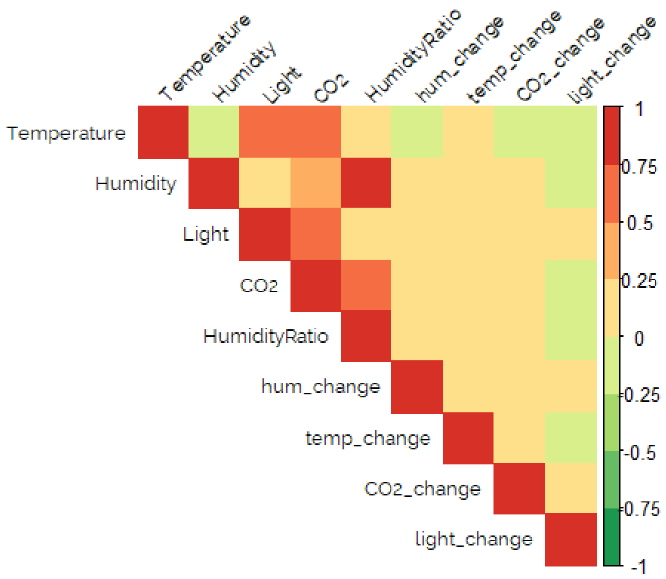

Figure 4: Correlations.

Except a high correlation between humidity and humidity ratio, that we were expecting, other variables that seem to go hand in hand are Light, CO2 and Temperature.

2 Models

Our aim is to predict whether there are people in the room: such a question can be shaped into a binary classification problem within the domain of supervised learning. Many different models and analyses are suitable for the case, such as Logistic Regression, Linear Prediction Analysis (LDA), Quadratic Discriminant Analysis (QDA), K-Nearest-Neighbors (KNN), Support Vector Classifier (SVC), Classification Trees and Random Forests. In the next pages, we will examine each of them, focusing on the accuracy in making predictions and on the type of errors committed. Before running the analyses, we consider an appropriate measure to select the features, according to formal analysis and on visual inspection. Temperature, Humidity, Light, CO2, as provided by the original dataset, will be considered, as well as humidity variation, light variation and Day, while it seems that the variations in CO2 and temperature are not significant for predicting occupancy, as suggested by **Figure 3**. Let us start our (eco-friendly) journey, then!

2.1 Logistic Regression

The logistic regression is a particular type of generalized linear model that describes the behaviour of a binary dependent variable by assuming a linear relationship with the parameters. We build a model with the variables selected as explained above.

```
Call:
glm(formula = Occupancy ~ Temperature + Humidity + Light + CO2 +
    hum_change + light_change + Day, family = "binomial", data = train)
```

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.5634  -0.0472  -0.0283  -0.0112   2.3608

Coefficients:
               Estimate Std. Error z value Pr(>|z|)
(Intercept)  24.4212530  3.5698974   6.841 7.87e-12 ***
Temperature  -1.6843887  0.1751793  -9.615  < 2e-16 ***
Humidity     -0.0503321  0.0249911  -2.014  0.04401 *
Light         0.0221908  0.0009251  23.988  < 2e-16 ***
CO2           0.0063415  0.0005784  10.964  < 2e-16 ***
hum_change    2.8974722  1.0676778   2.714  0.00665 **
light_change -0.0054737  0.0026218  -2.088  0.03682 *
DayTRUE       0.6657299  0.3681227   1.808  0.07054 .
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 8420.30  on 8142  degrees of freedom
Residual deviance:  825.92  on 8135  degrees of freedom
AIC: 841.92

Number of Fisher Scoring iterations: 9
```

The results show that all the variables have a significant effect on the occupancy. As we are dealing with a non-linear model, we cannot quantify the exact probability of getting 0 or 1, but the z-values suggest that *Light* has the greatest influence on the dependent variable. As for the direction of the relation, *Light*, *CO2* and *hum_change* have a positive relation with Occupancy, meaning that as they increase, so does the probability of people being in the room. Also, during the day, there is a higher chance of the room being occupied. On the other hand, although it may seem counterintuitive, the estimates of *Temperature* and *Humidity* are negative, meaning that when the temperature or the humidity levels get higher, it is less likely that there are people in the room. We believe that this result should be taken with a grain of salt, since it is evident from the Figure 1 and 2 that the mean of both of these variables, as of all the other conditions, is higher when the room is occupied. Although counterintuitive, this outcome should not come as a total surprise, since it may happen that the effect of a single predictor on a variable changes when more predictors are considered together. Also, we notice that during the weekend, while the light and the CO2 levels are firmly low, both humidity and temperature behave differently, which may play a role in the model.
Anyway, before testing, we want to make sure that the exclusion of the other features (CO2 variation and temperature variation) is reasonable, so we build a second model.

```
Call:
glm(formula = Occupancy ~ Temperature + Humidity + Light + CO2 +
    hum_change + light_change + Day + CO2_change + temp_change,
    family = "binomial", data = train)
```

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.4781  -0.0467  -0.0281  -0.0111   2.3775

Coefficients:
               Estimate Std. Error z value Pr(>|z|)
(Intercept)  24.5793393  3.5980650   6.831 8.42e-12 ***
Temperature  -1.6914873  0.1763612  -9.591  < 2e-16 ***
Humidity     -0.0504298  0.0251232  -2.007  0.04472 *
Light         0.0222955  0.0009393  23.736  < 2e-16 ***
CO2           0.0062306  0.0005841  10.668  < 2e-16 ***
hum_change    2.9667472  1.0799926   2.747  0.00601 **
light_change -0.0055737  0.0026200  -2.127  0.03339 *
DayTRUE       0.7219009  0.3695713   1.953  0.05078 .
CO2_change    0.0036947  0.0029048   1.272  0.20340
temp_change  -3.1974889  3.6046362  -0.887  0.37505
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 8420.30  on 8142  degrees of freedom
Residual deviance:  824.15  on 8133  degrees of freedom
AIC: 844.15

Number of Fisher Scoring iterations: 9
```

As expected, these two variables are not significant predictors, and their inclusion slightly increases the Akaike Information Criterion (AIC), which is an estimate of the relative information lost, hence the new model is not more informative.
It is now time to test the model. We start with a default classification threshold of 0.5.

```
          Reality
Prediction    0    1
         0 9291  156
         1  105 2865

Accuracy:  0.979
```

The accuracy is pretty high: almost 98% of the predictions are correct. But we are confident that it is possible to do better than this. With the aim of making a hypothetical automated energy saving System sustainable and efficient, we want to improve the prediction accuracy.
By looking at the confusion matrix, we notice that the logistic regression model, in this case, tends to commit more type II errors (i.e. occupied room being classified as empty). Therefore, to improve accuracy, we try to make it easier for the model to predict a positive class: after lowering the threshold to 0.4, we compare the outcome with the performance of a model with a higher threshold. To summarize how the accuracy varies accordingly, it is useful to draw the ROC curve for each model.

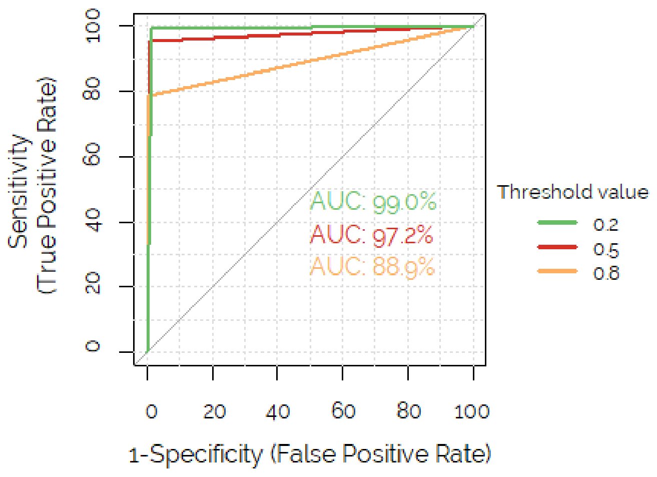

Figure 5: Logistic Regression Performance.

As **Figure 5** shows, the model reaches the best prediction accuracy with a lower threshold, that is when we reduce the number of type II errors (i.e. false negatives).

```
          Reality
Prediction    0    1
         0 9288   31
         1  108 2990

Accuracy:  0.9888
```

2.2 Linear Discriminant Analysis

Linear Discriminant Anaysis allows us to express the categorical Occupancy variable as a linear combination of the features, by turning to the Bayes' theorem. Again, we will consider the variables selected beforehand. And again, we want to improve the classification accuracy.

```
          Reality
Prediction    0    1
         0 9268   10
         1  128 3011

Accuracy:  0.9889
```

The confusion matrix reveals that the Linear Discriminant Analysis behaves differently from the Logistic Regression, in terms of most frequent errors. While the previous model tends to commit more false negatives (type II error), LDA has a higher rate of false positives (i.e., the room being categorized as occupied when it is actually empty). This time, the threshold shall be risen to improve the accuracy, and the increase must be significant to have an effect.

```
          Reality
Prediction    0    1
         0 9290   16
         1  106 3005
```

```
Accuracy:  0.9902
```

The results obtained with different thresholds are summarized by the ROC curves.

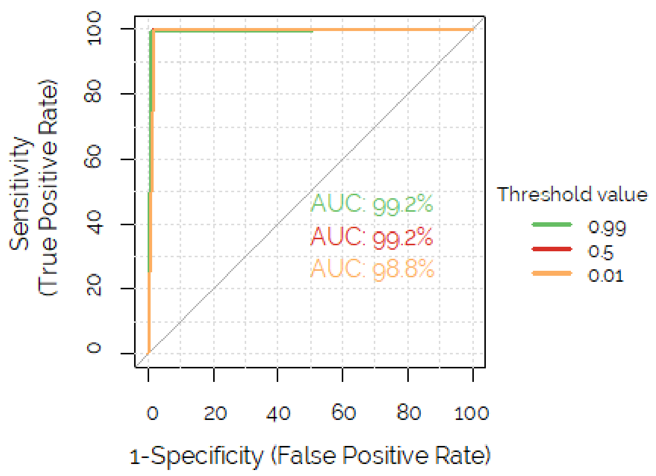

Figure 6: LDA Performance.

Overall, the LDA method performs very well on the validation set, reaching an accuracy of 99% with a threshold of 0.99. Interestingly, with respect to the Logistic Regression, we have to change the threshold by a large amount before appreciating an improvement in accuracy.

2.3 Quadratic Discriminant Analysis

An alternative to LDA is the Quadratic Discriminant Analysis (QDA), which, as the former, is based on Bayes' theorem to perform predictions. The main difference is that QDA does not assume that the two classes have the same covariance matrix, instead it considers that each one has its own. As a consequence, QDA has more parameters to estimate, hence it is more flexible than LDA.

```
       truth
predict    0    1
      0 9150   22
      1  246 2999

Accuracy:  0.9784
```

The output shows that the classification accuracy on the validation set has slightly decreased with respect to LDA. Such a result suggests that less flexible models may work better and prevent overfitting. In fact, it may be the case that models which are less flexible are more suitable to generalize our data.

2.4 Support Vector Classifier

We look for the hyperplane that best splits our data in two. The natural choice is the maximal margin hyperplane, which is the separating hyperplane that is the farthest from the training observations, if it exists. However, we allow for some observations to violate the hyperplane separation, to guarantee some generalization. The parameter *cost* of the function *svm* in R controls for the amount of flexibility of our model: we want a margin that is as accurate as possible on the training data. Hence, we fit our data hard indicating a high cost for each of the violations. The kernel we have chosen is *linear*. Although it is impossible to have a mental map of the distribution of multi-dimensional data, we imagine that points are approximately split on the occupancy dimension by a linear decision boundary.

```
Call:
svm(formula = Occupancy ~ Temperature + Humidity +
    Light + CO2 + Day + hum_change + light_change,
    data = train, kernel = "linear", cost = 100,
    scale = T)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  100
      gamma:  0.125

Number of Support Vectors:  287

 ( 144 143 )


Number of Classes:  2

Levels:
 0 1
```

The only observations that do affect the support vector classifier are those that lie directly on the margin or on the wrong side of it. In our first attempt we have 314 support vectors. Even though with such a high cost parameter the model is going to behave very well on the training data, this does not necessarily imply the best behavior on test data.

```
       truth
predict    0    1
      0 9288   18
      1  108 3003

Accuracy:  0.9899
```

As our previous models, performance is good. In particular, the Support Vector Classifier by default tends to minimize false negatives: the situation “occupancy predicted when the room is empty” will occur more often than “Emptiness predicted when room is occupied”.
Even though there are few prediction errors when we validate our model, it is quite likely that

we can further improve the performance by allowing for more violations. We make our model less sensible to the training data by enlarging the margin of the classifier. For choosing the best cost parameter the simplest way is to compare the performance on the validation set.

```
> SVCs <- data.frame(matrix(, nrow=, ncol=2))
> names(SVCs) <- c("cost", "Accuracy")
> counter = 1
> for (i in c(50, 10, 5, 1, 0.1, 0.01, 0.001)){
   svc.fun <- svm(Occupancy ~ Temperature + Humidity + Light + CO2 +
                  Day + hum_change + light_change, train, kernel='linear',
                  cost=i, scale=T)
   svc.pred <- predict(svc.fun, test)
   SVCs[counter,] <- c(i, mean(svc.pred == test$Occupancy))
   counter = counter + 1
 }
> SVCs
```

```
   cost  Accuracy
1 5e+01 0.9898526
2 1e+01 0.9898526
3 5e+00 0.9898526
4 1e+00 0.9898526
5 1e-01 0.9897721
6 1e-02 0.9893694
7 1e-03 0.9860675
```

The best performing cost parameter is 1: further decreasing cost makes our margin too wide and increases the number of resulting violations too much. In the bias-variance tradeoff we start with high variance but low bias when we have a high cost parameter and a consequent narrow margin. For this reason we start decreasing the cost value. However, after an equilibrium at around 1 the margin starts becoming too wide: the number of support vectors is becoming too large and we are going towards a situation of high bias. In fact, we are fitting our data less hard and at a certain point (for cost = 1),the amount of rigidity of the system is such that the performance starts decreasing.

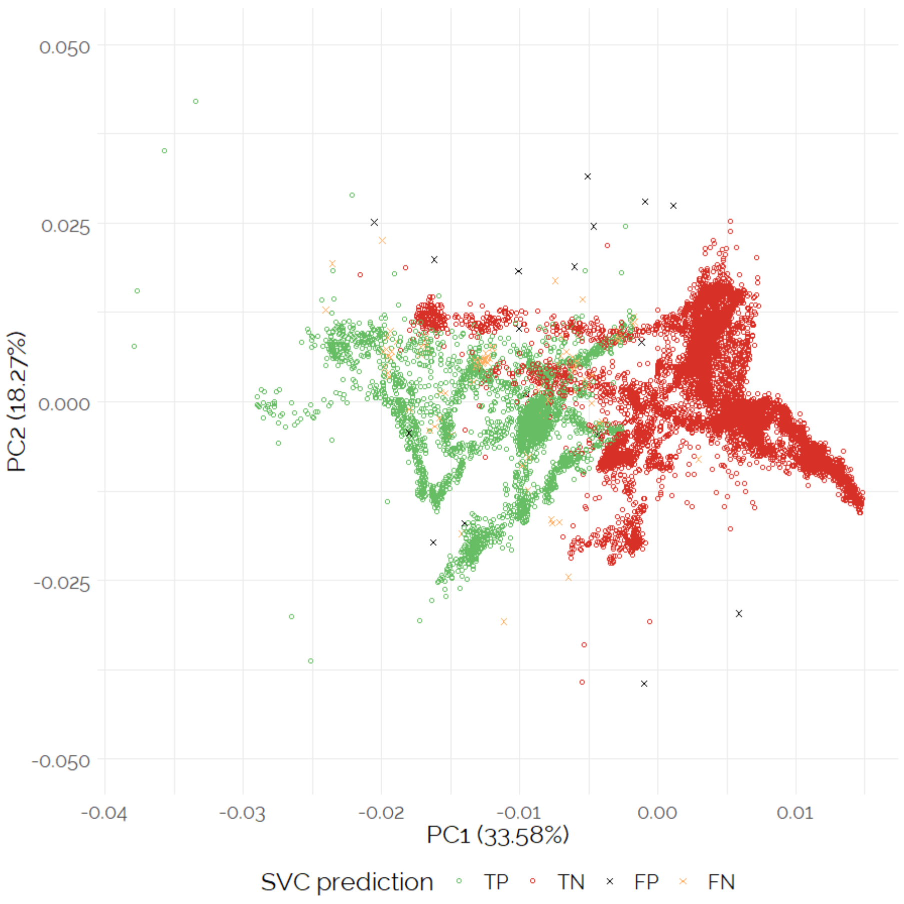

Figure 7: SVC Performance.

Ideally, if we were able to visualize data in a multi-dimensional way, we would have the chance of looking at a separating hyperplance with only 324 vectors (4% of the observations) lying on it. Since unfortunately this is not the case, we will settle for a bidimensional visualization thanks to a Principal Component Analysis algorithm. True positives are drawn with a green circle, true negatives with a red circle, false positives with a black cross and false negatives with a yellow cross. Of course with such a dimensionality reduction much of the accuracy is lost and the two classes are not as split up as they would be in a multi-dimensional representation.

2.5 K-Nearest Neighbours

The K-Nearest Neighbour is the simplest and most naive algorithm available: we simply conclude that the class of the observation we are testing is the one of the majority of the k-closest observations in the n-dimensional space (using the Euclidean distance as similarity metric). This means that the dimensions with higher variability are going to be most relevant ones when predicting the outcome. The variables we take in consideration when building the n-dimensional space are the numerical ones used for building our statistical models, hence we exclude the variable Day. We fix the parameter k at 3.

```
       truth
predict    0    1
```

```
       0 9072  191
       1  324 2830

Accuracy:  0.9585
```

The algorithm performs quite poorly with respect to the previous models. In general, we notice how KNN and the other flexible models as QDA tend to be less accurate than the most rigid choices, such as LDA. In the bias-variance trade-off, performance improves moving from situations of higher variance towards situations of higher bias. Reducing the number of dimensions by PCA, we shall obtain a simple plot of the misclassified observations.

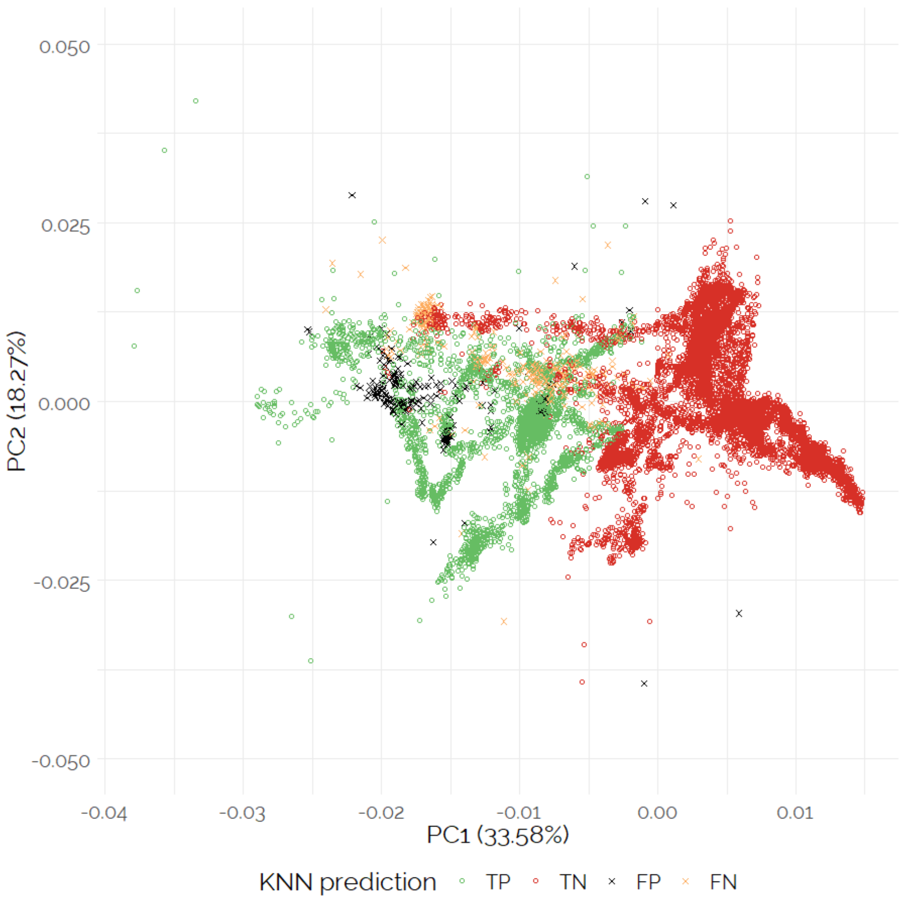

Figure 8: KNN Performance.

2.6 Decision Tree

Trees are trivial and visually pleasant ways of modelling classification problems. We simply segment the predictor space into a number of regions, where the dimensions on which the splits take place are the most significant ones. Decision trees tend to perform quite bad when taken singularly, being them very sensitive to small changes in the data, and hence quite non-robust. Therefore, in this phase, we do not expect to achieve a high level of accuracy, yet we are interested in investigating how the predictor space is going to be subset.

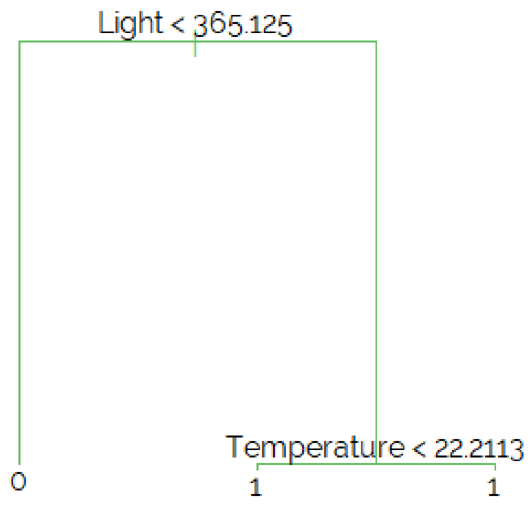

Figure 9: The decision tree.

What a tiny tree! After examining complex models with large variance, the simplicity of this model is surprising. In fact, we could expect to better account for the variability of the data by means of a not too complicated model, as the previous analyses have suggested, but still we did not imagine to end up with such a simple subdivision. Is this tree with only three leaves going to capture the variability of the test set as well?

```
          Reality
Prediction    0    1
         0 9287   15
         1  109 3006

Accuracy:  0.9900137
```

The tree predictions are actually very accurate, with only 1% of the observations being misclassified. In other words, to correctly guess whether there are people in the room, it is sufficient to know the level of luminous emittance: if it is lower than around 365 lux, the room is extremely likely to be empty; if it is higher, someone is inside. We may speculate that the office is equipped with an automated lightening control system based on motion sensor.
Furthermore, if we look at the right part of the tree, we can notice that an additional split is made based on the temperature, which further discriminates within class 1. To understand the meaning of it, we should keep in mind that a decision tree continues splitting until the leaf nodes are sufficiently pure. However, for our prediction purpose, we do not ask the model to perfectly fit the training data, as this would result in overfitting. For this reason, we expect that pruning the tree to make it even simpler shall not affect the prediction accuracy.

```
          Reality
Prediction    0    1
         0 9287   15
         1  109 3006

Accuracy:  0.9900137
```

2.7 Forests

Random forest is a procedure to reduce the variance of a single tree, by building a large number of classification trees on bootstrapped training samples. In this case, however, there is no need to reduce the variance, as we are keeping only one variable. As a consequence, we presume that the prediction accuracy of the random forest is going to decrease as the number of features included in the model rises. This behaviour can be easily inspected by adjusting the tuning parameter *mtry* in the *randomForest* function.

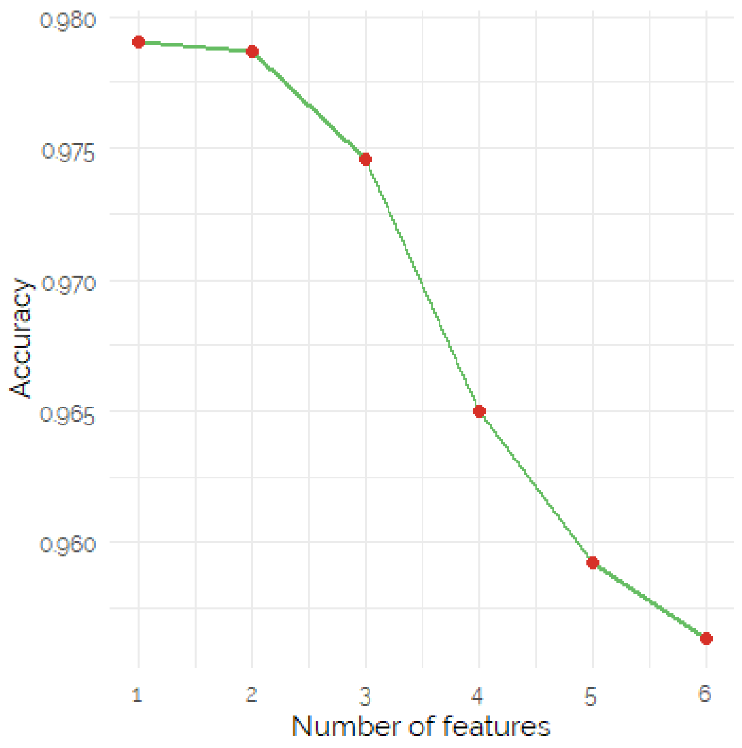

Figure 10: Performance of Random Forests.

3 Turning Lights off

Predicting occupancy with a sensor measuring the luminous flux per unit area is trivial, as demonstrated. However, let us think about a System built for turning lights off when the room is not occupied. Would it be convenient for the model to include light as a variable for predicting occupancy? Given that there is not an automated lightening system based on motion, this is questionable. The choice of whether turning the lights off or not would be simply based on whether the luminous emittance would exceed a threshold (of around 350, in the case of trees). Hypothetically, we shall imagine that once the light is turned on, the System would never switch it off simply because it records a high value of luminous emittance and hence predicts that the room is occupied. To avoid the problem, we shall simply decide to ground the occupancy-non occupancy choice not on all the variables but only on the ones that are not measured in the lux unit. In other words, we exclude variables *Light* and *light_change*.

Let us build a generalized linear model of family binomial to observe whether variables that were excluded from the previous analyses might end up being more significant now that the feature light is not considered.

```
Call:
glm(formula = Occupancy ~ Temperature + Humidity + CO2 + hum_change +
    CO2_change + temp_change + Day, family = "binomial", data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-8.4904  -0.2159  -0.1002  -0.0344   3.5172

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.8868929  1.4235175  -2.028   0.0426 *
Temperature -0.0408222  0.0650069  -0.628   0.5300
Humidity    -0.2359571  0.0145733 -16.191  < 2e-16 ***
CO2          0.0091317  0.0003148  29.012  < 2e-16 ***
hum_change   3.0468746  0.5915763   5.150 2.60e-07 ***
CO2_change   0.0742962  0.0077720   9.559  < 2e-16 ***
temp_change 11.8851964  1.5444946   7.695 1.41e-14 ***
DayTRUE      2.4048255  0.2063903  11.652  < 2e-16 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 8420.3  on 8142  degrees of freedom
Residual deviance: 3094.6  on 8135  degrees of freedom
AIC: 3110.6

Number of Fisher Scoring iterations: 8
```

Interestingly, now the variables *CO2_change* and *temp_change* have become consistently significant, so we are going to include them in our following analyses. **Figure 11** shows the behavior of the different algorithms when tested on the validation set. For predicting occupancy, we use a threshold of 0.5 when using Logistic Regression, LDA and QDA. For the Support Vector Classifier, the choice was on a cost parameter of 1, and for the KNN algorithm we have considered k=3.

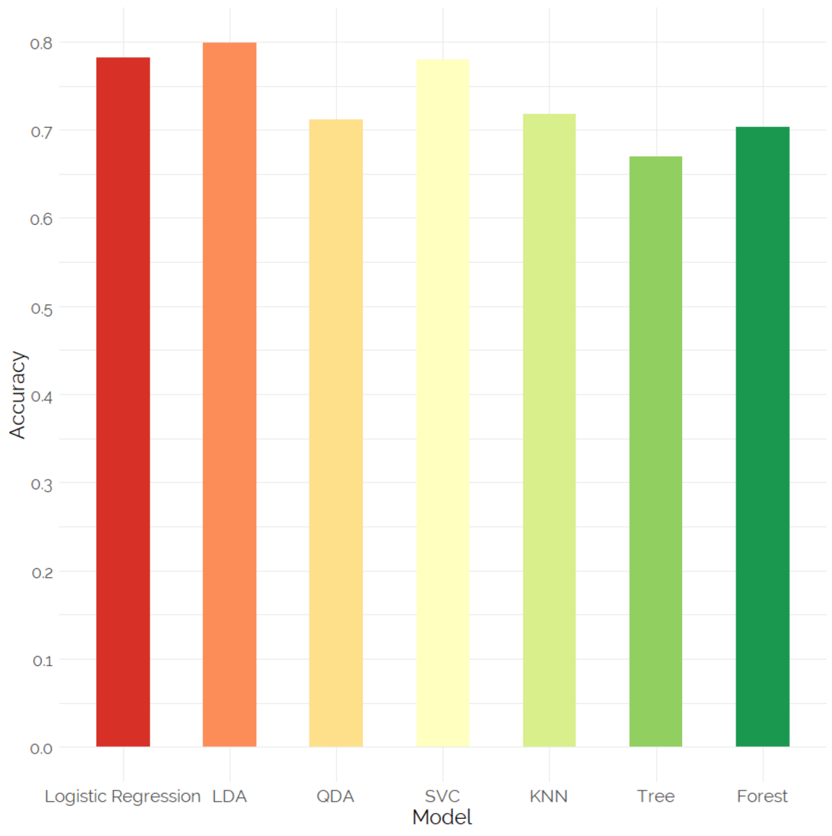

Figure 11: Accuracy of models when excluding light.

GLM, SVC and LDA perform comparably, with the latter reaching an accuracy around 80%. Performances of QDA and KNN algorithms are instead inferior, as it happened before excluding the light variable. The accuracy of classification trees and random forests (building each tree with only one variable) is even worse: respectively, only 67% and 70% of the predictions were right.

4 How many people?

Until now, our focus was on predicting the simple dependent variable "occupied/not occupied". More challenging and creative it is the idea of predicting how crowded the office is. With this goal in mind, we merge the train and validation datasets, but only select the observations where the room is occupied. This task being accomplished, we select the rows on which hierarchical clustering algorithms are to be applied. It is unlikely, for instance, that *light* and *light_change* have a real value when predicting the quantity of people. On the contrary, we are inclined to think that humidity, temperature and CO2 will increase when more workers are present in the office, so we are going to use this information when doing visualizations.

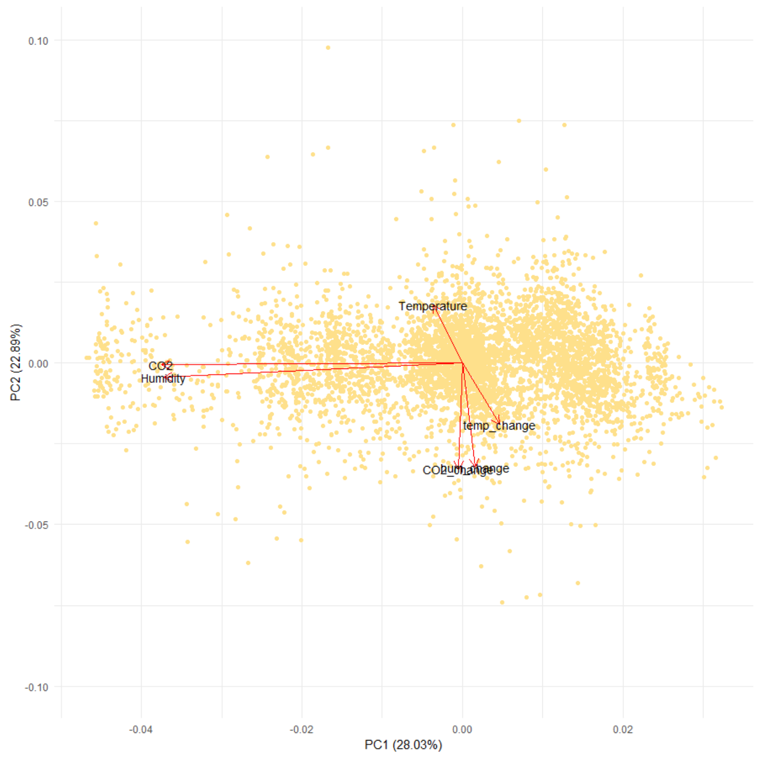

Figure 12: PCA with no light.

The two principal components explain around 50% of the variation. CO2 and humidity have the highest loadings on the first principal component, while the environmental changes over time have the highest loadings on the second principal component. In particular, CO2 and humidity shall be considered as good predictors for the level and the duration of the occupancy: points located on the left side of the plot likely represent observations taken at the end of the day and/or when the room has been occupied by workers for some time.

5 Conclusions

Let us try to draw a conclusion from all these results. We have found that the presence of people in the office can be inferred simply from the light level, as the classification tree and the random forest clearly reveal. This explains why more rigid models perform slightly better compared to more flexible ones, although the prediction accuracy of each of them is particularly high in any case. The variables providing information about other environmental conditions and their variations over time are still of interest. Since our aim was to optimize the System controlling for lights and temperature in an efficient and sustainable manner, we have imagined a scenario in which the illuminance level is unknown. In this case, the Logistic Regression is still able to reach a good accuracy. We have also explored all the observations at our disposal, from both the training and the test set, and taken an unsupervised approach by means of Principal Component Analysis. In particular, the changes in Humidity and Temperature considered together may be a proxy of something else going on in the room. And this, in turn, may be investigated in the path towards sustainability.