Predictions of Energy Efficiency of buildings using machine learning

Mami Daba Fam

2022-11-22

My second capstone project in Data Science Professional Certificate

1. Introduction

This report is the part 2 of my data science capstone project. The purpose is to do a project that will apply machine learning techniques that have been studied during the courses. I have to made the choice of the subject and the dataset to be used for this project.

I am very interested in the energy efficiency sector. Then, my project will use a dataset downloaded on the UCI Machine Learning Repository. I will study the estimation of energy efficiency of buildings by using models such as linear regression and RandomForest.

It is an enthusiastic project on establishing the effect of eight input variables on two output variables, namely heating load (HL) and cooling load (CL) of buildings. The predictors or features are identified in the dataset as X1, X2,. . . ., X8 and the two outcomes as Y1 and Y2.

The main question is how well can we predict the heating load (Y1) and the cooling load (Y2) based on the following parameters of buildings X1 (Relative compactness), X2 (Surface Area), X3 (Wall Area), X4(Roof Area), X5(Height), X6(Orientation), X7(Glazing area), X8(Glazing variations)?

Which predictors are most important variables on predicting heating load and cooling load?

To answer this, my report will be structured as following :

- Explore the dataset as it is already in a tidy format
- Make data visualization for better explanation of information given by our data and Better analysis of effects
- Present my modeling approach mainly focus on linear regression and random forest. The two models are established to perform well in the literature.
- Define and choose an evaluation method for models.
- Present results and models performances.
- Conclusion and recommendations.

2. Data exploration and visualization

The information given by Angeliki Xifara who created the dataset in UCI machine learning repository is below :

We perform energy analysis using 12 different building shapes simulated in Ecotect. The buildings differ with respect to the glazing area, the glazing area distribution, and the orientation, among other parameters. We simulate various settings as functions of the afore-mentioned characteristics to obtain 768 building shapes. The dataset comprises 768 samples and 8 features, aiming to predict two real valued responses. It can also be used as a multi-class classification problem if the response is rounded to the nearest integer.

All the buildings have the same volume, but different surface areas and dimensions.

2.1 Download and split data into training and test set

We will first download our data and divide it into a training set and a test set.

```
## Create a training set and a 20% testing set of
## our energy efficiency data set

# the public dataset loaded in UCI Machine
# learning is in xlsx format.

# Let's first load the required library

if (!require(tidyverse)) install.packages("tidyverse",
    repos = "http://cran.us.r-project.org")
if (!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
if (!require(randomForest)) install.packages("randomForest",
    repos = "http://cran.us.r-project.org")
if (!require(GGally)) install.packages("GGally", repos = "http://cran.us.r-project.org")
if (!require(Metrics)) install.packages("Metrics",
    repos = "http://cran.us.r-project.org")
if (!require(readxl)) install.packages("readxl", repos = "http://cran.us.r-project.org")

library(tidyverse)
library(caret)
library(randomForest)
library(GGally)
library(Metrics)
library(readxl)

## url of our energy efficiency public dataset is
## https://archive.ics.uci.edu/ml/machine-learning-databases/00242/ENB2012_data.xlsx

# Download our public dataset on energy
# parameters of buildings

tmp_energy <- tempfile()
download.file("https://archive.ics.uci.edu/ml/machine-learning-databases/00242/ENB2012_data.xlsx",
    tmp_energy, quiet = TRUE, mode = "wb", method = "auto",
    cacheOK = TRUE, options(timeout = max(1000, getOption("timeout"))))

energy_efficiency <- read_excel(tmp_energy)

# energy_test will be used as a validation set.
# For testing our models, we will only train on
# energy_training dataset

set.seed(1)
test_index <- createDataPartition(y = energy_efficiency$Y1,
    times = 1, p = 0.2, list = FALSE)
energy_training <- energy_efficiency[-test_index, ]
energy_test <- energy_efficiency[test_index, ]
```

2.2 Exploration of our datasets

This is an important part that let me better understand characteristics of energy efficiency of buildings used in this dataset and the relation between variables.

The entire energy_efficiency that is downloaded, has the following structure and dimension.

```
str(energy_efficiency)
```

```
## tibble [768 x 10] (S3: tbl_df/tbl/data.frame)
##  $ X1: num [1:768] 0.98 0.98 0.98 0.98 0.9 0.9 0.9 0.9 0.86 0.86 ...
##  $ X2: num [1:768] 514 514 514 514 564 ...
##  $ X3: num [1:768] 294 294 294 294 318 ...
##  $ X4: num [1:768] 110 110 110 110 122 ...
##  $ X5: num [1:768] 7 7 7 7 7 7 7 7 7 7 ...
##  $ X6: num [1:768] 2 3 4 5 2 3 4 5 2 3 ...
##  $ X7: num [1:768] 0 0 0 0 0 0 0 0 0 0 ...
##  $ X8: num [1:768] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Y1: num [1:768] 15.6 15.6 15.6 15.6 20.8 ...
##  $ Y2: num [1:768] 21.3 21.3 21.3 21.3 28.3 ...
```

```
dim(energy_efficiency)
```

```
## [1] 768  10
```

There are 768 observations in the public energy dataset and 10 variables. After splitting, our training dataset have 612 observations and 10 variables.

In the introduction, we have quickly explain what the predictors X and outcomes Y are related. The table below gives more details for this variables.

Data	Description (unit)	Number of possible value
X1	Relative Compactness - No units	12
X2	Surface Area - m^2	12
X3	Wall Area - m^2	7
X4	Roof Area - m^2	4
X5	Height - m	2
X6	Orientation - 2:North, 3:East, 4:South, 5:West - No units	4
X7	Glazing Area - 0%, 10%, 25%, 40% (of floor area) - No units	4
X8	Glazing Variations - 1:Uniform, 2:North, 3:East, 4:South, 5:West	6
Y1	Heating Load - kWh/m^2	586
Y2	Cooling Load - kWh/m^2	636

For next step in data exploration, I will use the training set to show existing correlation between predictors and outcomes and the behavior of our variables.

2.2.1 Summarizing the statistics of our variables

```
summary(energy_training)
```

```
##        X1              X2              X3              X4
##  Min.   :0.6200   Min.   :514.5   Min.   :245.0   Min.   :110.2
##  1st Qu.:0.6900   1st Qu.:612.5   1st Qu.:294.0   1st Qu.:122.5
##  Median :0.7600   Median :661.5   Median :318.5   Median :147.0
##  Mean   :0.7659   Mean   :670.2   Mean   :318.1   Mean   :176.0
```

```
##  3rd Qu.:0.8200   3rd Qu.:735.0   3rd Qu.:343.0   3rd Qu.:220.5
##  Max.   :0.9800   Max.   :808.5   Max.   :416.5   Max.   :220.5
##        X5              X6              X7              X8
##  Min.   :3.500   Min.   :2.000   Min.   :0.0000   Min.   :0.00
##  1st Qu.:3.500   1st Qu.:3.000   1st Qu.:0.1000   1st Qu.:2.00
##  Median :7.000   Median :3.000   Median :0.2500   Median :3.00
##  Mean   :5.267   Mean   :3.497   Mean   :0.2339   Mean   :2.82
##  3rd Qu.:7.000   3rd Qu.:4.000   3rd Qu.:0.4000   3rd Qu.:4.00
##  Max.   :7.000   Max.   :5.000   Max.   :0.4000   Max.   :5.00
##        Y1              Y2
##  Min.   : 6.01   Min.   :10.90
##  1st Qu.:12.99   1st Qu.:15.64
##  Median :18.95   Median :22.51
##  Mean   :22.33   Mean   :24.59
##  3rd Qu.:31.67   3rd Qu.:33.08
##  Max.   :43.10   Max.   :48.03
```

2.2.2 Correlation between variables

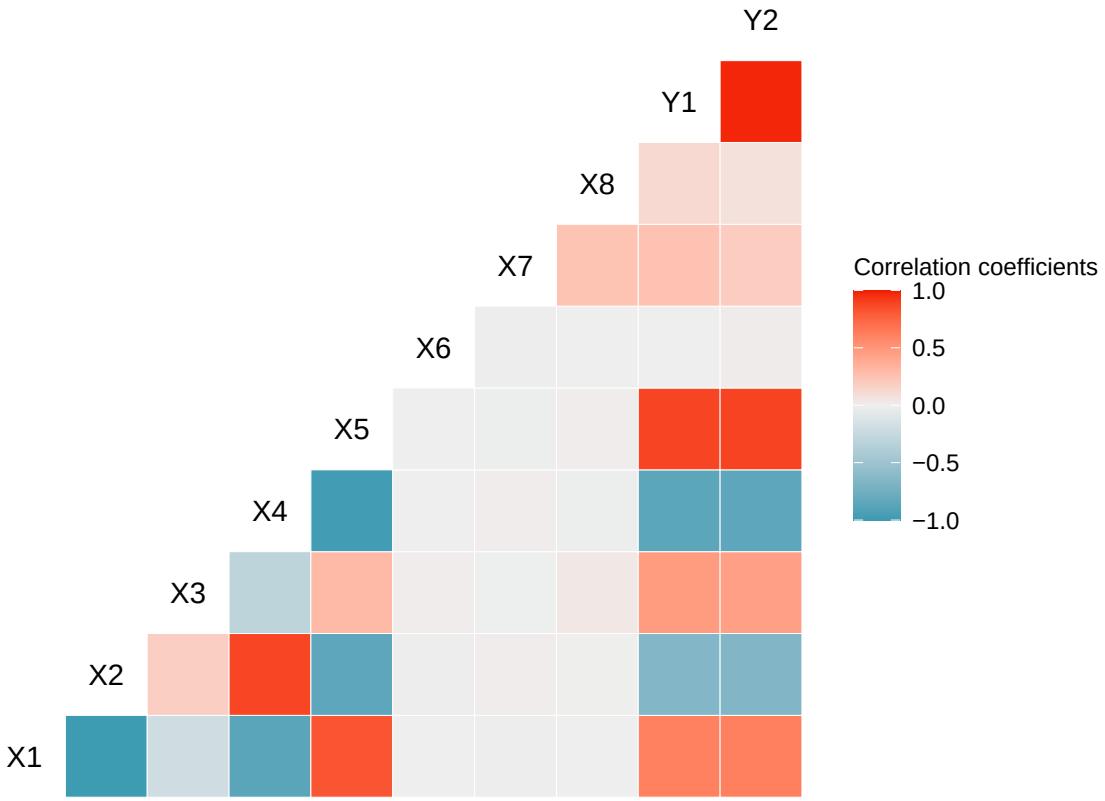
```r
## Correlation between variables
cor_matrix <- round(cor(energy_training), 2)


## As, it is a symmetrical matrix we can only
## show the half
as.dist(cor_matrix)
```

```
##       X1    X2    X3    X4    X5    X6    X7    X8    Y1
## X2 -0.99
## X3 -0.20  0.19
## X4 -0.87  0.88 -0.30
## X5  0.82 -0.85  0.30 -0.97
## X6  0.00  0.00  0.01  0.00  0.00
## X7 -0.01  0.01  0.00  0.01 -0.02  0.01
## X8  0.00  0.00  0.04 -0.02  0.01  0.00  0.24
## Y1  0.61 -0.65  0.47 -0.86  0.89  0.00  0.26  0.12
## Y2  0.62 -0.66  0.45 -0.86  0.89  0.02  0.19  0.08  0.98
```

```r
# Install Gcally library for a visual
# representation of our variables correlation

library(GGally)
ggcorr(energy_training, name = "Correlation coefficients")
```



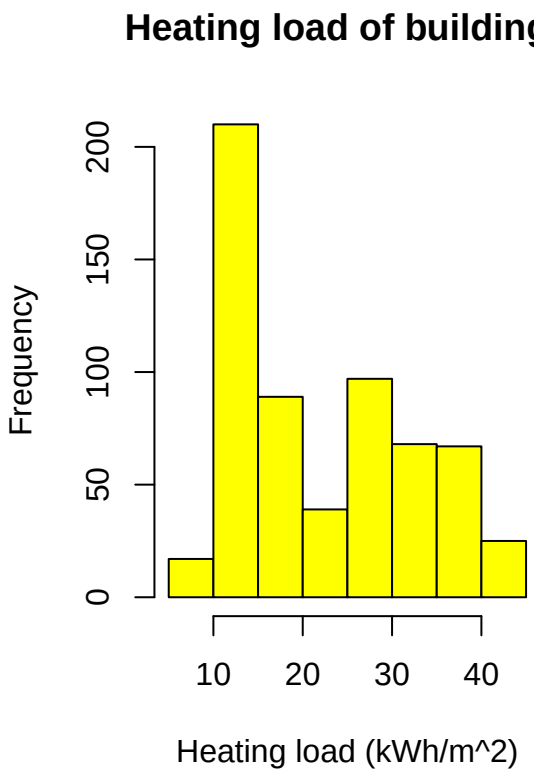
The correlation matrix show us that some variables are more correlated then others :

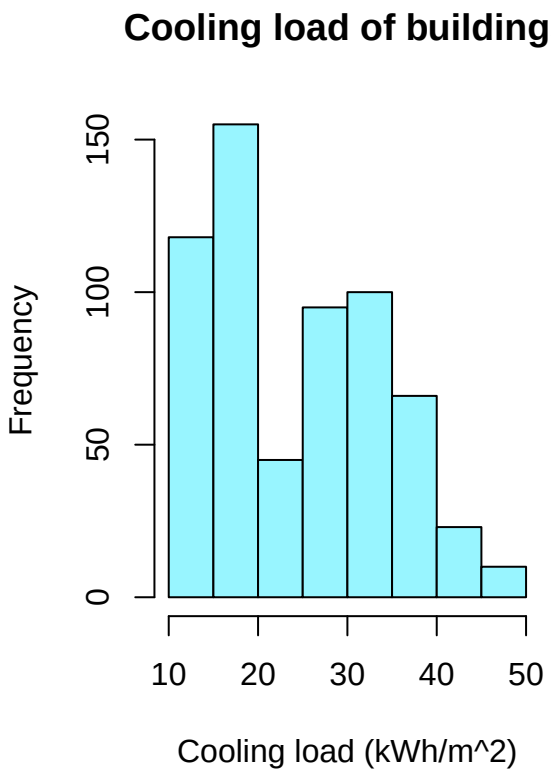
- The two outcomes Y1(Heating load) and Y2 (Cooling load) are highly correlated with a coefficient of 0.98. It is logical. If a building has a high energy efficiency, their heating and cooling load are both low.
- The height of the building (X5) is also highly correlated to the two outcomes around 0.9.
- all the predictors have similar coefficient of correlation for the two outcomes. It is also conform regarding the high correlation between the outcomes.
- Some predictors like X6 (Orientation of the building) has zero correlation with all others variables.
- Some predictors are much more correlated to the outcomes then others. We know that features have more predictive power when they are correlated to outcome and thus provide a better estimate of our outcomes.
- X7 : glazing area has a very low correlation with the outcomes (Y1) heating load and (Y2) cooling load.

2.2.3 Behavior of variables by histogram visualization

The two outcomes have similar behavior. Many buildings have low heating/ cooling load around 10 and 20 kWh/m^2. The statistics information corroborated this as the mean of heating load is $22,33 kWh/m^2$ and mean of cooling load is $24,59 kWh/m^2$.

```
par(mfrow = c(1, 2))
hist(energy_training$Y1, col = "yellow", xlab = "Heating load (kWh/m^2)",
    main = "Heating load of buildings")
hist(energy_training$Y2, col = "cadetblue1", xlab = "Cooling load (kWh/m^2)",
    main = "Cooling load of buildings")
```





However, some buildings have worst energy efficiency with high cooling and heating load.

```
# Buildings with heating load > 40 kWh/m^2

energy_training %>%
    filter(Y1 >= 40) %>%
    summarise(n())
```

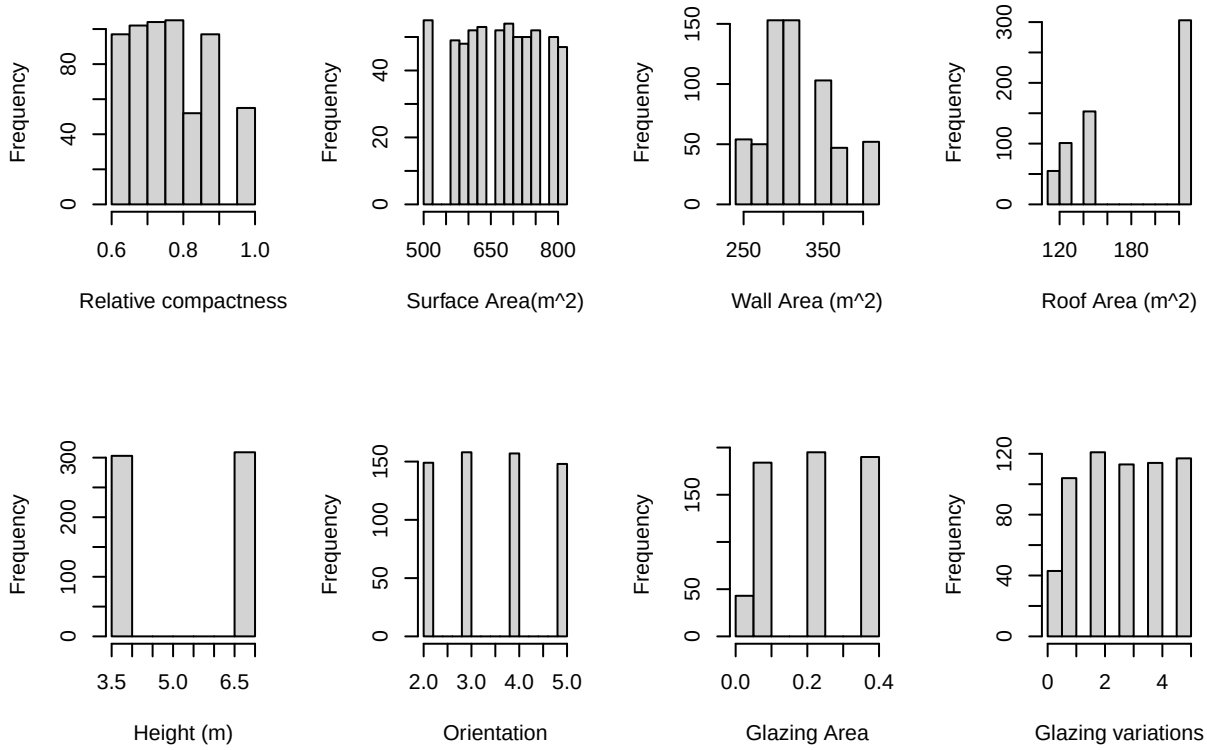
```
## # A tibble: 1 x 1
##   `n()`
##   <int>
## 1    26
```

```
# Buildings with cooling load > 40 kWh/m^2

energy_training %>%
    filter(Y2 >= 40) %>%
    summarise(n())
```

```
## # A tibble: 1 x 1
##   `n()`
##   <int>
## 1    33
```

What about our predictors behavior??



- Variables have a huge range of unit scales; very different to each others. I will apply standardization and normalization to compare the performances of my two models.

- Building height is divided in two main category: buildings height of 3,5m and 7m. It is a huge difference in the volume to heat or cool as the figure above has already demonstrate that this feature is highly correlated to outcomes.

```
# Buildings height categories
energy_training %>%
    group_by(X5) %>%
    summarise(n())
```

```
## # A tibble: 2 x 2
##      X5 `n()`
##   <dbl> <int>
## 1   3.5   303
## 2   7     309
```

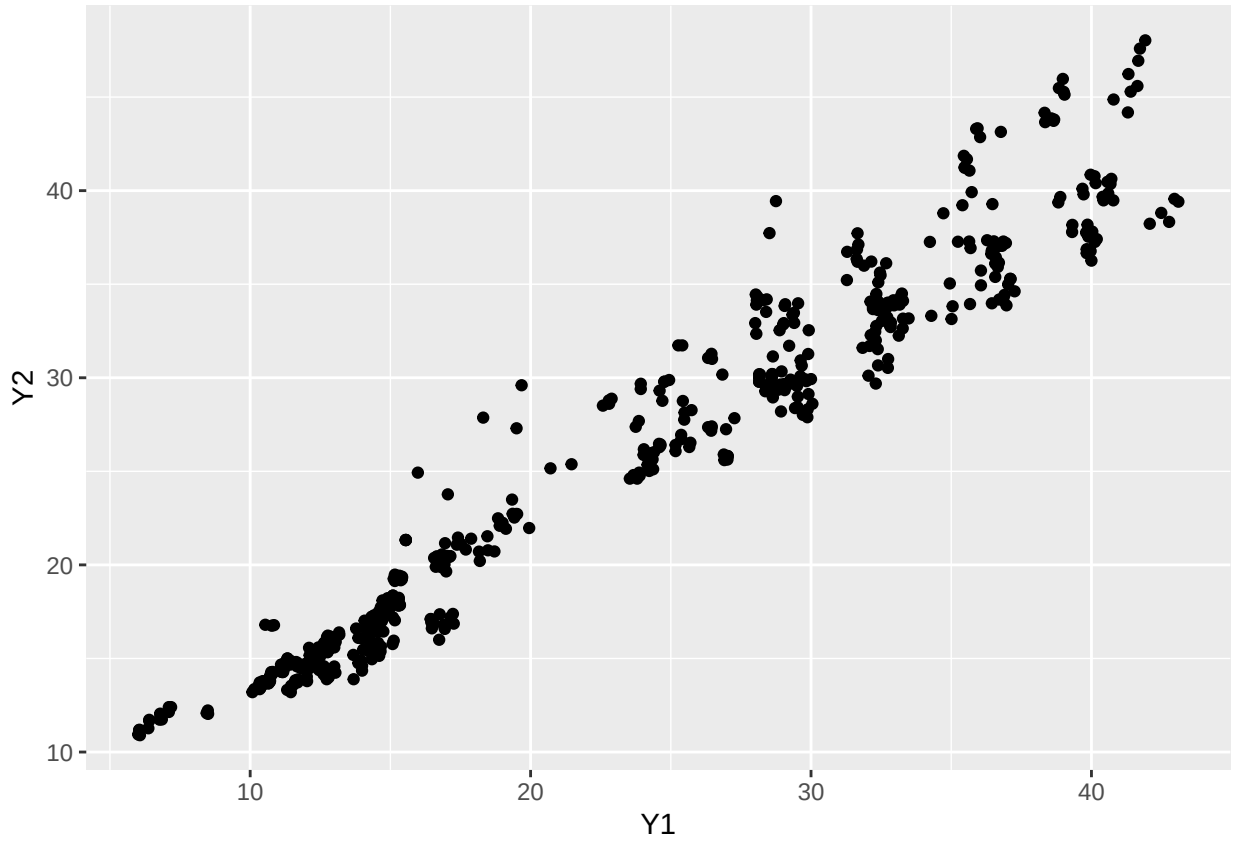
- Roof area is highly correlated to outcomes Y1, Y2 and height. Buildings have strong differences in roof area from one to double in only four categories. Roof Area of the buildings is between $110,25m^2$ and $220,5m^2$.

```
# Buildings Roof Area
energy_training %>%
    group_by(X4) %>%
    summarise(n())
```

```
## # A tibble: 4 x 2
##      X4 `n()`
```

```
##   <dbl> <int>
## 1  110.    55
## 2  122.   101
## 3  147    153
## 4  220.   303
```

```
# the two outcomes Y1 and Y2 are linearly
# correlated
energy_training %>%
    ggplot(aes(Y1, Y2)) + geom_point()
```



3. Models and evaluation methods

3.1 Model 1- linear regression

We will use linear regression for predicting Heating load Y1 and cooling load Y2. Y1 and Y2 are continuous. Our model is a multivariate linear regression. We have to predict two outcomes as a linear function of 8 predictors. I will train a unique model and a separate model for the two outcomes with the train() function include in the caret package.

$$Y_1 = \beta_{0,1} + \beta_{1,1}X_1 + ...\beta_{8,1}X_8$$
$$Y_2 = \beta_{0,2} + \beta_{1,2}X_1 + ...\beta_{8,2}X_8$$

3.2 Model 2 - random forest

A random forest is a supervised machine learning algorithm that is constructed from decision tree algorithms.It is used to solve regression problems. A random forest algorithm consists of many decision trees. The 'forest' generated by the random forest algorithm is trained through bagging or bootstrap aggregating. Bagging is an ensemble meta-algorithm that improves the accuracy of machine learning algorithms.

Random forest is a popular algorithm as it is simple to train and can perform very well by building multiple decisions trees and merging their prediction to get more accurate.

I will train random forest for each outcome. The aim of this model is to really improve my results regarding the first model of linear regression.

The train() function in caret package will also be use to fit my models.

3.3 Evaluation Methods

I will made a calculation of evaluation metrics usually adapted to linear regression : loss functions. I have choose MSE (Mean Squared Error) and RMSE (Root Mean Squared Error). RMSE is interesting as it is in the same unit of the outcomes. Calculations are made to define the loss between the predictor and actual outcome.

In this project, $\hat{Y_1}$ the predicted heating load and Y_1 the observed heating loading will be compared to determine the MSE and RMSE. The same error calculations will be made with the cooling load outcomes for each model.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(\hat{Y_i} - Y_i)^2$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{Y_i} - Y_i)^2}$$

Calculations of MSE and RMSE will be made directly with the mse() and rmse()functions available on metrics package.

This evaluation metrics will allow me to compare the performance of models. The best model is which minimizes MSE and RMSE.

4. Results and models performances

4.1. linear regression models with original data

Linear model for predicting the value of heating load Y1

```
# Fit the linear model of Y1 with all predictors

# Original data

original_model_lm_1 <- train(Y1 ~ X1 + X2 + X3 + X4 +
    X5 + X6 + X7 + X8, data = energy_training, method = "lm")

summary(original_model_lm_1)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.8290 -1.3473 -0.0259  1.4163  7.7523
##
## Coefficients: (1 not defined because of singularities)
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  80.536068  21.980732   3.664  0.00027 ***
## X1          -63.287451  11.846165  -5.342 1.30e-07 ***
## X2           -0.083318   0.019745  -4.220 2.82e-05 ***
## X3            0.057842   0.007639   7.572 1.38e-13 ***
## X4                  NA         NA      NA       NA
## X5            4.279611   0.388969  11.002  < 2e-16 ***
## X6           -0.035780   0.109988  -0.325  0.74506
## X7           19.716198   0.937074  21.040  < 2e-16 ***
## X8            0.241064   0.080458   2.996  0.00285 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.006 on 604 degrees of freedom
## Multiple R-squared:  0.9123, Adjusted R-squared:  0.9113
## F-statistic: 897.7 on 7 and 604 DF,  p-value: < 2.2e-16
```

```
varImp(original_model_lm_1)
```

```
## lm variable importance
##
##    Overall
## X7  100.00
## X5   51.54
## X3   34.98
## X1   24.22
## X2   18.80
## X8   12.89
## X6    0.00
```

```
# Predicting the trained model on test data
original_lm_Y1_hat <- original_model_lm_1 %>%
    predict(energy_test)

# Calculate RMSE

rmse(original_lm_Y1_hat, energy_test$Y1)
```

```
## [1] 2.647052
```

```
mse(original_lm_Y1_hat, energy_test$Y1)
```

```
## [1] 7.006883
```

We can observe a "NA" in the predictor X4- Roof Area that is explained by his highly correlation with X2 surface Area. I will train again the linear model without this predictors X4.

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.8290 -1.3473 -0.0259  1.4163  7.7523
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  80.536068  21.980732   3.664  0.00027 ***
## X1          -63.287451  11.846165  -5.342 1.30e-07 ***
## X2           -0.083318   0.019745  -4.220 2.82e-05 ***
## X3            0.057842   0.007639   7.572 1.38e-13 ***
## X5            4.279611   0.388969  11.002  < 2e-16 ***
## X6           -0.035780   0.109988  -0.325  0.74506
## X7           19.716198   0.937074  21.040  < 2e-16 ***
## X8            0.241064   0.080458   2.996  0.00285 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.006 on 604 degrees of freedom
## Multiple R-squared:  0.9123, Adjusted R-squared:  0.9113
## F-statistic: 897.7 on 7 and 604 DF,  p-value: < 2.2e-16
```

```
## lm variable importance
##
##    Overall
## X7  100.00
## X5   51.54
## X3   34.98
## X1   24.22
## X2   18.80
## X8   12.89
## X6    0.00
```

The error of the model does not change by dropping down X4.

Let's put in place a table which summarizes- our different results obtain through models training.

- **RESULTS**

```
## # A tibble: 1 x 3
##   method                                     RMSE   MSE
##   <chr>                                     <dbl> <dbl>
## 1 Model 1- LINEAR MODEL OF Y1 Original data  2.65  7.01
```

Linear model for predicting the value of cooling load Y2

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.5694 -1.5598 -0.1764  1.3724 11.2821
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  96.139657  23.881298   4.026 6.40e-05 ***
## X1          -70.717695  12.870444  -5.495 5.78e-08 ***
```

```
## X2           -0.086813   0.021452  -4.047 5.87e-05 ***
## X3            0.044563   0.008299   5.369 1.13e-07 ***
## X5            4.308799   0.422601  10.196  < 2e-16 ***
## X6            0.099099   0.119498   0.829    0.407
## X7           14.680888   1.018099  14.420  < 2e-16 ***
## X8            0.051385   0.087414   0.588    0.557
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.266 on 604 degrees of freedom
## Multiple R-squared:  0.8829, Adjusted R-squared:  0.8815
## F-statistic: 650.5 on 7 and 604 DF,  p-value: < 2.2e-16
```

```
## lm variable importance
##
##    Overall
## X7 100.000
## X5  69.462
## X1  35.474
## X3  34.569
## X2  25.007
## X6   1.746
## X8   0.000
```

- **RESULTS**

```
## # A tibble: 2 x 3
##   method                                      RMSE   MSE
##   <chr>                                      <dbl> <dbl>
## 1 Model 1- LINEAR MODEL OF Y1 Original data  2.65  7.01
## 2 Model 1- LINEAR MODEL OF Y2 original data  2.95  8.69
```

4.2. Random Forest models with original data

Fit RandomForest model for heating load Y1

```
# Original data Random Forest

original_model_rf_1 <- train(Y1 ~ X1 + X2 + X3 + X4 +
    X5 + X6 + X7 + X8, data = energy_training, method = "rf")

original_model_rf_1
```

```
## Random Forest
##
## 612 samples
##   8 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 612, 612, 612, 612, 612, 612, ...
## Resampling results across tuning parameters:
##
##   mtry  RMSE       Rsquared   MAE
##   2     1.1908835  0.9864905  0.9321857
##   5     0.5894285  0.9965572  0.3968864
```

```
##   8     0.6193517  0.9961822  0.4042285
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was mtry = 5.
```

```
# Variables importance
varImp(original_model_rf_1)
```

```
## rf variable importance
##
##    Overall
## X1 100.000
## X5  69.869
## X4  34.375
## X2  26.086
## X7  18.312
## X3   9.907
## X8   6.872
## X6   0.000
```

```
# Predicting the trained model on test data
original_rf_Y1_hat <- original_model_rf_1 %>%
    predict(energy_test)

# Calculate RMSE and MSE
RMSE_Y1_rf <- rmse(original_rf_Y1_hat, energy_test$Y1)
MSE_Y1_rf <- mse(original_rf_Y1_hat, energy_test$Y1)
```

- **RESULTS**

```
## # A tibble: 3 x 3
##   method                                    RMSE   MSE
##   <chr>                                    <dbl> <dbl>
## 1 Model 1- LINEAR MODEL OF Y1 Original data 2.65  7.01
## 2 Model 1- LINEAR MODEL OF Y2 original data 2.95  8.69
## 3 Model 2- RANDOMFOREST OF Y1 original data 0.416 0.173
```

Fit RandomForest model for cooling load Y2

```
# Original data Random Forest

original_model_rf_2 <- train(Y2 ~ X1 + X2 + X3 + X4 +
    X5 + X6 + X7 + X8, data = energy_training, method = "rf")

original_model_rf_2
```

```
## Random Forest
##
## 612 samples
##   8 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 612, 612, 612, 612, 612, 612, ...
## Resampling results across tuning parameters:
##
##   mtry  RMSE      Rsquared   MAE
```

```
##   2     1.905976  0.9607318  1.376012
##   5     1.855706  0.9624956  1.121938
##   8     1.909934  0.9602588  1.136327
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was mtry = 5.
```

```
# Variables importance
varImp(original_model_rf_2)
```

```
## rf variable importance
##
##    Overall
## X2 100.000
## X4  56.099
## X1  30.177
## X5  13.119
## X3  10.589
## X7   8.996
## X8   2.053
## X6   0.000
```

```
# Predicting the trained model on test data
original_rf_Y2_hat <- original_model_rf_2 %>%
    predict(energy_test)

# Calculate RMSE and MSE
RMSE_Y2_rf <- rmse(original_rf_Y2_hat, energy_test$Y2)
MSE_Y2_rf <- mse(original_rf_Y2_hat, energy_test$Y2)
```

- **RESULTS**

```
## # A tibble: 4 x 3
##   method                                      RMSE   MSE
##   <chr>                                      <dbl> <dbl>
## 1 Model 1- LINEAR MODEL OF Y1 Original data 2.65  7.01
## 2 Model 1- LINEAR MODEL OF Y2 original data 2.95  8.69
## 3 Model 2- RANDOMFOREST OF Y1 original data 0.416 0.173
## 4 Model 2- RANDOMFOREST OF Y2 original data 1.80  3.23
```

4.3 Models performances analysis on original data.

I have obtain interesting results in this first step of modeling. RandomForest is more accurate as I have supposed. It allow a lowest RMSE of 0.4161931.

Coefficients and intercept are given the summary of our linear model. In this first model, Glazing area (X7) is the most important variable for predicting heating load(Y1) and cooling load(Y2).

Orientation of buildings (X6) is the worst importance variables.It was also a variable without any correlation with any variables.

RMSE for predicting cooling load Y2 is always high on randomForest Models.

I will perform a standardization of variables to estimate how well this step could improve the model. Data Standardization is a preprocessing step that allow to put in a common format all the variables. In our dataset for example, we have height of building(X5) of 3.5m and 7m. Surface, wall and floor area are in a range of $600m^2$

4.4 linear regression models with standardized data

```
# Standardization of the data
energy_efficiency_S <- as.data.frame(scale(energy_efficiency,
    center = TRUE, scale = TRUE))

# Split standardize data into train and test set
set.seed(1)
test_index_S <- createDataPartition(y = energy_efficiency_S$Y1,
    times = 1, p = 0.2, list = FALSE)
energy_training_S <- energy_efficiency_S[-test_index,
    ]
energy_test_S <- energy_efficiency_S[test_index, ]

# Standardized linear model for Y1 without X4
standard_model_lm_1 <- train(Y1 ~ X1 + X2 + X3 + X5 +
    X6 + X7 + X8, data = energy_training_S, method = "lm")

summary(standard_model_lm_1)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.97411 -0.13353 -0.00257  0.14037  0.76830
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.003415   0.012046  -0.284  0.77687
## X1          -0.663454   0.124186  -5.342 1.30e-07 ***
## X2          -0.727356   0.172371  -4.220 2.82e-05 ***
## X3           0.250088   0.033028   7.572 1.38e-13 ***
## X5           0.742720   0.067505  11.002  < 2e-16 ***
## X6          -0.003967   0.012195  -0.325  0.74506
## X7           0.260312   0.012372  21.040  < 2e-16 ***
## X8           0.037054   0.012367   2.996  0.00285 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2979 on 604 degrees of freedom
## Multiple R-squared:  0.9123, Adjusted R-squared:  0.9113
## F-statistic: 897.7 on 7 and 604 DF,  p-value: < 2.2e-16
```

```
varImp(standard_model_lm_1)
```

```
## lm variable importance
##
##    Overall
## X7  100.00
## X5   51.54
## X3   34.98
## X1   24.22
## X2   18.80
```

```
## X8   12.89
## X6    0.00
```

```
# Predicting with standardization model on test
# standardized set

standard_lm_Y1_hat <- standard_model_lm_1 %>%
    predict(energy_test_S)

# Calculate RMSE and MSE
RMSE_Y1_lm_S <- rmse(standard_lm_Y1_hat, energy_test_S$Y1)
MSE_Y1_lm_S <- mse(standard_lm_Y1_hat, energy_test_S$Y1)
```

- **RESULTS**

```
## # A tibble: 5 x 3
##   method                                          RMSE    MSE
##   <chr>                                          <dbl>  <dbl>
## 1 Model 1- LINEAR MODEL OF Y1 Original data     2.65  7.01
## 2 Model 1- LINEAR MODEL OF Y2 original data     2.95  8.69
## 3 Model 2- RANDOMFOREST OF Y1 original data     0.416 0.173
## 4 Model 2- RANDOMFOREST OF Y2 original data     1.80  3.23
## 5 Model 1- linear model OF Y1 standardized data 0.262 0.0688
```

```
# Standardized linear model for Y2 without X4
standard_model_lm_2 <- train(Y2 ~ X1 + X2 + X3 + X5 +
    X6 + X7 + X8, data = energy_training_S, method = "lm")

summary(standard_model_lm_2)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.90078 -0.16396 -0.01854  0.14426  1.18593
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.005744   0.013881  -0.414    0.679
## X1          -0.786303   0.143105  -5.495 5.78e-08 ***
## X2          -0.803825   0.198632  -4.047 5.87e-05 ***
## X3           0.204359   0.038059   5.369 1.13e-07 ***
## X5           0.793132   0.077789  10.196  < 2e-16 ***
## X6           0.011654   0.014053   0.829    0.407
## X7           0.205585   0.014257  14.420  < 2e-16 ***
## X8           0.008377   0.014251   0.588    0.557
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3433 on 604 degrees of freedom
## Multiple R-squared:  0.8829, Adjusted R-squared:  0.8815
## F-statistic: 650.5 on 7 and 604 DF,  p-value: < 2.2e-16
```

```
varImp(standard_model_lm_2)
```

```
## lm variable importance
##
##    Overall
## X7 100.000
## X5  69.462
## X1  35.474
## X3  34.569
## X2  25.007
## X6   1.746
## X8   0.000
```

```
# Predicting with standardization model on test
# standardized set

standard_lm_Y2_hat <- standard_model_lm_2 %>%
    predict(energy_test_S)

# Calculate RMSE and MSE
RMSE_Y2_lm_S <- rmse(standard_lm_Y2_hat, energy_test_S$Y2)
MSE_Y2_lm_S <- mse(standard_lm_Y2_hat, energy_test_S$Y2)
```

- **RESULTS**

```
## # A tibble: 6 x 3
##   method                                          RMSE    MSE
##   <chr>                                          <dbl>  <dbl>
## 1 Model 1- LINEAR MODEL OF Y1 Original data      2.65  7.01
## 2 Model 1- LINEAR MODEL OF Y2 original data      2.95  8.69
## 3 Model 2- RANDOMFOREST OF Y1 original data      0.416 0.173
## 4 Model 2- RANDOMFOREST OF Y2 original data      1.80  3.23
## 5 Model 1- linear model OF Y1 standardized data 0.262 0.0688
## 6 Model 1- linear model OF Y2 standardized data 0.310 0.0960
```

4.5 RandomForest models with standardized data

```
# Standardized randomForest for Y1

standard_model_rf_1 <- train(Y1 ~ X1 + X2 + X3 + X4 +
    X5 + X6 + X7 + X8, data = energy_training_S, method = "rf")

standard_model_rf_1
```

```
## Random Forest
##
## 612 samples
##   8 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 612, 612, 612, 612, 612, 612, ...
## Resampling results across tuning parameters:
##
##   mtry  RMSE       Rsquared   MAE
```

```
##   2     0.12196653  0.9856240  0.09413136
##   5     0.06282660  0.9958909  0.04111101
##   8     0.06396915  0.9957938  0.04134681
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was mtry = 5.
```

```
varImp(standard_model_rf_1)
```

```
## rf variable importance
##
##    Overall
## X2 100.000
## X4  79.220
## X1  32.717
## X7  17.849
## X5   9.538
## X3   9.517
## X8   6.352
## X6   0.000
```

```
# Predicting with standardization model on test
# standardized set

standard_rf_Y1_hat <- standard_model_rf_1 %>%
    predict(energy_test_S)

# Calculate RMSE and MSE
RMSE_Y1_rf_S <- rmse(standard_rf_Y1_hat, energy_test_S$Y1)
MSE_Y1_rf_S <- mse(standard_rf_Y1_hat, energy_test_S$Y1)
```

- **RESULTS**

```
## # A tibble: 7 x 3
##   method                                          RMSE     MSE
##   <chr>                                          <dbl>   <dbl>
## 1 Model 1- LINEAR MODEL OF Y1 Original data     2.65   7.01
## 2 Model 1- LINEAR MODEL OF Y2 original data     2.95   8.69
## 3 Model 2- RANDOMFOREST OF Y1 original data     0.416  0.173
## 4 Model 2- RANDOMFOREST OF Y2 original data     1.80   3.23
## 5 Model 1- linear model OF Y1 standardized data 0.262  0.0688
## 6 Model 1- linear model OF Y2 standardized data 0.310  0.0960
## 7 Model 2- RANDOMFOREST OF Y1 standardized data 0.0421 0.00177
```

```
# Standardized randomForest for Y2

standard_model_rf_2 <- train(Y2 ~ X1 + X2 + X3 + X4 +
    X5 + X6 + X7 + X8, data = energy_training_S, method = "rf")

standard_model_rf_2
```

```
## Random Forest
##
## 612 samples
##   8 predictor
##
## No pre-processing
```

```
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 612, 612, 612, 612, 612, 612, ...
## Resampling results across tuning parameters:
##
##   mtry  RMSE       Rsquared   MAE
##   2     0.1978368  0.9609675  0.1425062
##   5     0.1908248  0.9633349  0.1161639
##   8     0.1959170  0.9614006  0.1176823
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was mtry = 5.
```

```
varImp(standard_model_rf_2)
```

```
## rf variable importance
##
##    Overall
## X5 100.000
## X1  94.709
## X4  66.797
## X2  42.488
## X7  13.484
## X3  10.543
## X8   3.098
## X6   0.000
```

```
# Predicting with standardization model on test
# standardized set

standard_rf_Y2_hat <- standard_model_rf_2 %>%
    predict(energy_test_S)

# Calculate RMSE and MSE
RMSE_Y2_rf_S <- rmse(standard_rf_Y2_hat, energy_test_S$Y2)
MSE_Y2_rf_S <- mse(standard_rf_Y2_hat, energy_test_S$Y2)
```

- **RESULTS**

```
## # A tibble: 8 x 3
##   method                                          RMSE     MSE
##   <chr>                                          <dbl>   <dbl>
## 1 Model 1- LINEAR MODEL OF Y1 Original data     2.65   7.01
## 2 Model 1- LINEAR MODEL OF Y2 original data     2.95   8.69
## 3 Model 2- RANDOMFOREST OF Y1 original data     0.416  0.173
## 4 Model 2- RANDOMFOREST OF Y2 original data     1.80   3.23
## 5 Model 1- linear model OF Y1 standardized data 0.262  0.0688
## 6 Model 1- linear model OF Y2 standardized data 0.310  0.0960
## 7 Model 2- RANDOMFOREST OF Y1 standardized data 0.0421 0.00177
## 8 Model 2- RANDOMFOREST OF Y2 standardized data 0.189  0.0358
```

My evaluation metrics RMSE and MSE get better value for both heating and cooling load when data was standardize; an significant improvement of the two models.

5. Conclusions and recommendations

In this second project capstone, I have really improve my skills in data visualization. I have trained two popular machine learning algorithms : linear regression and random forest.

I have also practice the importance of standardization of data for improving prediction models.

My models metrics evaluation shows that we can predict accurately with simplest machine learning algorithm such as linear regression and randomforest the energy efficiency of buildings. Buildings are energy consuming. Optimizing their efficiency allow a better climate protection.

This dataset and the exploration phase demonstrates that the glazing area and the orientation of buildings are not correlated to the outcomes : heating load and cooling load. However, with the linear regression model the glazing area is the most important variables. That's sound logical as glazing area can result in a lower energy consumption than opaque walls.

For future work, I will explore others machine learning algorithms that can be fitted to this type of data. I also have to perform references inclusion to .rmd report.

I aim to check sources of overtrain or overfit when I have standardize all variables. I will also explore options of standardizing some variables and not all of them. An other model training is to drop down some variables as the orientation of building(X6) to check how predictions should be accurate with and without some variables.

This work has been made by reading the following documents :

Estimation of Energy Performance of Buildings Using Machine Learning Tools