INF 442 PROJECT

Power Consumption Weather

May 12, 2024

Huu-Hoang Nguyen & Gabriel Pereira de Carvalho

CONTENTS

1 INTRODUCTION

This report presents a solution to the programming project **Power Consumption Weather** for the course INF442: *Algorithmes pour l'analyse de donnéss en C++* at École Polytechnique.

The code was implemented using the Python programming language and can be accessed on the project's Github repository.

2 DATA PREPROCESSING

2.1 Generating dataframes

First of all, we must convert all the data stored across multiple *.csv* files and transform it into dataframes which we can easily manipulate.

We merged the four *.csv* files containing data from the household into a single dataframe. Then we converted all dates to *DateTime* and all the numeric data from *object* data-type to *float* data-type.

We repeated the same procedure for the data relative to the weather stations, merging all 48 *.csv* files into a single dataframe. We also filtered data to keep only the data relative to the *Orly* station which is pertinent to the household.

2.2 Treating missing data

To investigate the presence of missing data in our dataframes, we used the *missingno* python package. We generated bar plots that show the number of values present in each column.

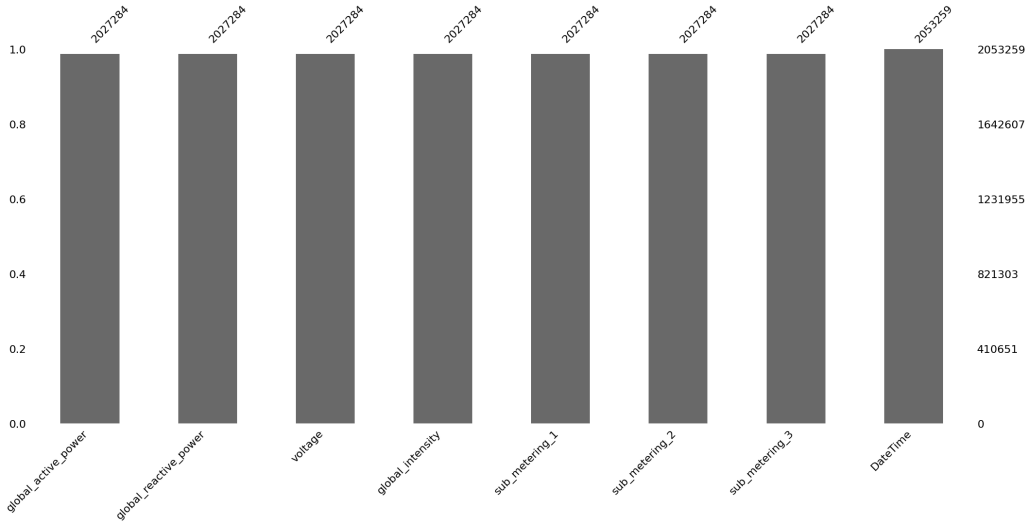


Figure 1: Bar plot with data present in household dataframe

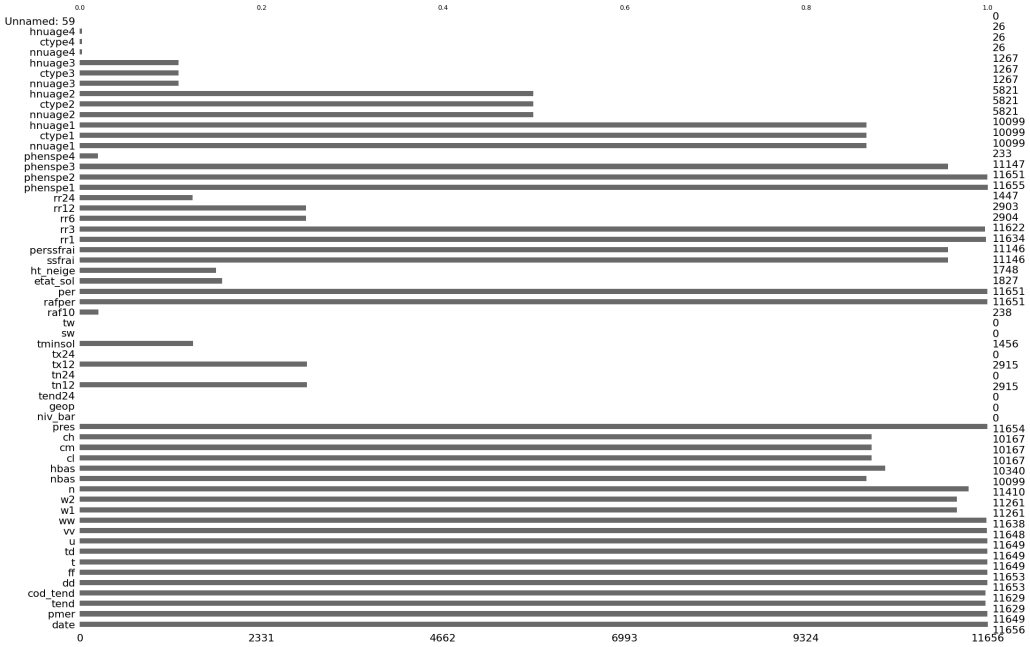


Figure 2: Bar plot with data present in weather station dataframe

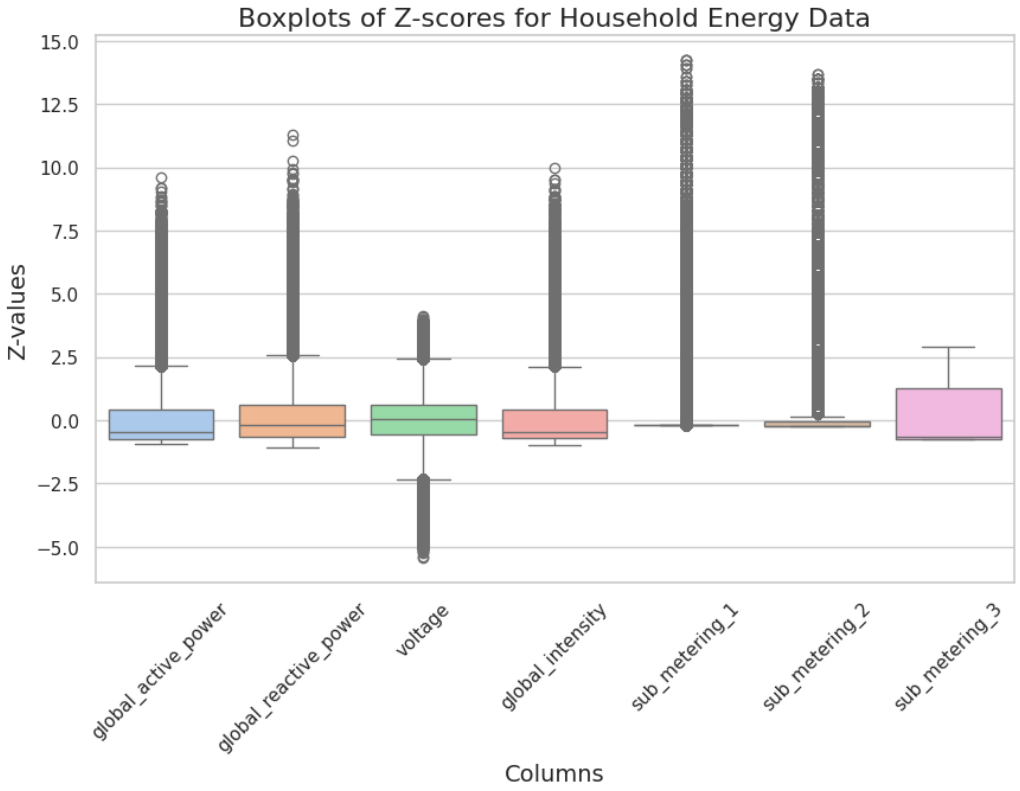
We observe that almost all rows are filled in the household dataframe. The plot shows very few missing data for us to interpolate. However, in the weather station dataframe, we see lots of missing data. Columns such as **tw**, **sw**, **tx24**, **tn24**, **tend24**, **geop** and **niv_bar** have more rows with missing data than with present data.

We decided that dropping such columns is better than attempting to backfill or interpolate because we do not have a lot of data to work with. In addition, the weather station data is

already collected at a lower frequency than the household data which makes this phenomenon even more consequential.

2.3 Treating outliers

We calculated z-scores for each feature in the dataframes and made boxplots in order to visualise their distribution for our data.



In order to eliminate outliers we utilised *winsorization*, replacing extreme values with the observations closest to them.

3 SEASONALITY DETECTION

Using the household dataframe, we calculated the energy consumption for each hour by summing over the `global_active_power` column.

Then, the *k-means* algorithm was used for clustering. We used the **elbow method** to determine the optimal value of $k = 3$.

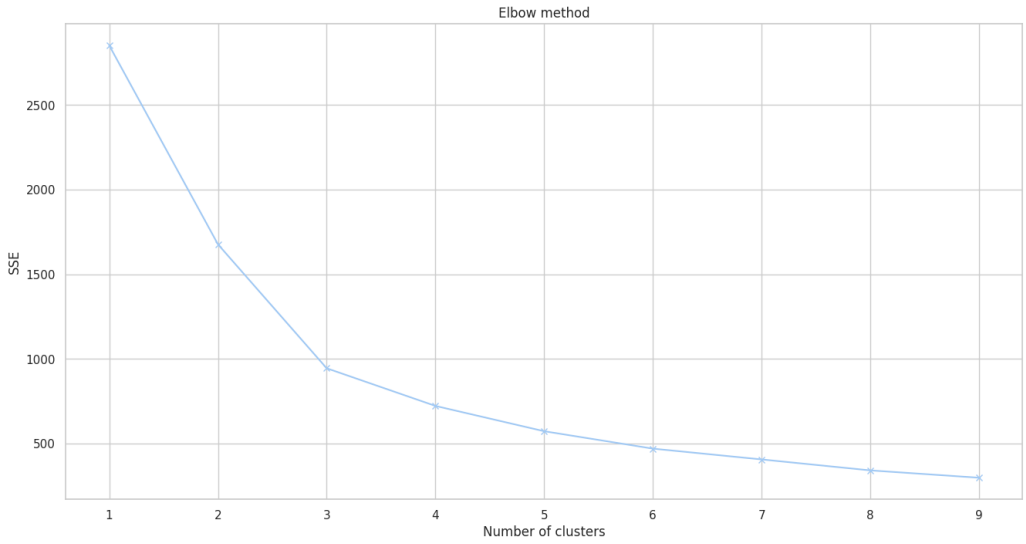


Figure 3: Elbow in min variance plot is at $k = 3$

Over a one year period, we observe 3 clusters characterized by low and high energy consumption corresponding with hot and cold seasons. The first cluster at the beginning of the year corresponds to a cold winter/spring season. The middle cluster corresponds to summer and the last cluster at the end of the year corresponds to autumn/winter season.

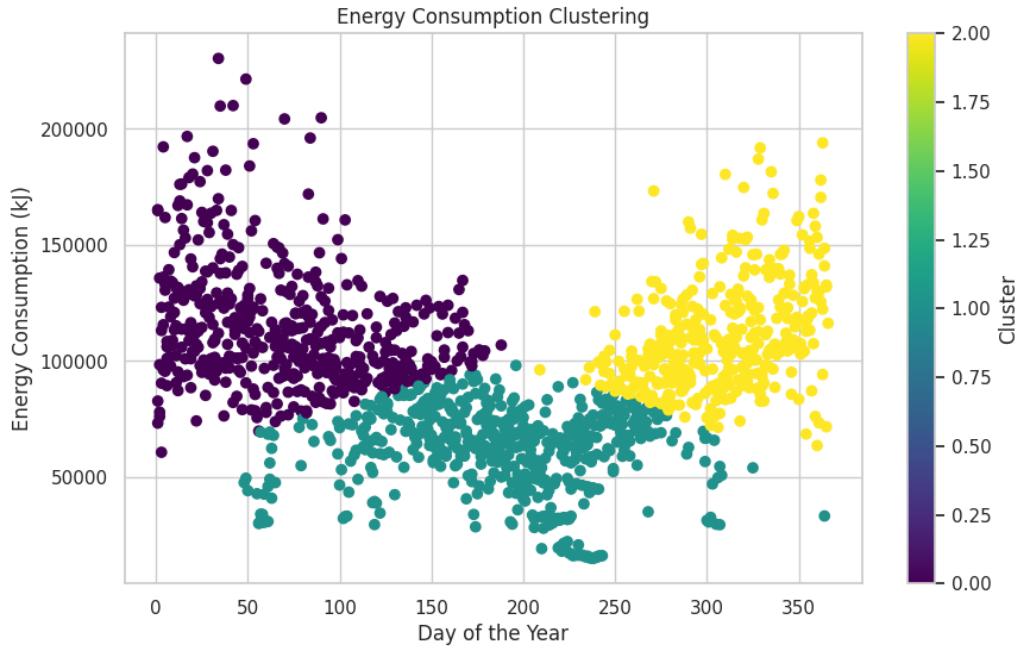


Figure 4: Clusters seem to follow cold/hot seasons of the year

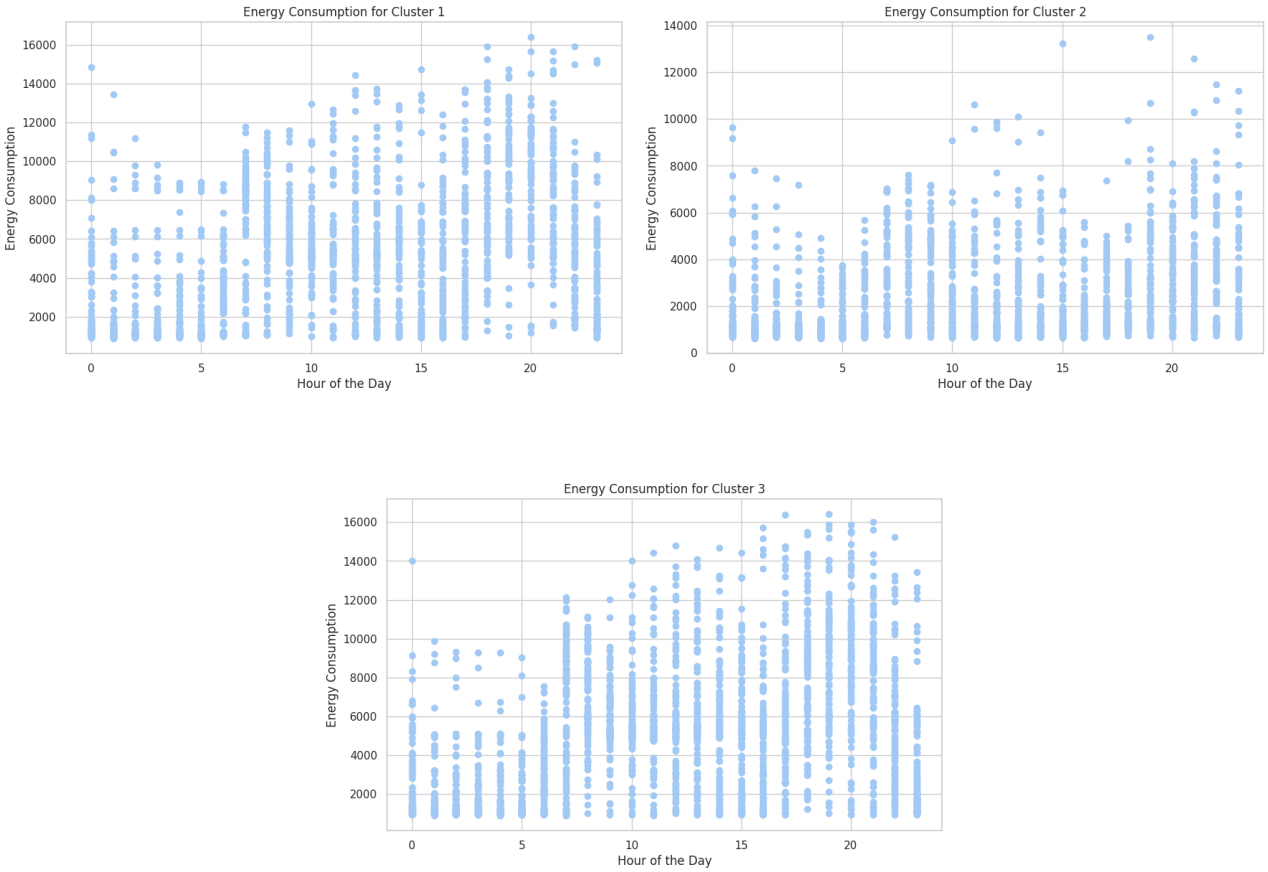


Figure 5: Distribution over one day for each cluster

4 DISTRIBUTION OF HOURLY CONSUMPTION

We plotted the distribution of energy consumption over a one day period for multiple days in each cluster, and then built histograms in order to analyse the frequency of each energy value and the underlying probability distribution.

In order to test for a gaussian distribution, we perfomed the **Kolmogorov-Smirnov test**. For all three of the clusters, the p-values obtained where value low. Therefore, we conclude that daily energy consumption does not follow a gaussian distribution.

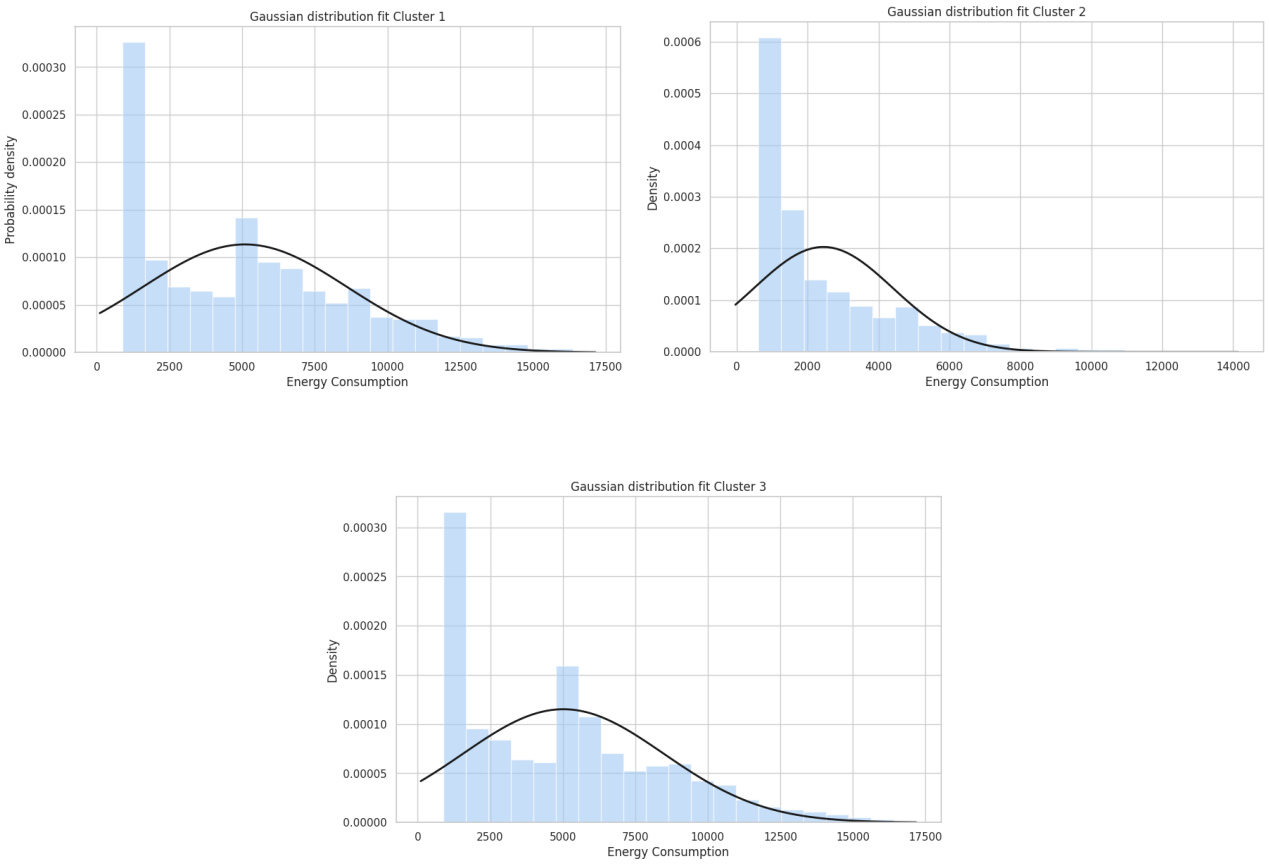
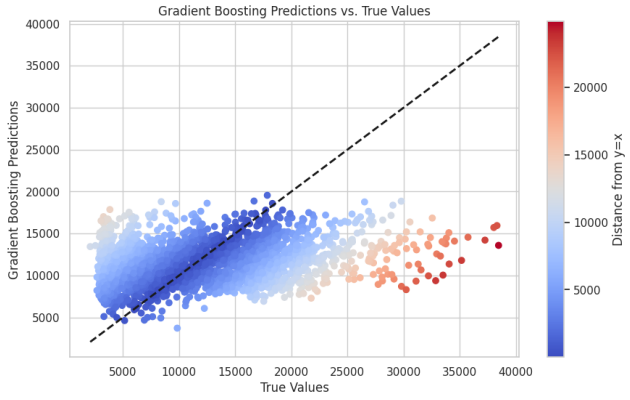


Figure 6: Gaussian fit for each cluster

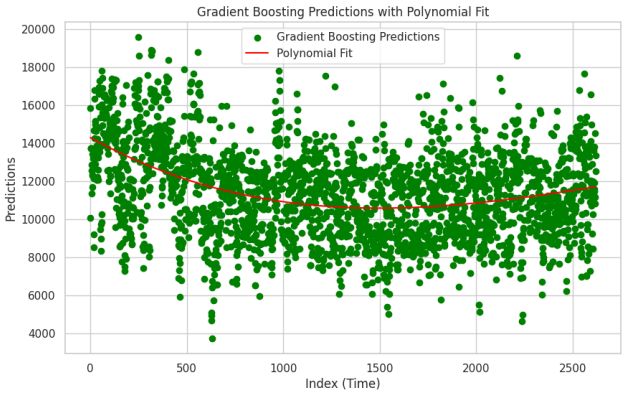
5
WEATHER INFLUENCE

We used the `merge_asof` method in the pandas library to integrate the household data with the weather station data into a single dataframe. The energy consumption obtained from the `global_active_power` by integrating over each 3 hour period was used as the y value for prediction and the features in the weather station dataframe were used as the x values. We split our data into training and test dataframes corresponding respectively to the periods 2007-2009 and 2010.

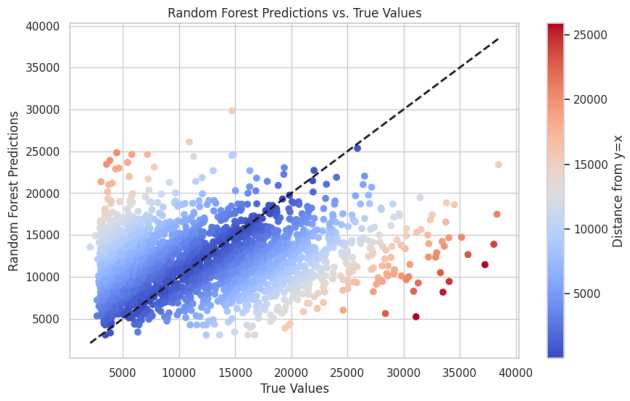
We used two different regression models for prediction: **Random Forest** and **Gradient Boosting**. We chose these models because they have strong performance in high dimensions.



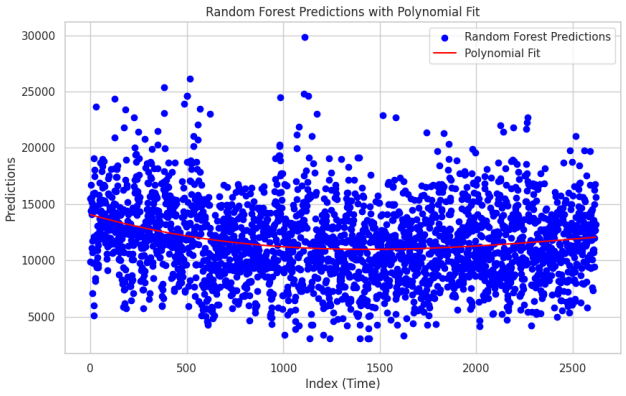
(a): Performance of Gradient Boost regressor on 2010 test data



(b): Scatter plot of Gradient Boost predictions on 2010 test data with a polynomial fit



(c): Performance of Random Forest regressor on 2010 test data



(d): Scatter plot of Random Forest predictions on 2010 test data with a polynomial fit

Feature	Importance
rr24	0.190920
etat_sol	0.144903
rr12	0.129627
rr6	0.122634
ctype3	0.055209
tend	0.043987
w2	0.040767
ctype2	0.038995
ctype4	0.038054
rr3	0.036395

Figure 7: Most important features for Random Forest regressor

Feature	Importance
etat_sol	0.292453
rr12	0.150750
tend	0.133651
ctype4	0.109710
rr24	0.088593
rr6	0.072265
rr3	0.030567
w2	0.023846
ctype3	0.020823
n	0.018048

Figure 8: Most important features for Gradient Boosting regressor

We observe a lower error(RMSE) for the Gradient Boost regressor, and in the scatter plot we can observe the seasonality as in the cluster scatter plot. We have higher predictions on average for the regions corresponding to clusters 1 and 3, and lower predictions on average for the cluster 2 region.

For each model, we extracted the ten most important features (characterized by bigger coefficients). We observe that rain levels (**rrN**); state of the terrain (**etat_sol**) and cloud type(**ctypeN** are important for both models which implies a high correlation between these factors and the household energy consumption.