TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies

Maksym Maliarov 179425IAIB

PREDICTING RESIDENTIAL ELECTRICITY CONSUMPTION USING MACHINE LEARNING ALGORITHMS

Bachelor's thesis

Supervisor: Margarita Spitšakova
PhD

Tallinn 2020

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Maksym Maliarov 179425IAIB

ELURUUMIDE ELEKTRITARBIMISE PROGNOOSIMINE MASINÕPPE ALGORITMIDE ABIL

Bakalaureusetöö

Juhendaja: Margarita Spitšakova
PhD

Tallinn 2020

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Maksym Maliarov

03.08.2020

Abstract

The main goal of this thesis is to build a ML (*Machine Learning*) model for predicting REC (*Residential Electricity Consumption*) based on the consumption patterns in the past. In order to efficiently solve this task and determine the possible determinants of REC, various data was gathered, and an analysis based on this data was conducted.

Python programming language and its data science modules were utilised to develop the software solution for solving the task of this thesis. The goals of the software are to help analyse relationships between REC and external variables, gather and process data, and to build and validate REC predictive models.

The result of this thesis is several developed and validated REC predictive models for the test dwelling and software that can be used for predicting REC of an arbitrary dwelling using various ML algorithms and factors that impact REC.

This thesis is written in English and is 29 pages long, including 7 chapters, 16 figures and 5 tables.

Annotatsioon

Eluruumide elektritarbimise prognoosimine masinõppe algoritmide abil

Käesoleva bakalaureusetöö põhieesmärgiks on masinõppe mudeli loomine eluruumide elektritarbimise prognoosimiseks varasemate elektritarbimise harjumuste põhjal. Selle ülesande lahendamiseks ja eluruumide elektritarbimise väliste mõjurite määramiseks oli kogutud erinevaid andmeid ning nende põhjal viidi läbi analüüs.

Antud töö probleemi lahendamiseks kasutati *Python*-i programmeerimiskeelt koos selle andmeteaduse tarkvarateekidega. *Python*-i ja andmeteaduse tarkvara põhjal loodi tarkvaraline lahendus, mille eesmärkideks on eluruumide elektritarbimise ja väliste mõjurite omavaheliste suhete analüüs. Teisteks eesmärkideks on välisandmete kogumine, töötlemine ja prognoosimudelite arendamine ning valideerimine.

Selle bakalaureusetöö tulemuseks on mitu arendatud ja valideeritud eluruumi elektritarbimise prognoosimudelit ühe eluruumi jaoks. Lisaks sellele on arendatud tarkvara, mida saab kasutada suvalise eluruumi elektritarbimise prognoosimiseks, kasutades erinevaid elektritarbimise mõjureid ning masinõppe algoritme.

Lõputöö on kirjutatud inglise keeles keeles ning sisaldab teksti 29 leheküljel, 7 peatükki, 16 joonist, 5 tabelit.

List of abbreviations and terms

REC	*Residential Electricity Consumption*
ML	*Machine Learning*
IoT	*Internet of Things*
SVM	*Support-Vector Machine*
GB	*Gradient Boosting*
API	*Application Programming Interface*
MAPE	*Mean Absolute Percentage Error*

Table of contents

List of figures

List of tables

1 Introduction

There is no denying the fact that electricity powers the modern and technological word of today. Electricity consumption has very quickly become an integral part of people's lives. Given such a high demand for electricity these days, different techniques for reducing consumption are developed and applied at a fast pace.

The goal of this thesis is to explore what variables play a role in REC, combine the best of them and build REC predictive models with reasonable accuracy. In this thesis REC prediction problem is treated as regression problem which is solved by using various ML algorithms. The ones that suit the most for this problem were chosen, their performance has been measured and the results that they produced have been validated.

1.1 Problem statement

Power demand predictive models are developed by electricity suppliers to optimize operational costs of business. These predictions are usually made by analysing historical demand data of the entire customer base and building corresponding predictive models. These models then can be used for effective day-ahead or intraday trading of energy thus minimizing the overall costs for end users and suppliers themselves.

As the number of electricity-drawing appliances is only growing, it has become crucial to optimize and minimise the overall consumption of power as much as possible. Furthermore, devices that people use on a daily basis have become a lot "smarter" over the past few years. *IoT* (*Internet of Things*) enabled devices and home automation systems can utilise electricity consumption prediction data to cut running costs. Knowing expected REC in the future, for example, it is also possible to manage heaters and lighting appliances more efficiently. Since the price for a unit of power is not constant and depending on external factors it keeps changing throughout a day, more accurate predictions result in lower costs for end consumers.

The idea of this thesis is to take a different approach to REC predictions by analysing consumption data of one specific dwelling rather than analysing electricity consumption of numerous unrelated dwellings together. Since each customer has their unique pattern of electricity consumption, this approach is more beneficial for end consumers.

1.2 Goals

The main goal of this thesis is to build a REC predictive model for a specific dwelling. In order to accomplish this, variables that relate to residential electricity need to be determined. Different sets of these variables could be used to build various predictive models using ML algorithms. Then the quality of these models should be validated and the best among them will be chosen.

In order to fulfil the goal of this thesis, specific software was developed. This software helps to analyse REC patterns and builds unique predictive models knowing only historical data of REC ahead of time.

The software was designed to meet the following functional requirements:

- Pre-process, re-structure, and re-format historical REC data from electricity supplier.
- Visualise REC data.
- Clean data and handle outliers.
- Build custom input variables for building and validating ML models.
- Explore relation between input variables and REC.
- Split data into training and testing sets.
- Wrap ML algorithms in autoregressive model.
- Build predictive models, visualise and compare their performance depending on ML algorithm used.
- Visualise learning progress of ML models.

- Find optimal hyperparameters for ML algorithms.
- Validate the quality of predictions.
- Save developed predictive models.

1.3 Thesis structure

This thesis document has a total of seven chapters.

In the first section the functional requirements of software are set. The task of the thesis is defined, and the goals are set.

In the second section the methodology of tackling the problem is described. It includes the selection of tools, the approach taken to solve the problem and the test subject overview.

The third section is about gathering and processing data that is used for building REC predictive models using ML algorithms.

In the fourth section relationships between REC and selected input variables (features) for ML algorithms are analysed.

In the fifth section some aspects of building predictive models are described. The strategy for splitting data is explained and the chosen ML algorithms are overviewed.

The sixth section is about demonstrating, and analysing achieved results. It also covers the chosen methodology for validating the performance of developed models and describes how these models could improve.

The seventh section is the summary of this thesis.

2 Methodology

REC prediction problem can be treated as regression problem. For one or several input variables, one continuous output value needs to be calculated. Knowing what variables are the most relevant to the target variable (REC) it should be possible to calculate (predict) it by training a ML regression algorithm. In this thesis three different classes of ML algorithms were tested: *elastic net*, *SVM* (*Support-Vector Machine*) and *GB* (*Gradient Boosting*).

2.1 Test dwelling overview

It is worth mentioning that given correct historical data the software written for this thesis could be reused to work with data of an arbitrary dwelling. All the data for this thesis was obtained and analysed for a dwelling located in Eastern Estonia. It is a two-room flat with two residents. This flat does not have electric heaters and uses central heating as the main source of heating. Neither boilers nor air conditioning units are installed in the flat.

2.2 Factors of residential electricity consumption

Prior to building ML models, a correct set of factors that explicitly or implicitly affect REC needs to be selected. In this thesis the following factors were analysed [1]:

- Past consumption.
- Daylight duration.
- Type of day (weekday, weekend, holiday).
- Season (summer, autumn, winter, spring).
- Weather temperature.

The analysis of the abovementioned determinants was performed and their relevancy to the test dwelling was estimated.

2.3 Tools

When it comes to ML problems, there are several different development environments that can be chosen. Each environment has its own upsides and downsides. In data science *Python* with its outstanding data science ecosystem has become over the last couple decades a first-class tool for scientific computing tasks, including the analysis and visualisation of data [2].

Other viable options for solving the problem of this thesis were using either *R* or *C++* languages. However, while reviewing the functional requirements of the software, the decision to use *Python* has been made. *Python* is general purpose programming language and hence is more flexible than *R*, which is geared more towards statistical computing and graphics [3].As for *C++*, it has been assessed as too low-level for the given task, which inevitably would add unnecessary complexity to the process of writing the software.

Python has excellent data science ecosystem with *scikit-learn* being the option of choice for the majority of ML tasks [4]. It offers comprehensive documentation and covers theoretical aspects of ML. *scikit-learn* has been chosen as the module for building and validating ML models.

The following tools were utilised for fulfilling the goal of this thesis and meeting the functional requirements of the software:

- *scikit-learn* for building and validating ML models.
- *pandas* for storing and manipulating data.
- *matplotlib* for visualising data.
- *numpy* for performing mathematical operations on data and operations on multidimensional arrays.
- *requests* for working with RESTful API-s (*Application Programming Interface*).

- *jupyter notebook* for putting all the pieces of software together and experimenting.

3 Data gathering and processing

Prior to building predictive models, it was essential to gather and process the data of selected determinants of REC.

3.1 Gathering data

Historical electricity consumption data of the test dwelling was manually gathered from an Estonian electricity supplier [5]. Being unprocessed this data is in a form of an hourly timestamp, day/night rate and corresponding electricity consumption all in a time frame of one past year.

Table 1. Historical electricity consumption data

Kuupäev ja tunni algus	**Tunnus**	**38ZEE-07621137-K**
01.02.2019 0:00	öö	0,342
01.02.2019 1:00	öö	0,309
…	…	…
31.01.2020 22:00	päev	0,05
31.01.2020 23:00	öö	0,06

Daylight duration data was gathered via public *Sunrise Sunset* RESTful API [6]. This API expects latitude and longitude of an arbitrary location and a date for which the data should be returned as parameters. Response of this API contains *day_length* value which was used as daylight duration data.

Type of day data was gathered using both the historical REC data and *Calendarific* RESTful API [7]. Whether a day is a weekday, or a weekend was determined from the historical data. Holidays, however, could not be guessed from historical data and using external sources for this type of data was a necessity. *Calendarific* API expects a year and a country as parameters and responds with various information about holidays for a given year and country.

Season data was gathered by transforming the historical REC data.

Weather temperature data was gathered using *Meteostat* RESTful API [8]. This API was used to find the nearest weather station to the location of the test dwelling and collect temperature data from that station for a given interval.

3.2 Processing data

REC data required re-structuring and re-formatting before it could be analysed. Two different approaches to data re-structuring were considered - using hourly data as shown in *Table 1* or resampling data to daily consumption values by summing all hourly consumption values of a corresponding day. Further research has shown that using daily values is overall a better idea as this approach resulted in more accurate predictions. The reason for such results is that hourly consumption data of only one dwelling is very inconsistent and it does not correlate with any of the selected determinants of REC which was not the case with daily data. After re-structuring and re-formatting, consumption data was in a form of 365 consequent days and their corresponding consumption values in kW • h units of energy.

Table 2. Pre-processed historical REC data

Date	**Consumption (kW • h)**
2019-02-01	9.915
2019-02-02	6.260
…	…
2020-01-30	3.760
2020-01-31	3.180

The next stage of data pre-processing was to transform gathered data of REC determinants to the similar daily format as existing consumption data.

4 Data analysis

In this section both gathered data and relationships between REC and selected REC determinants are analysed. Based on this analysis different sets of determinants (features) are combined to form input values for ML algorithms. The analysis is based on the test dwelling described in *2.1* section of this thesis.

4.1 Outliers

An outlier in data can be defined as "an observation which appears to be inconsistent with the remainder of that set of data" [9]. Many ML algorithms are not resistant to outliers. If kept as is, these observations may degrade the model performance, so they must be handled in one way or another.

While exploring historical REC data, some abnormally low and high consumption values were noticed.

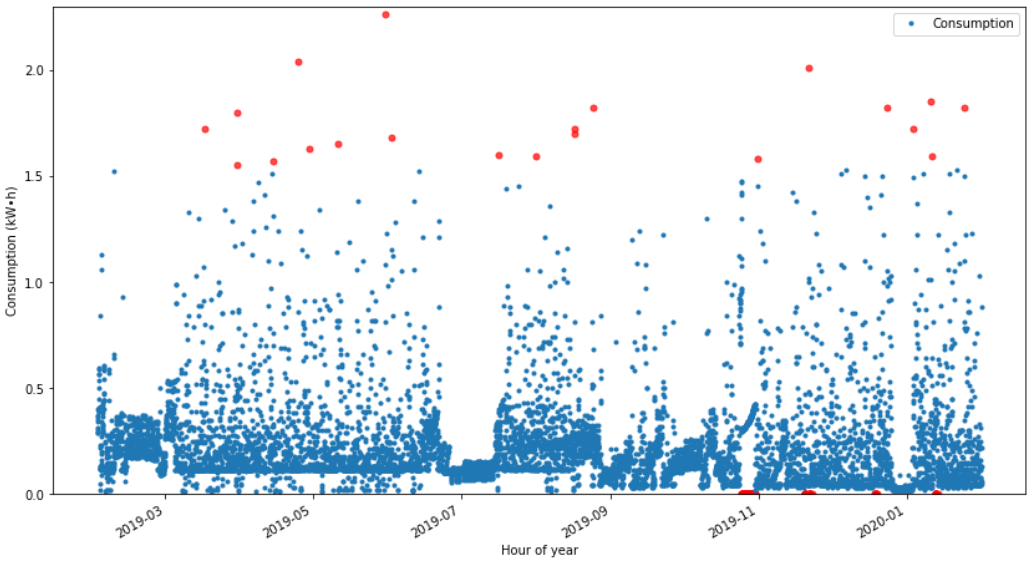

Figure 1. Hourly consumption over a year

Due to this issue, the decision to develop some outliers' detection and handling technique was made.

The chosen outlier detection technique was inspired by another statistical outlier detection technique of using interquartile range. The idea is to use 0.9975 quantile as the upper bound and 0.001 quantile as the lower bound on the initial hourly consumption data and 0.99 quantile as the upper bound and 0.01 quantile as the lower bound on the daily data. The reason for choosing such low values for the lower bound and such high values for the upper bound is to preserve data integrity and find only extreme cases of outliers.

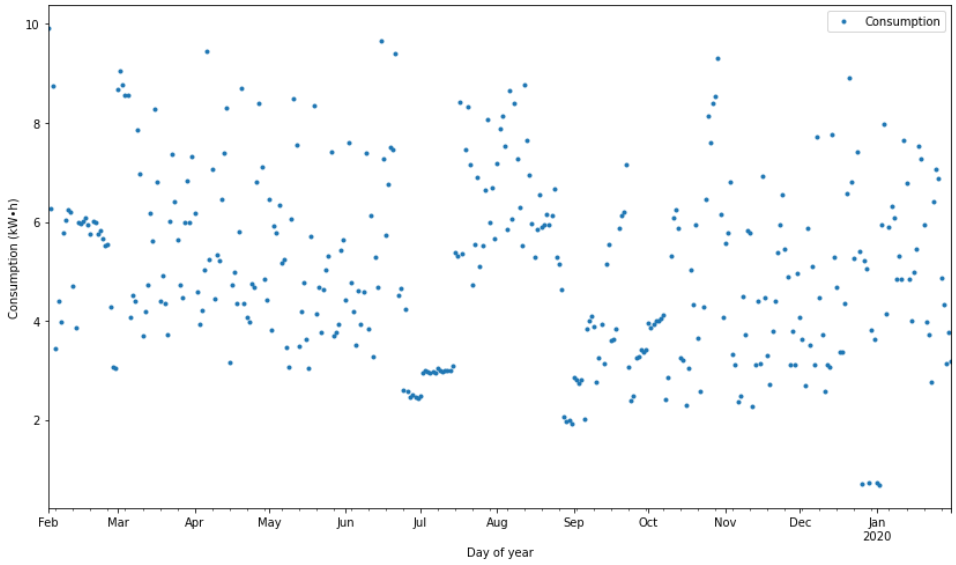

Figure 2. Daily consumption over a year (handled outliers)

Given the fact that historical REC data is ordered time series, just removing outliers was not an option as that would destroy data ordering, which was essential for feature engineering process. Because of that the mean value of previous observations over 7 days was chosen as the new value of an outlier.

4.2 Feature engineering

Feature selection is an integral part of solving any ML task. In order to have maximize the accuracy of predictions and minimize bias, it is crucial to select only features that have effect on REC. In this subsection relationships between REC and other variables are analyzed.

4.2.1 Past consumption

Consumption data for a given day is impacted by consumption values in the not-too-distant past.

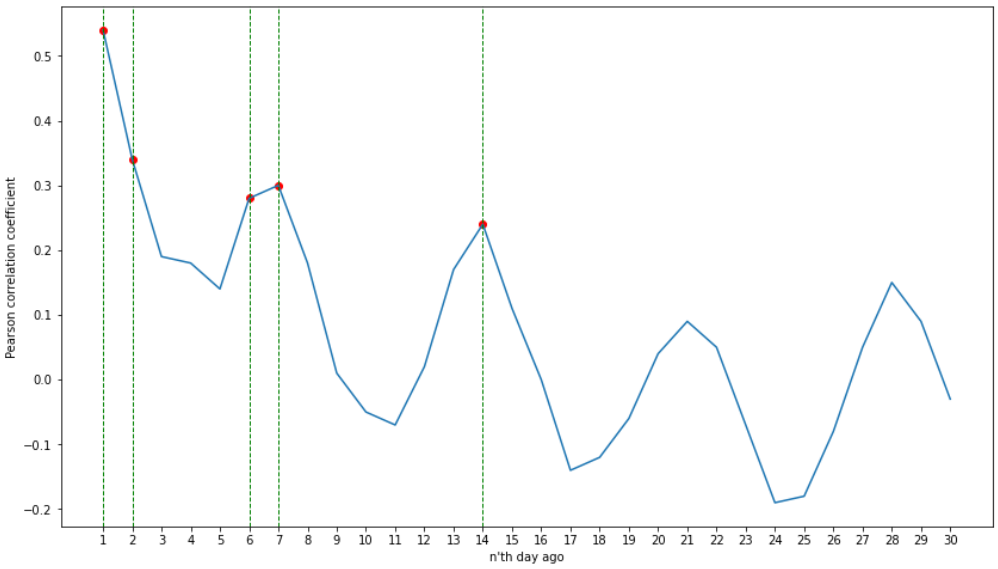

Figure 3. Past consumption correlation

1 day ago

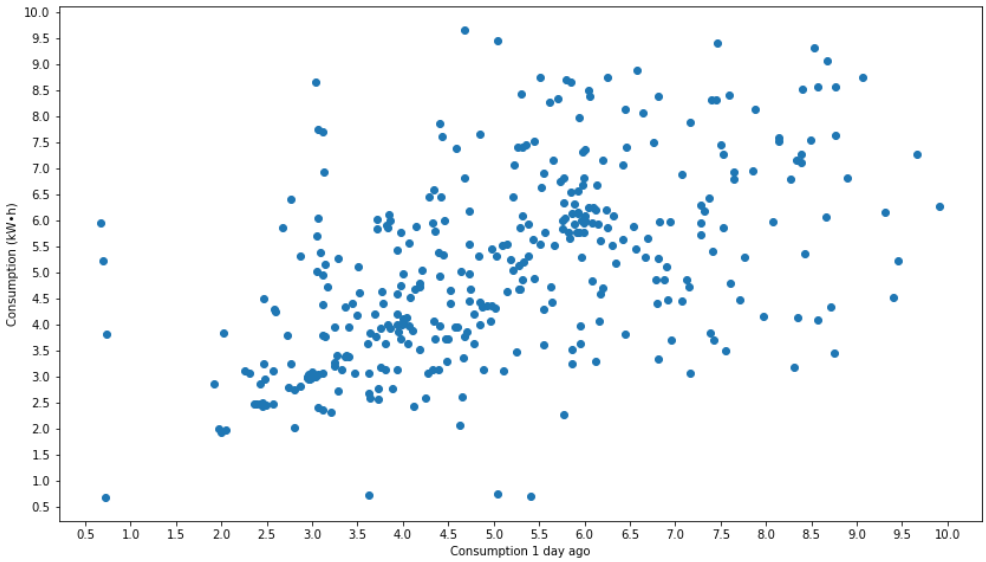

Figure 4. Consumption vs. Consumption 1 day ago

Consumption values have the highest autocorrelation with values from 1 day ago. The Pearson coefficient of correlation is *0.54* for this feature.

1 week ago

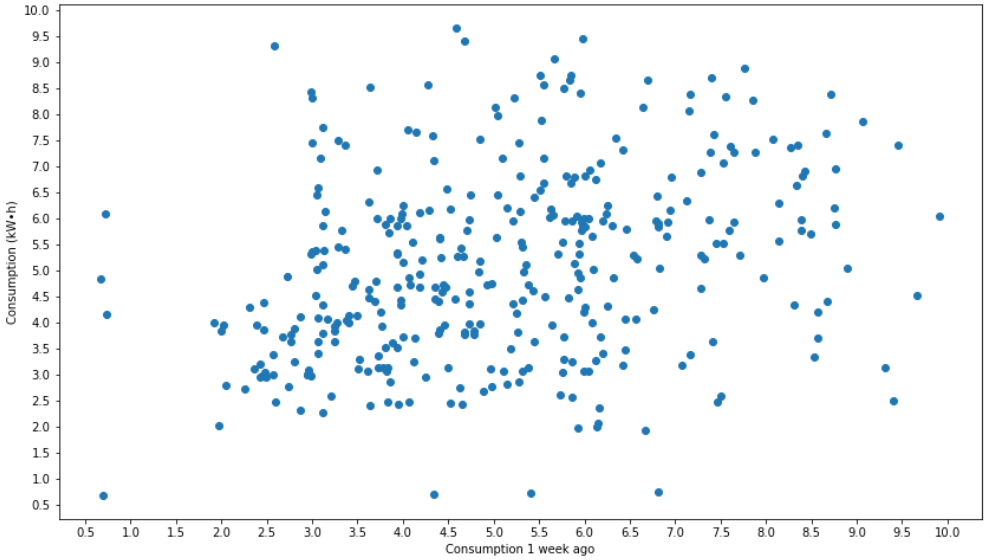

Figure 5. Consumption vs. Consumption 1 week ago

Consumption values from 1 week ago is top 2 feature of past consumption in terms of correlation coefficient. The coefficient of correlation is *0.3* for this feature.

2 weeks ago

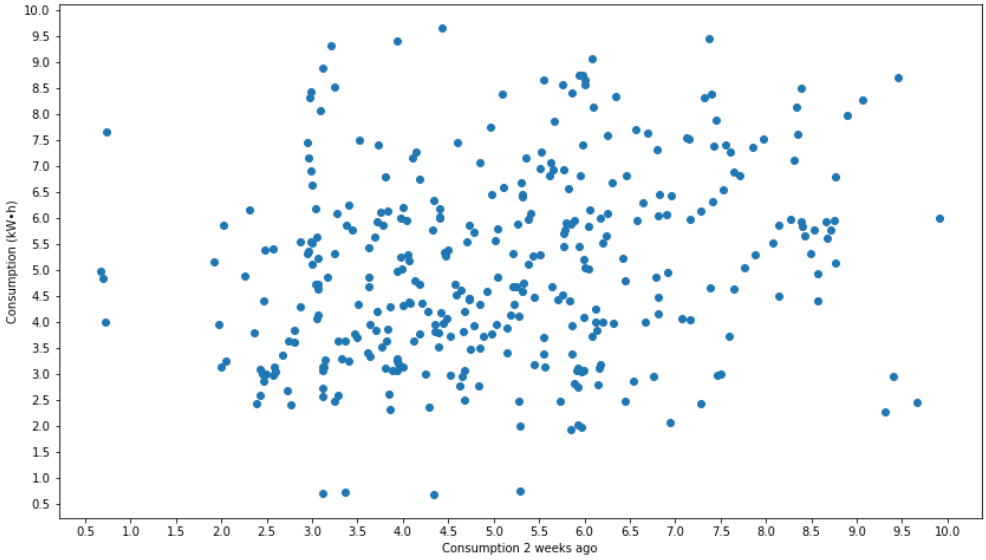

Figure 6. Consumption vs. Consumption 2 weeks ago

The coefficient of correlation of this feature is *0.25*. Although this feature does not have the next best correlation coefficient (slightly outperformed by 2 days ago with correlation coefficient of *0.32*), it should be valued more as there appears to be a strong weekly seasonality in the consumption data. The further analysis has shown that opting for day 2 or day 6 values would only add unnecessary noise to the predictive model.

4.2.2 Daylight duration

Daylight duration feature is the time from sunrise to sunset for a given day. When selecting this feature as a determinant of REC, the idea was that the longer daylight is present the less artificial lighting people will use which was expected to result in less electricity consumption overall.

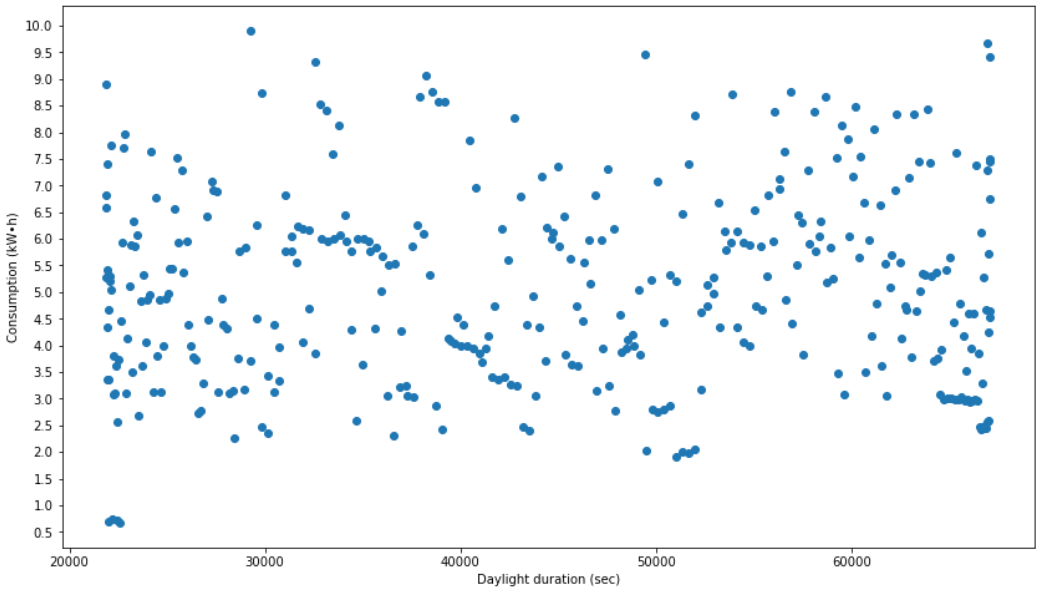

Figure 7. Consumption vs. Daylight duration

As seen in the *Figure 7*, there seems to be no correlation between this feature and consumption values. This is probably because of the efficiency of modern artificial lighting and its relatively low power consumption compared to other home appliances. The correlation coefficient of this feature is only *0.07* and therefore should not be used as a feature for ML algorithms.

4.2.3 Type of day

As people tend to have different daily routines depending on the type of day, electricity consumption values vary depending on the type of day. For each day of week this feature uses relative weights from 0 to 1 with 1 being the type of day with the highest mean consumption among all days of week [1]. The same strategy is used for calculation of holidays weight. They receive their own weight which results in total of 8 (7+1) different weights representing this feature. The coefficient of correlation of this feature is *0.38.*

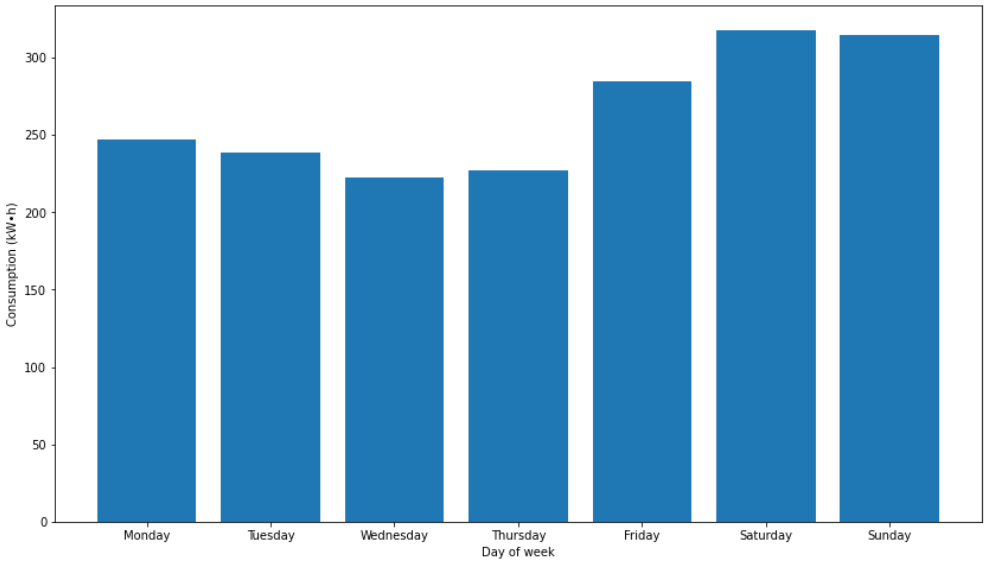

Figure 8. Consumption by day of week

4.2.4 Season

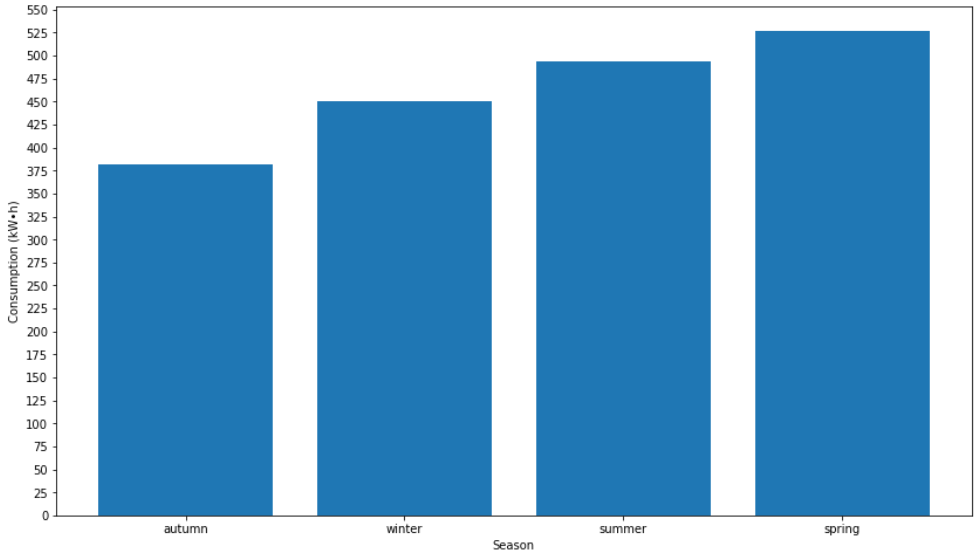

Figure 9. Consumption by season

There appears to be some seasonality for the given data which is caused by different patterns of electricity depending on the season. This feature uses the same weight algorithm [1] as *type of day* feature resulting in 4 different weights for summer, autumn, winter and spring. The correlation coefficient between this feature and electricity consumption is *0.25*.

4.2.5 Weather temperature

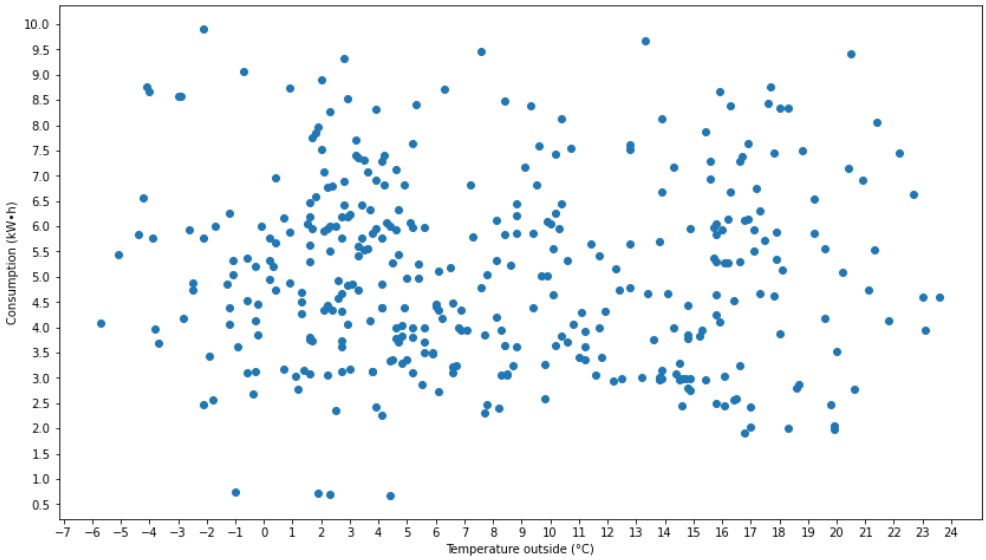

Figure 10. Consumption vs. Weather temperature

The *Figure 10* illustrates that weather temperature does not correlate with electricity consumption of the test dwelling. As already mentioned in *2.1* section, the test dwelling does not have boilers, electric heaters or air conditioning units installed. For dwellings that do have at least some of the abovementioned units, this feature should offer more value. The coefficient of correlation is only *-0.03* for this feature and therefore it will not be used in the predictive model for the test dwelling

4.3 Feature sets

In this subsection features are combined into two different sets which were used for building and validating predictive models. The features for these sets were selected based on their correlation with REC and the accuracy of their predictions. The first set includes 5 features while the second only 3.

Set #1

The following features were selected for set #1: *consumption 1 day ago, consumption 2 days ago, consumption 1 week ago, consumption 2 weeks ago, type of day*.

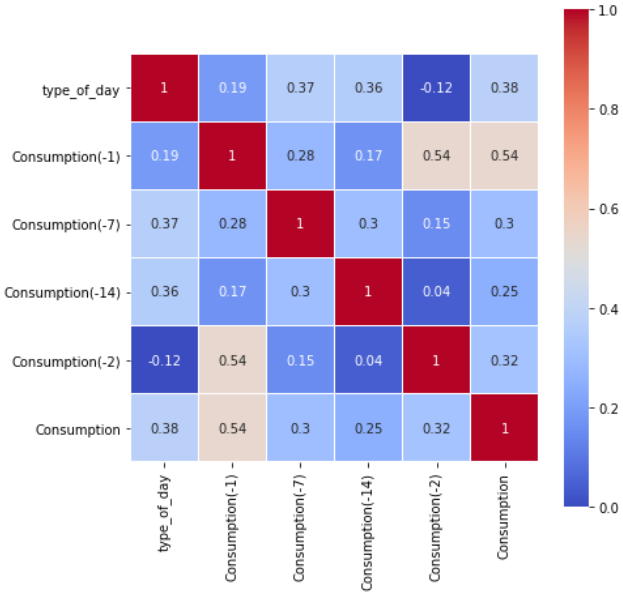

Figure 11. Correlation heat matrix for feature set #1

Set #2

The following features were selected for set #1: *consumption 1 week ago, consumption 2 weeks ago, type of day*.

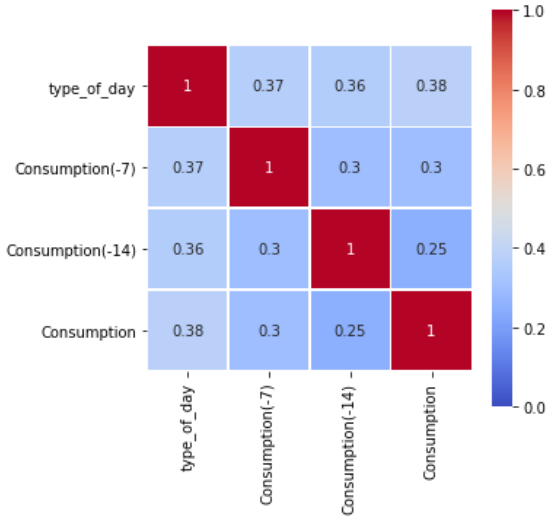

Figure 12. Correlation heat matrix for feature set #2

5 Building predictive models

In this section the process of building predictive models using three different regression estimators from *scikit-learn* is described. Also, it covers the data splitting technique and autoregression implementation used for making predictions into the future.

5.1 Data splitting

For the task of this thesis, it was important to be mindful of choosing the strategy for splitting data into training and testing sets. Usually it is done by using shuffling methods of splitting data. However, given the fact that our data is time series and there are some lagged variable features such as past consumption selected, the main goal was to avoid breaking the ordering of data. In order to accomplish that, time series cross-validator *TimeSeriesSplit* provided by *scikit-learn* module was utilised. Its objective is to split time series data samples in training and testing sets [4]. For each next consecutive split, this splitter combines all data from previous splits with new data. As it provides an option to set the number of splits, this was useful for estimating the optimal size of training set for REC prediction problem.

Table 3. An example of 3 splits and 15 observations

Split	**Training data indices**	**Testing data indices**
1	0, 1, 2, 3, 4, 5	6, 7, 8
2	0, 1, 2, 3, 4, 5, 6, 7, 8	9, 10, 11
3	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11	12, 13, 14

When analysing how the size of training set affects the accuracy of predictive models, it was noticed that using a lot of training samples does not result in higher accuracy. On the contrary, longer time interval of training data usually caused degraded performance of the predictive models. Using smaller test set but with only recent data led to more consistent accuracy overall.

5.2 Autoregression

There was a problem with using lagged features for predicting REC values in the future. The problem was that the number of days for which a prediction could be made was limited by the feature with the lowest lag value. For instance, if consumption from one day ago (*t-1*) is a feature, then it is possible to predict only for today (*t*). If a prediction for the next day (*t+1*) is needed, then we would need to know the consumption of today (*t*) ahead of time, which is not possible to do. In order to tackle this problem a custom autoregressive model was implemented. The idea of this model is to instead of relying on static values of lagged features, it updates all the features after each prediction by using that prediction as the actual value for a given lagged feature.

5.3 Algorithms

When it comes to solving ML problem, one of the most important steps is to choose the right algorithm. It is usually not the easiest task, since it requires a lot of testing and tuning of hyperparameters. As a part of this thesis, several supervised ML algorithms were tested for suitability for solving REC prediction problem. This section gives an overview of three different classes of regression estimators which could be seen among the best performers for REC prediction.

5.3.1 Elastic net

Elastic net is a linear model that uses l_1 and l_2 norm regularization of the coefficients [4]. In ML regularization is the process of preventing overfitting of models by controlling the growth of their coefficients [10]. “Elastic-net is useful when there are multiple features which are correlated with one another” [4].

scikit-learn implementation of elastic net estimator was used for building the predictive model. This implementation allows to tweak *l1_ratio* and *alpha* hyperparameters. The penalty is controlled by *alpha* parameter. The regularization is controlled by *l1_ratio* parameter. When *l1_ratio*=0, *l2* norm regularization is used, when *l1_ratio*=1, *l1* norm regularization is used [4]. For $0 < l1\_ratio < 1$, a combination of both regularizations is used [4].

5.3.2 Support-vector machine

The idea of a support vector machine is to construct the most optimal separating hyper-plane for data vectors of different classes [4]. SVM was initially used for solving classification problems, but its approaches were also applied to regression problems.

scikit-learn implementation *SVR* of SVM regression was used as the estimator for the predictive model. This implementation provides several tweakable hyperparameters, among which are *kernel*, ε, *C* and some other less relevant ones. The testing has shown that SVM with linear kernel is the best option for this problem. SVM regression uses ε-insensitive loss function which is used to control the penalty for errors [11]. For this SVM, small ε value means that it should use more support vectors for a higher quality separation in order to avoid errors [11]. In SVMs *C* constant is the means of controlling regularization and is a trade-off between minimizing training error and maximizing the width of the separating hyperplane [11].

5.3.3 Gradient boosting

GB is a tree-based ML method. Its goal is to iteratively improve predictions by combining additional trees that correct the mistakes made in previous iterations [12]

scikit-learn implementation *GradientBoostingRegressor* of GB was used for building the predictive model. GB performance depends on a variety of parameters such as a loss function, number of trees, maximum depth and a learning rate. In order to select the optimal hyperparameters for the given task, *GridSearchCV* provided by *scikit-learn* cross validation method was utilised.

6 Results and validation of predictive models

In this section the results of REC predictive models that were built using elastic net, SVM and GB are provided, and their accuracy is measured across two feature sets.

The accuracy of predictions was measured using MAPE (*Mean Absolute Percentage Error*) error metric. Since the quality of predictions was varying depending on the sizes of training and testing sets, cross-validation testing method was used to provide a more general overview of the quality of predictive models. Also, this section demonstrates a standard use case, when the model is trained for 30 days and then is used to predict REC for the upcoming 14 days.

6.1 Standard use case

The models were built using data from 44 recent days. Since in both feature sets there is a *consumption 2 weeks ago* feature, this resulted in 30 days of training. After the training, the models were asked to make predictions for the upcoming 14 days.

Feature set #1

Features: *consumption 1 day ago*, *consumption 2 days ago*, *consumption 1 week ago*, *consumption 2 weeks ago*, *type of day*.

Table 4. Standard use case validation. Feature set #1

Algorithm	Train size (days)	Train error (MAPE)	Test size (days)	Test error (MAPE)
Elastic net	30	14.79	14	14.05
SVM	30	15.77	14	13.99
GB	30	7.58	14	13.01

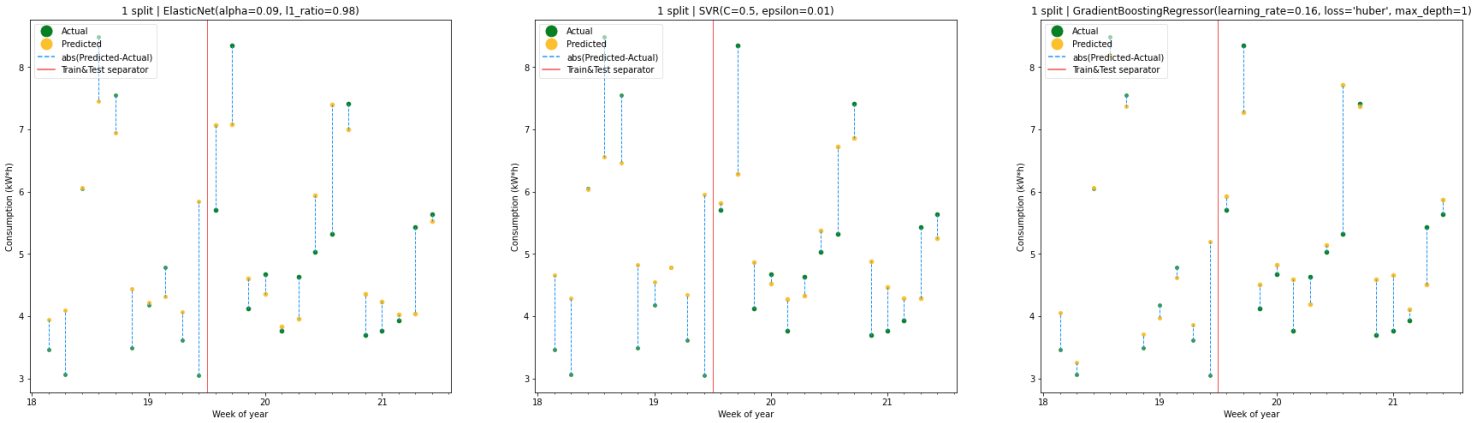

Figure 13. Standard use case validation. Feature set #1

Feature set #2

Features: *consumption 1 week ago*, *consumption 2 weeks ago*, *type of day*.

Table 5. Standard use case validation. Feature set #2

Algorithm	Train size (days)	Train error (MAPE)	Test size (days)	Test error (MAPE)
Elastic net	30	15.20	14	14.47
SVM	30	14.21	14	12.81
GB	30	8.83	14	13.15

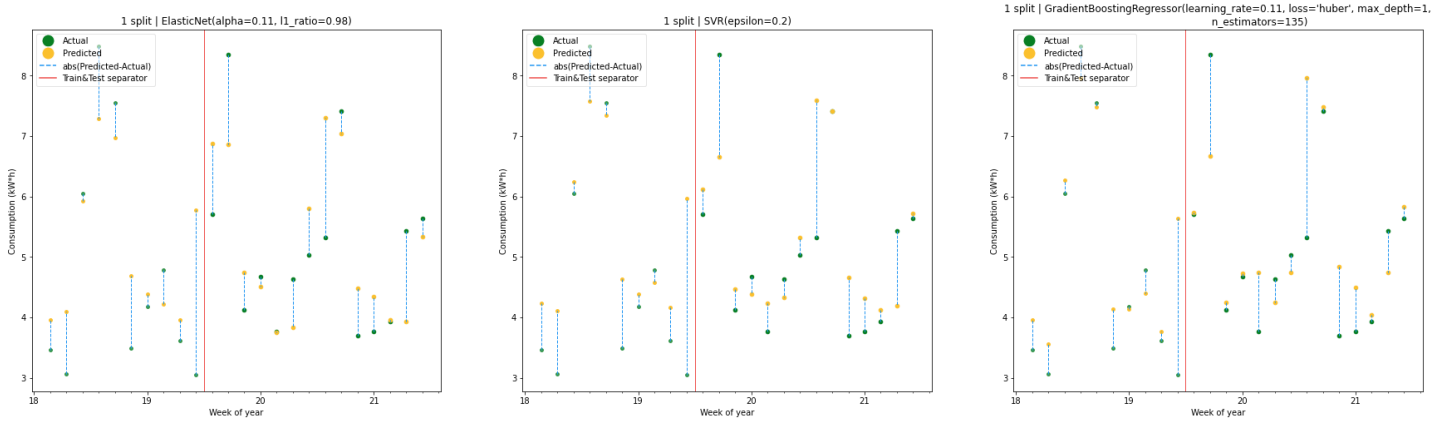

Figure 14. Standard use case validation. Feature set #2

6.2 Cross-validation

K-Fold with *k*=5 was used for cross-validation. This type of cross-validation splits data into *k* equally sized folds and on each iteration selects one of these bins as a testing bin with remaining being training bins. In order to better understand performance of the

models, the visualisation of their learning curves was implemented. The green and red areas on a learning curve plot visualise the standard deviation of models' errors.

Feature set #1

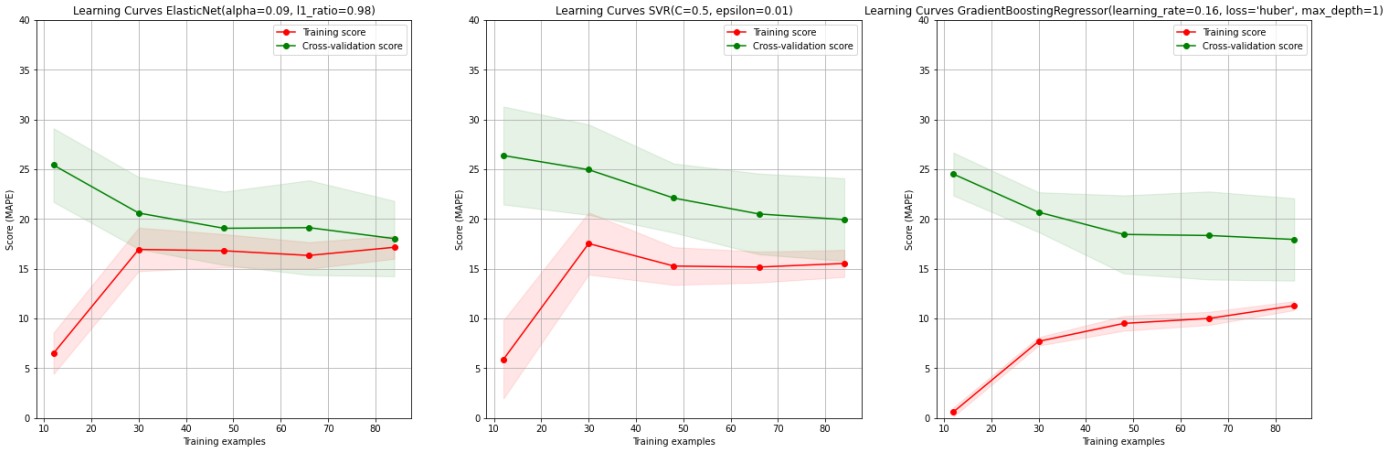

Figure 15. Cross-validation. Feature set #1

The above figure demonstrates that in terms of cross-validation MAPE elastic net and GB performed almost equally well. As for SVM, it did not do as well as other algorithms. In addition, the standard deviation of the quality of these predictions was a bit higher.

Feature set #2

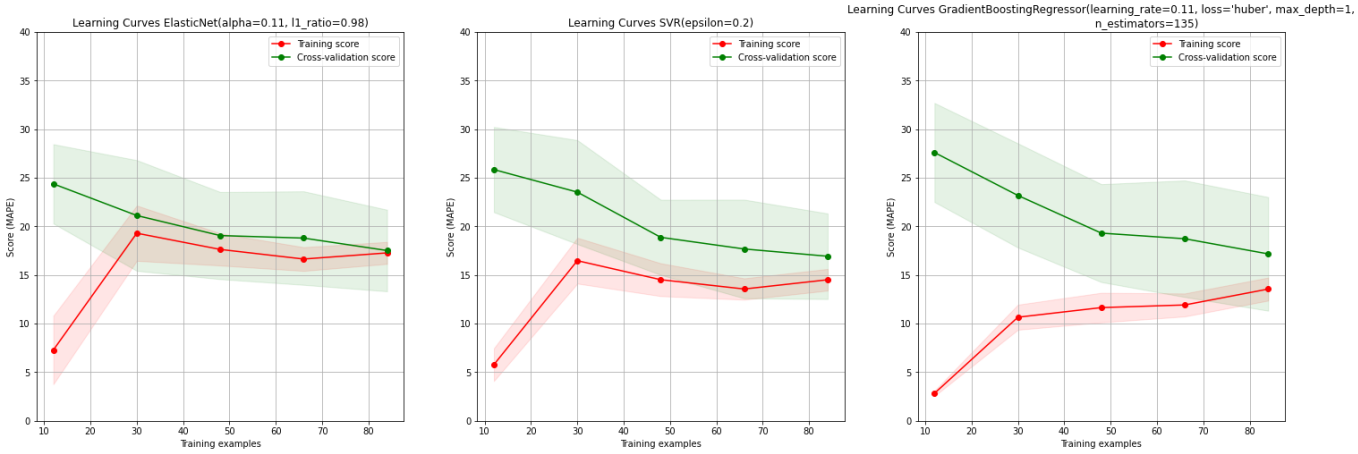

Figure 16. Cross-validation. Feature set #2

This time SVM did noticeably better and it appears to the best option for this feature set.

Judging by cross-validation results, the accuracy of predictions could still improve if they are given more training data. However, this should only apply to cases when one model is used for a long period of time without re-training it on recent data. For cases,

when the model is constantly given new data, lower training set sizes should be more desirable as REC patterns keep changing over time.

6.3 Analysis

The validation has shown that the REC predictive models have acceptable performance. Factors such as selected feature set, the number of days to predict for and the size of training set play a big role in the MAPE of the models.

Despite having an overall higher correlation with REC, some feature sets performed noticeably worse due to their irrelevancy in a short interval of time (e.g., feature sets that include *Season* feature). Although either of demonstrated feature sets is sufficient for predicting REC for a given dwelling, that could not be said about every feature set combination tested. Since the models rely on time lagged features and autoregression, the number of days for which REC predictions need to be made is a big deciding factor of models' accuracy. Even one low quality prediction in the start, is very likely to result in higher error of predictions for all the following days. As for training size, it also has a significant impact on quality of models. If size of a training set is too low (e.g., <14), then ML algorithms do not have enough data to learn from. Given the nature of this problem, very high number of observations used for training is not advisable as well. The testing has shown that for common use cases the optimal training set size lies somewhere in 28-56 range.

As far as ML algorithms are concerned, they had very similar performance. Elastic net having the lowest standard deviation of the accuracy of predicted REC values appears to be the most consistent option. On average predictions made by GB were less consistent but had the best MAPE score among other estimators. As for SVM, it proved to be the option of choice when using lower number of features.

Overall, the validation results can be deemed reasonable, but there is some room for improvement. The focus of this thesis was to explore only non-confidential external determinants of REC. However, these determinants do not have the best correlation with REC. One of the ways of improving the accuracy of predictions could be integrating some internal factors of REC such as working, studying and holidays schedule of the dwelling residents into existing models. Obtaining more general data about specific

dwelling such as presence of electrical heating, boiler or air conditioning units would also be beneficial for determining the most optimal feature set and hence improving the quality of REC predictions.

7 Summary

The goal of this thesis was to build REC predictive models using ML algorithms. In order to accomplish this, appropriate software was developed. To begin with, a research of external variables that might impact REC was conducted. Then these variables were gathered, pre-processed and analysed. Based on the analysis, they were combined into so called feature sets that were used for building and validating REC predictive models for the test dwelling. The results achieved by two of models built using the most optimal feature sets was demonstrated. Over the course of this thesis various combinations of features sets and ML algorithms were tested, and the quality of their predictions was evaluated.

The validation has shown that the quality of developed models depends on several factors. Choosing the right features, size of training set and ML algorithms is essential for building a high-quality REC predictive model. On average, the developed REC predictive models' accuracy using MAPE metric was noticed to be in the 10-25 range. Furthermore, the achieved results have shown that elastic net, SVM and GB ML algorithms can be deemed suitable for the problem of this thesis as they produced rather similar results.

To conclude, the goals of this thesis work have been successfully achieved. The result of this thesis is several developed REC predictive models for the test dwelling and software foundation that could be used for building predictive models for an arbitrary dwelling and further development and improving of existing models.

References

[1] M. Spichakova, J. Belikov, K. Nõu and E. Petlenkov, “Feature Engineering for Short-Term Forecast of Energy Consumption,” 2019.

[2] J. VanderPlas, “Why Python?,” in *Python Data Science Handbook: Essential Tools for Working with Data*, O'Reilly Media, Inc, 2016.

[3] “R: What is R?,” [Online]. Available: https://www.r-project.org/about.html. [Accessed 21 07 2020].

[4] “scikit-learn: Machine Learning in Python,” [Online]. Available: https://scikit-learn.org/stable/. [Accessed 21 07 2020].

[5] “Viru Keemia Grupp,” [Online]. Available: https://www.vkg.ee/en/. [Accessed 21 07 2020].

[6] “Sunset and sunrise times API,” [Online]. Available: https://sunrise-sunset.org/api. [Accessed 21 07 2020].

[7] “Calendarific Holiday API Documentation,” [Online]. Available: https://calendarific.com/api-documentation. [Accessed 21 07 2020].

[8] “Meteostat Developers - Historical Weather API,” [Online]. Available: meteostat.net/en. [Accessed 21 07 2020].

[9] V. Barnett and T. Lewis, in *Outliers in Statistical Data*, Wiley, 1987, p. 4.

[10] Statistics for High-Dimensional Data, Berlin: Springer, 2011.

[11] K. P. Bennett and C. Campbell, “Support Vector Machines: Hype or Hallelujah?,” Association for Computing Machinery, New York, 2000.

[12] Y. Zhang and A. Haghani, “A gradient boosting method to improve travel time prediction,” *Transportation Research Part C: Emerging Technologies*, 2015.