

Université de Lorraine
LUT University
Luleå University of Technology

Askar Serikov

ECOFEED: A BETTER ENERGY CONSUMPTION FEEDBACK SYSTEM

Examiners: Professor Eric Rondeau
Professor Jari Porras
Professor Karl Andersson

Supervisors: Professor Jari Porras
Associate Professor Pedro Juliano Nardelli

This thesis is prepared as part of an European Erasmus Mundus programme PERCCOM - PERvasive Computing & COMMunications for sustainable development.



Co-funded by the
Erasmus+ Programme
of the European Union



This thesis has been accepted by partner institutions of the consortium (cf. UDL-DAJ, n°1524, 2012 PERCCOM agreement). Successful defense of this thesis is obligatory for graduation with the following national diplomas:

- Master in Complex Systems Engineering (University of Lorraine)
- Master of Science in Technology (LUT University)
- Master in Pervasive Computing and Communications for Sustainable Development (Luleå University of Technology)

ABSTRACT

Université de Lorraine
LUT University
Luleå University of Technology

Askar Serikov

EcoFeed: a better energy consumption feedback system

Master's Thesis

74 pages, 37 figures, 5 tables

Examiners: Professor Eric Rondeau
Professor Jari Porras
Professor Karl Andersson

Keywords: Energy consumption, Energy conservation, Machine learning, Data visualization, Sustainability.

Remotely readable electricity meters have become commonplace. They generate a lot of data that is currently of little use to the consumers. At most, they have an opportunity to see their energy consumption dynamics over time as charts or graphs. This visualization is uninformative and does not reflect how everyday actions affect household's energy consumption. In this work, we propose a system that utilizes machine learning in order to create a better, near real-time visual feedback to the end users on their energy consumption. We call our solution EcoFeed. EcoFeed is aimed at providing the consumers with a better idea of their energy consumption and how their actions affect it. Studies have shown that when presented with better feedback, people tend to change their behaviour towards energy conservation and thus live more sustainable life. The main constraint we followed while developing EcoFeed was to make it easily implementable in real life. Hence, EcoFeed is developed using existing open-source technologies and utilizes only smart meters data and data from open sources. We have conducted a survey to evaluate how well EcoFeed communicates energy consumption to people and how it performs against the conventional visualization - a graph. Survey results show that EcoFeed is much better at communicating energy consumption to the end-users.

ACKNOWLEDGEMENTS

This work would not be possible without an enormous support and inspiration from the PERCCOM family: professors, staff, alumni and, of course, my classmates. I also want to thank my family and friends from my home country as well as the friends I've made during the programme.

Thank you <3

Rakhmet <3

Askar Serikov

June 2019

Lappeenranta, Finland

TABLE OF CONTENTS

ABSTRACT

ACKNOWLEDGMENTS

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1	INTRODUCTION	10
1.1	Background	10
1.2	Problem definition	11
1.3	Goals and Delimitations	11
1.4	Research methodology	12
1.5	Structure of the thesis	13
2	RELATED WORK	15
2.1	Machine learning and energy consumption	15
2.2	Data visualization and energy consumption	17
3	MACHINE LEARNING	19
3.1	Machine learning overview	19
3.2	Data description	20
3.3	Data preparation	23
3.4	Tools and Libraries	26
3.4.1	The Jupyter Notebook	27
3.4.2	Python	29
3.4.3	Scikit-learn	29
3.4.4	NumPy	30
3.4.5	pandas	31
3.4.6	TensorFlow	31
3.4.7	Matplotlib	32
3.5	Experiments	32
3.5.1	Coefficient of determination	32

3.5.2	Linear Regression	34
3.5.3	Support Vector Machines	35
3.5.4	Gradient Boosting	38
3.5.5	Long Short-Term Memory	39
3.6	Results and discussion	43
4	DATA VISUALIZATION	47
4.1	Visualization of energy consumption	47
4.2	Feedback visualization	50
4.3	EcoFeed: a better Energy consumption Feedback system	52
4.3.1	System Architecture	53
4.3.2	Developing a prototype	54
4.3.3	Creating a survey	56
4.4	Results and discussion	60
5	DISCUSSION, CONCLUSIONS AND FUTURE WORK	62
5.1	Discussion	62
5.2	Conclusions	63
5.3	Future Work	64
	REFERENCES	66

LIST OF FIGURES

1.1	Research methodology	13
3.1	Correlation between active energy consumption and outside temperature in the first 4000 records of Dataset 1	22
3.2	Correlation between active energy consumption and outside temperature in the first 4000 samples of Dataset 2	22
3.3	One-hot encoding example	24
3.4	Splitting <i>Date</i> feature set into several feature vectors	25
3.5	Expanding the initial dataset with <i>Light hour</i> feature vector	26
3.6	The Jupyter Notebook Interface	27
3.7	Google Colaboratory Interface	28
3.8	The Jupyter Notebook environment provided by CSC is no different from the regular Jupyter environment	29
3.9	Importing necessary scikit-learn modules in Jupyter Notebook	30
3.10	Using pandas to import a CSV file	31
3.11	Formula for calculating RMSE	33
3.12	Formula for calculating MAE	33
3.13	Formula for calculating R^2	33
3.14	The formula for multiple linear regression (Kenton, 2019)	34
3.15	Using scikit-learn Linear regression model	35
3.16	Linear SVM Model (Gupta and Rathee, 2016)	36
3.17	Using scikit-learn SVR model	38
3.18	Using scikit-learn Gradient Boosting Regressor model	39
3.19	Sigmoid function is the special case of the logistic function	40
3.20	The LSTM cell can process data sequentially and keep its hidden state through time	41
3.21	Reusable function for creating training data for LSTM	41
3.22	Using TensorFlow LSTM network	43
4.1	More informative energy bills (Wilhite and Ling, 1995)	47
4.2	Visualizing energy consumption per appliance provides better understand- ing of total energy consumption (Herrmann et al., 2018)	48
4.3	24 hours from the Dataset 1 where the actual consumption was signifi- cantly lower than expected	49
4.4	Colored circles convey a simple message: green - good, red - bad	51
4.5	EcoFeed can be easily integrated with modern smart IHDs	53

4.6	System architecture	53
4.7	Schematic depiction of actions performed every 2 seconds in the proto- type frontend	55
4.8	The circle changes color and time value in the center	55
4.9	Left: Google PowerMeter, right: our graph	56
4.10	Survey question with the graph	58
4.11	Survey question with the circle	59
4.12	Instruction for the questions with the circle	59
4.13	Survey results	61
5.1	Sustainability Analysis of EcoFeed	63

LIST OF TABLES

3.1	A dataset provided by the electricity providing company	21
3.2	Detailed description of the datasets	21
3.3	Performance evaluation results (R^2 scores) obtained during the experiments	44
3.4	Results of k -fold cross validation of the LSTM model using Dataset 1 . .	46
4.1	Demographics of the survey participants	60

1 INTRODUCTION

In this chapter, we discuss background of this work, we identify the research problem, set research goals and delimitations. This chapter also includes research methodology and describes the structure of the thesis.

1.1 Background

As per the revised in December 2018 European Union energy efficiency target, the goal is to achieve at least 32.5% reduction in energy consumption compared to previous projections by the year 2030 (EU, 2019). According to the latest reports, residential sector accounts for more than 25% of the total energy use in Europe (Eurostat, 2018). In order to increase energy savings on the household level, the EU's energy efficiency policy involves the planned rollout of close to 200 million smart meters for electricity and 45 million for gas by 2020. By the year 2017, in Finland 99% of energy consumption places were equipped with the remotely readable electricity meters (Energiavirasto, 2018). The smart meters are meant to provide better information to consumers protecting their rights to receive easy and free access to data on real-time and historical energy consumption. This measure is aimed at making Europeans use energy more efficiently, lower their bills and help protect the environment.

The latter is especially important since as of 2016 fuel combustion and fugitive emissions from fuels (without transport) constituted 52% of all greenhouse gas (GHG) emissions in Europe (EU, 2018). That fuel is used to generate energy and, as it was mentioned above, a quarter of it is used in residential sector. Therefore, improvements in energy savings on household level may lead to a significant reduction of GHG emissions.

Prior research conclude that better understanding of energy consumption leads to better energy conservation behaviour (Wilhite and Ling, 1995; Faruqui and Sergici, 2010; Faruqui et al., 2017). It means that smart meters data has a potential to decrease energy consumption in residential sector and, as a result, reduce GHG emissions. Householder awareness, in particular, has a saving potential of around 5-15% (Faruqui and Sergici, 2010), meaning that just by slightly changing their daily behaviors, home users can save

up to 15% of their current energy needs. We only need to unleash that potential by properly utilizing the data.

1.2 Problem definition

While the introduction of smart meters may indeed affect consumers' energy use, there is not much progress done on communicating the smart meters data to the end-users. Most companies these days offer an online platform where users can see their current energy use and access historical energy consumption data. This data is presented as a graph that shows how household's energy consumption changes over time. Studies have shown that such representation of energy consumption data is difficult for people to understand (Chisik, 2011; Herrmann et al., 2018). Moreover, this solution requires consumers to manually access the data which makes it impossible for users to easily see how their everyday actions affect energy use. One existing solution for that is installing an in-house display that shows the data in real-time. The main problem remains though, it is still difficult to clearly understand energy use visualized as a graph. A user has to study current energy consumption and compare it to preceding several days in order to understand if the energy use is higher or lower than normal. Hence, such feedback is less likely to make the user change their consumption behaviour.

We believe that with modern technologies and data analysis tools it is possible to create a better energy consumption feedback system. A system that will produce a visualization that is easy to understand and thus more likely to lead to a more efficient energy use.

1.3 Goals and Delimitations

The goal of this research is to propose a system that will utilize existing smart meters data to provide its users with a better feedback on their energy use. This way, our solution will cause more sustainable behaviour. In order to achieve that we need to:

- Investigate how machine learning (ML) is utilized in the domain of energy use. We are especially interested in applications where single house energy consumption is

predicted using ML.

- Build a predictive model using real smart meters data. This involves experimenting with the data and different machine learning algorithms.
- Use the predictive model to create a novel energy consumption feedback system. This involves finding an efficient visualization and developing a system prototype.
- Test the proposed system on real people to find out if it does really provide better understanding of energy consumption. This involves developing a survey and comparing the novel visualization to the conventional one - graph.

As it was mentioned already, in this work our goal is to make people understand their energy consumption better. We do not study how to persuade them to use less energy. We will discuss possible persuasive techniques in “Discussion” section. Previous studies allow us to assume that better understanding de facto leads to better energy conservation. In other words, the aim is to develop sustainable consumer behavior - a behavior that improves consumers’ social and environmental performance as well as meet their needs. It studies why and how consumers do or do not incorporate sustainability issues into their consumption behavior (Peattie and Belz, 2010).

One of our goals is to develop such a system that can be implemented relatively easy using currently available, open source technologies and data. The only data that is not publicly available is data from smart meters. However, if this system will be considered by electricity providing companies, it should be feasible to deploy it without modifying existing architecture.

1.4 Research methodology

During working on this thesis we followed the waterfall model (Benington, 1983). The reason is that since some stages of the research depend on outcomes of other stages, it was important to do them in sequence. When we prepared the data and built the predictive model, we carried out an experiment, thus we explained in detail how the data was obtained and prepared for the experiment as well as described each ML algorithm we have tried, what parameters were used, what results were obtained. During the prototype

development we explained the choice of the technologies used, how exactly certain parts of the prototype were developed and so on. Finally, at the survey stage we described how the survey questions were created, what data was used, what response type was used, what the results were. Figure 1.1 illustrates the research methodology used in this work.

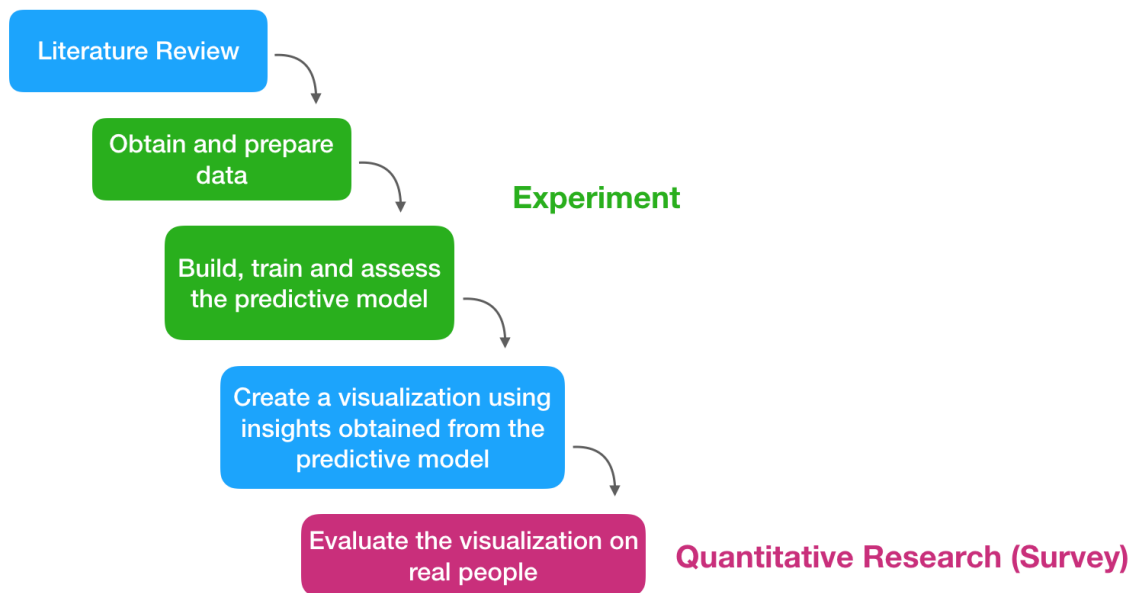


Figure 1.1: Research methodology

1.5 Structure of the thesis

This subsection provides an information about the thesis structure with a brief description of each chapter.

- **Introduction** provides an overview on the background of the research, what problems this research is trying to solve, what constraints this research has, what research methodology was followed.
- **Related work** describes some relevant studies that have been done in the research area and what results were obtained. This chapter is important as it explains why certain assumptions and decisions were made in this work .

- **Machine learning** describes everything that is related to machine learning in this research. It starts with data description, then how the data was prepared to be used for ML models training, what technologies, frameworks and libraries were used, what ML algorithms were tried, how they work, what results were obtained and some discussion in the end.
- **Data visualization** describes how the visualization was chosen for this work, how the visualization system prototype was built, what technologies were used, how it was evaluated using the survey, how the survey was developed, what survey results were and a some discussion in the end.
- **Discussion, conclusions and future work** contains discussions on the whole work, we talk about the outcomes of the thesis and what can be done in the future upon the results of the thesis.

2 RELATED WORK

In this chapter we briefly talk about relevant studies that we found during literature review. The papers described here have influenced the overall framework of this research. Some assumptions that we have made in this study as well as certain decisions were affected by the results of studies mentioned here. Since this work is split into two major parts: machine learning and data visualization, this chapter covers both of them separately.

2.1 Machine learning and energy consumption

Machine learning (ML) is the scientific study of algorithms and statistical models that computer systems use in order to perform a specific task effectively without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task (Koza et al., 1996). Machine learning is used in a wide variety of applications from email filtering to image recognition. Many studies have explored machine learning applications for modeling energy consumption, both in residential and commercial sectors.

In order to minimize the impact of individual user's actions on energy consumption, most studies in the field use aggregated data in either residential (Diao et al., 2017; Jain et al., 2014; Liu et al., 2017; Wijaya et al., 2015; Humeau et al., 2013) or commercial (Zhao and Magoulès, 2012; Grolinger et al., 2016; Abdelkader et al., 2016) buildings. It makes sense to work with aggregated data in order to create a more accurate predictive model since aggregated data has more regularities and less random in its nature. For instance, Jain et al. (2014) presented results of their predictive model based on data gathered from multi-family residential buildings, with single-families' consumption aggregated at building level. They built energy forecasting models using Support Vector Regressors (SVR) with various granularities of data. The results indicated that the most effective models were built with hourly consumption at floor level and had a standard error (standard deviation of the results obtained using the bootstrapping resampling method) of 28%. Liu et al. (2017) proposed a predictive system based on a sliding window, empirical mode

decomposition (SWEMD) and an Elman neural network (IENN) to predict the electricity use at a building level. Grolinger et al. (2016) in their study demonstrated that both neural networks (NN) and support vector machine (SVM) have a decent accuracy when forecasting energy use at event-organizing venues using energy consumption data and event characteristics. Zhang et al. (2016) investigated an institutional building's energy consumption using a weighted SVR, which was used to forecast half-hourly and daily electrical consumption. The work of Kaytez et al. (2015) evaluated different algorithms and concluded that the result of the proposed Least Squares Support Vector Machine (LS-SVM) model provided a quick prediction with higher accuracy than both traditional regression analysis and Artificial Neural Networks (ANN). (Li et al., 2016) use a different approach by applying language modelling to profile separate electrical appliances. The authors employ an aggregated dataset for sequences extraction of different appliances. In our case, sequences extraction of separate appliances is rather difficult as we operate with low-resolution datasets - households' total energy consumption.

Using aggregated data for single house prediction seems to work in some cases, for example when houses in the sample are similar in size and demography. However, if the households are different in certain characteristics, the interpolation does not provide good results. For this reason, we are more interested in building predictive models using single house data for training.

In the area of individual residential energy use prediction, several studies used highly detailed datasets for training ML models (Edwards et al., 2012; Dong et al., 2016). The datasets featured demographic information, household characteristics, ownership of certain appliances, and occupancy detection to predict household's electrical loads. These are typical bottom-up approaches (Grandjean et al., 2012). A research on residential buildings by Edwards et al. (2012) considered a specific research dataset: three residential units with 140 sensors collecting human actions such as using microwave oven, kettle, opening and closing refrigerators, as well as occupancy patterns. They presented results for commercial and residential consumption prediction and concluded that ANN-based algorithms perform best. The achieved prediction accuracy comes from decomposition of electrical usage behaviors by a large number of sensors in the experiments. It is virtually impossible to deploy such a system in real life in a cost-sensitive manner. In general, all cases presented in this paragraph are good for research purposes but are unimplementable in current conditions.

In this work, we aim to use only smart meters data and make forecasting on individual household level. Aforementioned studies used a wide range of ML algorithms and methods for energy use prediction. Mostly, ANN (Liu et al., 2017; Edwards et al., 2012), regression models (Gajowniczek and Zabkowski, 2017), SVR (Zhang et al., 2012; Li and Dong, 2017) and ensemble algorithms (Ben Taieb and Hyndman, 2014). We intend to try all 4 approaches in this research.

2.2 Data visualization and energy consumption

Data visualization is considered as a modern equivalent of visual communication. It involves the creation and study of the visual representation of data (Friendly, 2005). To communicate information clearly and efficiently, data visualization uses statistical graphics, plots, information graphics and other tools. Numerical data may be encoded using dots, lines, or bars, to visually communicate a quantitative message (Few, 2004). Effective visualization helps users analyze and reason about data and evidence. Data visualization makes complex data more accessible, understandable and usable.

Data visualization for better energy conservation is a part of the research area often referred to as “eco-feedback” (Froehlich et al., 2010). One of the first works on eco-feedback is dated 1995 and was conducted by Wilhite and Ling (1995). In their research they experimented with more informative energy bills that were paper based and were delivered to consumers at the end of the month. The three years long experiment showed that better feedback caused savings in energy consumption of up to 10%. Latter research pilots and surveys show (Andersen et al., 2009; He and Greenberg, 2009) that energy feedback is primarily a human-related task highly dependant on users and thus requiring user centered approaches for being tackled. Various kinds of feedbacks and visualizations may be employed and they can either induce changes into home inhabitants habits or be completely ignored depending on many factors including users’ green attitude, visual appearance, understandability of exposed data, etc. Among investigated mechanisms and visual solutions, the research community has currently reached a partial consensus on a set of basic interactions that are generally successful in promoting reductions in energy consumption. One of them is non-obtrusive displays, i.e., displays designed to weave themselves into the home environment, attracting the user attention when needed but avoiding intrusive settings and interactions that may foster interface abandoning or

disposal. Meanwhile, according to studies carried out by Wood and Newborough (2007), information on energy consumption must be presented in simple manner. They also conclude that it is difficult for householders to understand energy consumption presented in kWh. Herrmann et al. (2018) in their research show that aggregated energy consumption presented in graphs is uninformative to consumers and it takes time for them to study the graphs before they can get a useful information about their energy use.

From the aforementioned studies we can conclude that better feedback leads to better energy conservation. Users are generally willing to change their consumption behaviour when they understand when and how their behaviour is resulting in inefficient energy use. Another important finding here is that simpler energy feedback can be less efficient in terms of promoting energy conservation behaviour but performs well in terms of providing clear understanding of their current energy consumption. It is also worth noting that most of the studies have not used regular graphs for the feedback and one research showed that graphs are unclear to consumers.

3 MACHINE LEARNING

Before experimenting with different machine learning algorithms, it is important to have a general understanding of how machine learning works. The next section provides an overview of machine learning necessary for following this chapter. After, we will go through the datasets used in this research and how they were modified to achieve higher efficiency of machine learning algorithms.

3.1 Machine learning overview

Machine learning is a subfield of computer science that is concerned with building algorithms which, to be useful, rely on a collection of examples of some phenomenon. These examples can come from nature, be handcrafted by humans or generated by another algorithm. Machine learning can also be defined as the process of solving a practical problem by 1) gathering a dataset, and 2) algorithmically building a statistical model based on that dataset. That statistical model is assumed to be used somehow to solve the practical problem.

Learning can be supervised, semi-supervised, unsupervised and reinforcement (Burkov, 2019). In this research, the supervised learning was used.

In supervised learning, the dataset is the collection of labeled examples $\{(x_i, y_i)\}_{i=1}^N$. Each element x_i among N is called a feature vector.

A feature vector is a vector in which each dimension $j = 1, \dots, D$ contains a value that describes the example somehow. That value is called a feature and is denoted as $x^{(j)}$. For instance, if each example x in our collection represents a person, then the first feature, $x^{(1)}$, could contain height in cm, the second feature, $x^{(2)}$, could contain weight in kg, $x^{(3)}$ could contain gender, and so on. For all examples in the dataset, the feature at position in the feature vector always contains the same kind of information. It means that if $x_i^{(2)}$ contains weight in kg in some example x_i , then $x_k^{(2)}$ will also contain weight in kg in every example x_k , $k = 1, \dots, N$.

The label y_i can be either an element belonging to a finite set of classes $\{1, 2, \dots, C\}$, or a real number, or a more complex structure, like a vector, a matrix, a tree, or a graph. In

this research, we want the model to derive a real number - kWh value.

As it can be seen from the description above, in the datasets used in this research:

- kWh values are labels
- Temperature and Date are features

The goal of a supervised learning algorithm is to use the dataset to produce a model that takes a feature vector x as input and outputs information that allows deducing the label for this feature vector. For instance, the model created using the dataset of people could take as input a feature vector describing a person and output a probability that the person has cancer (Burkov, 2019). In our case, the goal is to derive an active energy consumption value at a given time using features such as temperature and time.

3.2 Data description

The datasets used in this research are energy consumption values obtained from smart meters installed by electricity providing companies in two different residential units in Lappeenranta, Finland. The electricity providing companies offer their customers a platform where they can check their bills and access historical energy consumption data with a one hour granularity. The datasets that were used in this research were downloaded from the platform and provided on a voluntarily basis by the owners of the residential units.

Dataset 1 belongs to a 320 m² two-storey house with 9 rooms, 2 garages and a sauna. A family of 3 adults is living in the house. The house has its own separate electric heating which greatly affects the energy consumption of the whole household.

Dataset 2 belongs to a 70 m² apartment with 2 bedrooms, a living room and a sauna with a family of 3 people living in it (2 adults and a 3 years old child). The building is connected to the central heating system of the city; therefore, the heating does not contribute to the apartment's energy consumption.

The structure of the datasets is shown in Table 3.1

Table 3.1: A dataset provided by the electricity providing company

Date	kWh	Temperature
01-01-2018 00:00	3.76	-1
...
30-12-2018 23:00	3.43	-2.40

The table above demonstrates how the energy consumption data is provided by the electricity providing companies. Both datasets used in this research for building predictive models consist of 8736 records each. The records in both datasets are observations taken within the same time period: from January 1st 2018 00:00 to December 30th 2018 23:00.

Detailed description of the datasets is provided in Table 3.2.

Table 3.2: Detailed description of the datasets

	Dataset 1		Dataset 2	
	kWh	Temperature	kWh	Temperature
# of records	8736	8736	8736	8736
Mean value	2.41	5.63	0.24	5.63
Standard deviation	1.49	10.95	0.30	10.95
Minimal value	0.37	-23.60	0.03	-23.60
Maximal value	11.05	31.80	5.34	31.80

Several important points that can be observed from the description:

- Since both residential units are located within the area of Lappeenranta city and the records refer to the same time period, temperature values are the same in both datasets.
- Energy consumption of the house (Dataset 1) is significantly higher than energy consumption of the apartment (Dataset 2): the mean value is 10 times bigger, 2.41 against 0.24. The reason is that the house is bigger in size, has more rooms and, this is very important, has internal heating, which greatly affects overall energy consumption of the house.

The fact that the electricity providing companies include the outside temperature in the historical data is indeed helpful while determining factors affecting energy consumption

of the residential units with internal heating as there is a strong correlation between active energy consumption and the temperature outside. Figure 3.1 demonstrates the correlation between the two in the Dataset 1.

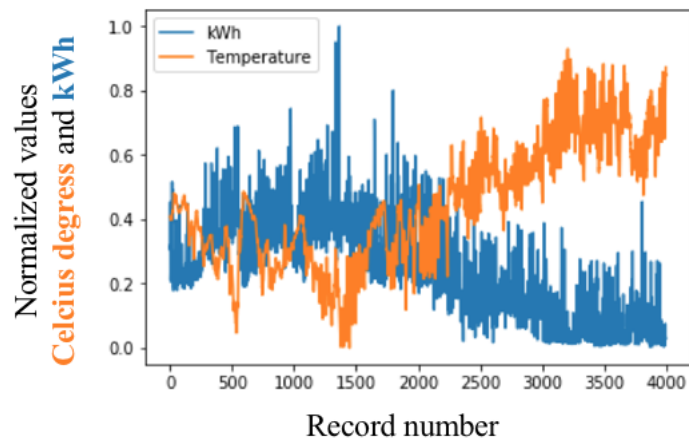


Figure 3.1: Correlation between active energy consumption and outside temperature in the first 4000 records of Dataset 1

Given the nature of many predictive modelling approaches such as linear regression and support vector machines (SVMs), it is beneficial for a predictive model to have features that correlate so well with the predicted, or target, value.

For the Dataset 2 however, this correlation is almost non-existent as can be seen on the Figure 3.2.

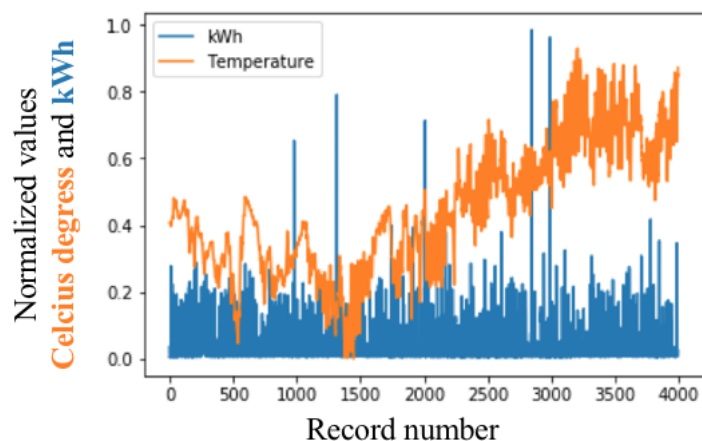


Figure 3.2: Correlation between active energy consumption and outside temperature in the first 4000 samples of Dataset 2

This can be explained by the absence of internal heating in the apartment. In this case, using temperature outside as a feature for building a predictive model may harm the accuracy of the model.

In the next section, we will prepare both datasets for building predictive models.

3.3 Data preparation

Before starting experimenting with different machine learning approaches, it is necessary to prepare the datasets. The process of data preparation for machine learning usually consists of several steps (Rençberoglu, 2019):

1. Checking for and handling missing values
2. Checking for and handling outliers
3. Binning
4. Scaling
5. Feature split
6. One-hot encoding

The initial datasets were free from missing values and, given the nature of the data, outliers (unusually high spikes of energy consumption) cannot be dropped out the datasets as they still carry an important information about energy consumption patterns. Since the goal of predictive models in this research is to derive exact numerical values (kWh), binning values is redundant. Scaling of the numerical values might improve performance of some machine learning algorithms and worsen others, thus the initial values will be left intact and be scaled where necessary afterwards. Feature split and one-hot encoding, however, can indeed improve our datasets.

As it was shown above, *Date* column has the following format: *Day-Month-Year Hour:Minute*. Obviously, this format implies that all values in the column are unique (e.g. *01-01-2018*

12:00 will only occur once in a dataset) which makes it a poor feature vector in the first place: no influence on the label can be derived from such vector.

One of the most efficient way of making date/time values consumable by machine learning algorithms is feature splitting by using a technique called one-hot encoding. One-hot encoding is one of the most common encoding methods in machine learning. This method spreads the values in a column to multiple feature vectors and assigns 0 or 1 to them.

User	City
1	Roma
2	Madrid
1	Madrid
3	Istanbul
2	Istanbul
1	Istanbul
1	Roma

→

User	Istanbul	Madrid
1	0	0
2	0	1
1	0	1
3	1	0
2	1	0
1	1	0
1	0	0

Figure 3.3: One-hot encoding example

It is important, however, to keep a reasonable number of feature vectors and disregard feature vectors that can worsen performance of the model.

Date column can be split into the following feature vectors:

- Month of the year [1, ..., 12]
- Days of the months [1, ..., 31]
- Days of the week [Monday, ..., Sunday]
- Hours of the day [0, ..., 23]

The months of the year indicate a season of the year, which can be an important factor affecting people's everyday routine, their energy consumption patterns and, as a result, energy consumption at the given time. However, the current month per se does not affect energy consumption, the temperature outside and the number of light hours a day do. Same applies to the days of the month, people's energy consumption behavior does not

depend on certain days in the month. There are “special” days such as public holidays but they can be marked separately, we will talk about it later.

The days of the week, on the other hand, can actually provide a better insight into energy consumption behavior of a household. There are many examples of activities people do at home weekly on certain days of the week: laundry on Fridays, cleaning the house on Saturdays and so on. That makes the days of the week a useful set of features. Moreover, we can add another feature that will indicate if a given day is a day off or not. It is obvious that people’s routine changes on the days off. This feature will cover not only weekends but also public holidays.

The hours of the day also highly correlate with the energy consumption behavior. People tend to use less electricity at night and more during peak hours such as lunch or dinner time. This makes them a useful set of features for some machine learning algorithms.

Applying one-hot encoding to *Date* column considering the aforementioned points expands the initial dataset into the following:

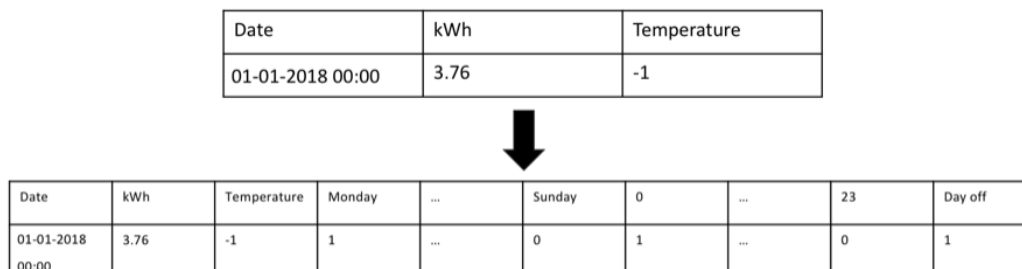


Figure 3.4: Splitting *Date* feature set into several feature vectors

Apart from modifying and adjusting the existing data, it is possible to expand the dataset by adding other relevant data from open sources. Weather condition can be such data. There is already outside temperature value set in the dataset but there are other parameters that affect energy consumption as well. One might consider wind as a provider of better insights into how cold or warm it actually is at a given time. While it might be true for colder seasons of the year, it does not make a difference during warmer seasons of the year. Light hours, however, may indicate when internal lighting turns on in the households. Since, according to (Stat.fi, 2018), artificial lighting constitute almost 3 per cent of

overall energy consumption in households as of 2017, it makes sense to add light hours as a feature vector to the dataset. To know if a given hour was a light hour, Python library called Astral (Kennedy, 2019) was used. The library uses geographical coordinates (longitude and latitude) to calculate sunrise and sunset times. Each hour in the dataset was compared with the times for the corresponding day, if the hour lied within an interval between sunrise and sunset it was marked as a light hour.

Date	kWh	Temperature	Monday	...	Sunday	0	...	23	Day off
01-01-2018 00:00	3.76	-1	1	...	0	1	...	0	1

↓

Date	kWh	Temperature	Monday	...	Sunday	0	...	23	Day off	Light hour
01-01-2018 00:00	3.76	-1	1	...	0	1	...	0	1	0

Figure 3.5: Expanding the initial dataset with *Light hour* feature vector

Unlike (Tso and Yau, 2007), where the authors assumed parameters such as ownership of certain appliances, size of residential units and the total income of a household in their dataset, in this research we have decided to use only the data that can be obtained from energy providing companies and open sources. The main reason is that the system proposed in this research must be easily implementable using existing tools and data sources. It is still possible to ask residents to manually enter that parameters, but it creates another set of possible complications: from privacy issues to the problems related to the reliability of manually entered data.

Having the dataset prepared, we can proceed to experimenting with different machine learning algorithms. In the next section we will overview the tools and frameworks used to carry out the experiments.

3.4 Tools and Libraries

In this work, several technologies, services, tools and libraries were employed. This section provides their detailed description.

3.4.1 The Jupyter Notebook

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more (Project Jupyter, 2019).

The Jupyter Notebook, along with other libraries used in this research, can be installed on Windows, macOS or Linux machines using as a part of Anaconda distribution (Anaconda Inc, 2019).

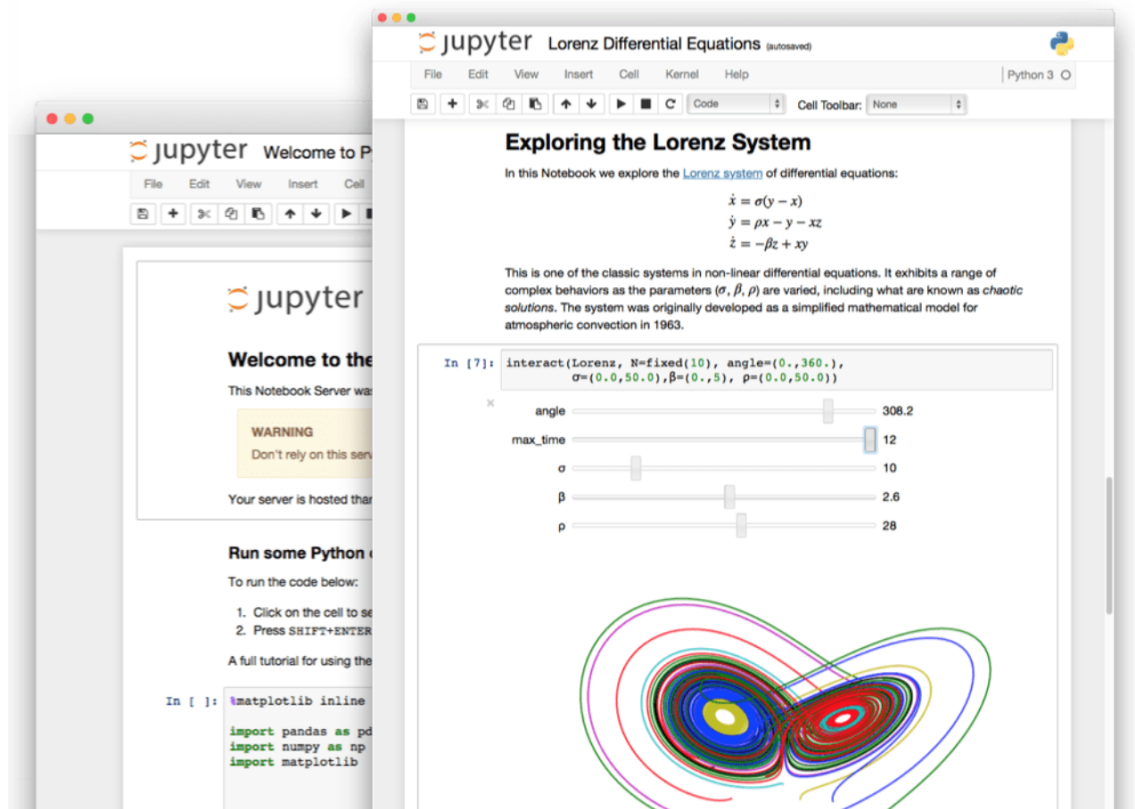


Figure 3.6: The Jupyter Notebook Interface

Since training machine learning models is a resource consuming task, a powerful station with a dedicated graphics card is usually preferred. Not having such a machine may result in slow model training process. One way to avoid this is to use cloud-based solutions for machine learning with The Jupyter Notebook.

Google provides a research tool for machine learning called Colaboratory (Google Inc., 2019a). There is no need to install anything on your local machine when using Google Colaboratory, necessary packages are also included. It is also possible to use GPU or TPU acceleration when running your code. One main disadvantage of the tool is that since it is free-to-use the GPU and TPU resources are not always available. It is also necessary to stay online when running experiments since quitting the environment for more than an hour will result in your session shutting down without a possibility to restore unsaved data such as variables stored in-memory. Another slight disadvantage of Google Colaboratory is that it uses a modified version of The Jupyter Notebook environment, which is not convenient when transferring notebooks from a standard Jupyter environment: slight modifications to the code might be required.

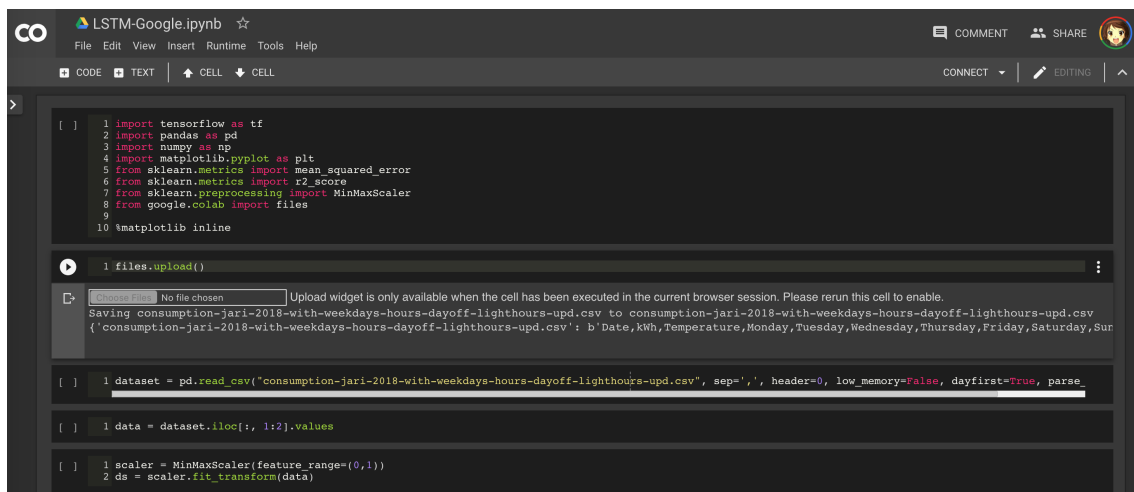


Figure 3.7: Google Colaboratory Interface

The solution we used in this research is a cloud-based Jupyter environment provided by CSC. CSC - IT Center for Science is a Finnish center of expertise in information technology owned by the Finnish state and higher education institutions (CSC, 2019). The company provides different research supporting services for Finnish students for free. One of them is a Jupyter environment for machine learning. The environment comes with all necessary packages pre-installed. From our experiments, it performs better than Google Colaboratory and provides a standard Jupyter environment, which makes it easy to transfer Jupyter notebooks between the cloud and local environments. Moreover, the environment is guaranteed to stay active for 10 hours, which gives us a possibility to run several time consuming experiments without the need of staying online.

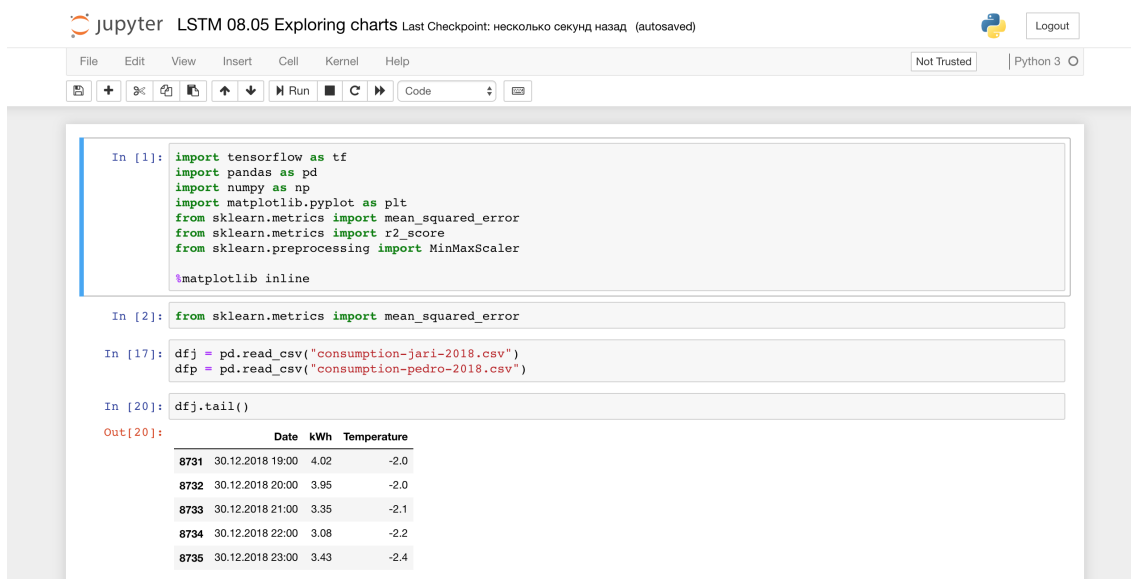


Figure 3.8: The Jupyter Notebook environment provided by CSC is no different from the regular Jupyter environment

3.4.2 Python

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aims to help programmers write clear, logical code for small and large-scale projects (Kuhlman, 2012).

Python was a programming language of choice for this research because it is considered a de facto standard language for machine learning applications due to the abundance of Python libraries made specifically for data analysis and machine learning. The Python version used in the experiments is 3.6.

3.4.3 Scikit-learn

Scikit-learn (formerly scikits.learn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting,

k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy (Pedregosa et al., 2011).

Scikit-learn modules used in this research include:

- Linear regression model
- Support Vector Machines model (SVM)
- Gradient Boosting ensemble model

Apart from machine learning models, scikit-learn comes with several metrics for performance evaluation of the models. In this research, coefficient of determination or R^2 is used for performance evaluation.

Figure 3.9 demonstrates how to import the library itself, the aforementioned models and the R^2 metrics in Jupyter Notebook:

```
import sklearn
from sklearn.neighbors import KNeighborsRegressor
from sklearn import linear_model
from sklearn.svm import SVR
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.metrics import r2_score
```

Figure 3.9: Importing necessary scikit-learn modules in Jupyter Notebook

3.4.4 NumPy

NumPy is the fundamental package for scientific computing with Python (NumPy Devs., 2019). NumPy enables powerful N-dimensional array objects, sophisticated (broadcasting) functions, tools for integrating C/C++ and Fortran code, useful linear algebra, Fourier transform, and random number capabilities. Our use of NumPy was limited to its array capabilities.

3.4.5 pandas

pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language (Mckinney and Pydata Development Team, 2019). We used pandas to import the datasets and manipulate them during the experiments.

The datasets used in this research are CSV formatted. pandas make it easy to import csv files and manipulate them by converting them into pandas's *DataFrame* two-dimensional size-mutable data structure. Figure 3.10 demonstrates how pandas is used to import csv files.

```
In [2]: import pandas as pd
dataset_1 = pd.read_csv("dataset-1.csv")
dataset_1.head()
```

```
Out[2]:
```

	Date	kWh	Temperature	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday	...	16	17	18	19	20	21	22	23	Date
0	1/1/18 0:00	3.76	-1.0	True	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	Fi
1	1/1/18 1:00	3.62	-1.0	True	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	Fi
2	1/1/18 2:00	5.21	-1.0	True	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	Fi
3	1/1/18 3:00	3.56	-1.2	True	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	Fi
4	1/1/18 4:00	2.61	-1.3	True	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	Fi

5 rows x 36 columns

Figure 3.10: Using pandas to import a CSV file

3.4.6 TensorFlow

TensorFlow is an open source software library for numerical computation using data flow graphs. The graph nodes represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) that flow between them. This flexible architecture enables one to deploy computation to one or more CPUs or GPUs in a desktop, server, or mobile device without rewriting code (Google Inc., 2019b).

TensorFlow was used in this research for its deep neural networks capability. In this research, we used Long Short-Term Memory (LSTM) type of recurrent neural networks.

3.4.7 Matplotlib

Matplotlib is a 2D graphics package used for Python for application development, interactive scripting, and publication-quality image generation across user interfaces and operating systems (Hunter, 2007). The package was used to plot graphs for visual assessment and demonstration purposes. Most of the graphs presented in this thesis were generated using Matplotlib.

3.5 Experiments

In this section we describe which machine learning algorithms we experimented with, what parameters were used in the software environment. We also explain the performance evaluation method employed in the experiments. The experiments results are described and discussed in the next section.

3.5.1 Coefficient of determination

Before we start experimenting with different machine learning algorithms, it is important to pick the right performance metrics. In this research, we use Coefficient of determination denoted R^2 to evaluate how accurate the models predict continues values.

There are three main metrics for evaluating regression models:

- Root Mean Square Error (RMSE)
- Mean Absolute Error (MAE)
- R Squared (R^2)

The reason we have not chosen RMSE or MAE is that both of them depend on the distance (difference) between predicted and actual values.

This approach imposes a problem when we want to compare the performance of the same model against our datasets. Dataset 1 labels, as it was mentioned before, are generally

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{j=1}^n (y_j - \hat{y}_j)^2}$$

y_j – predicted value
 \hat{y}_j – actual value
 n – number of predictions

Figure 3.11: Formula for calculating RMSE

$$\text{MAE} = \frac{1}{n} \sum_{j=1}^n |y_j - \hat{y}_j|$$

y_j – predicted value
 \hat{y}_j – actual value
 n – number of predictions

Figure 3.12: Formula for calculating MAE

bigger than the ones in Dataset 2. Mean of all *kWh* values in Dataset 1 is 2.41 while in Dataset 2 it is 0.24. Naturally, the MAE of 0.15, for example, is a decent result for Dataset 1 but a poor result for Dataset 2. Therefore, using MAE or RMSE is not convenient for evaluation of the model's performance against two different datasets.

R^2 , on the other hand, shows how well the model's prediction set fits to the actual values. R^2 score lies between 0 and 1 for no-fit and perfect fit respectively. This metric enables an easy comparison of the model's performance against different datasets.

$$\hat{R}^2 = 1 - \frac{\sum_{i=1}^n (Y_i - \hat{Y}_i)^2}{\sum_{i=1}^n (Y_i - \bar{Y})^2} = 1 - \frac{\frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2}{\frac{1}{n} \sum_{i=1}^n (Y_i - \bar{Y})^2}$$

Y_j – predicted value
 \hat{Y}_j – actual value
 n – number of predictions

Figure 3.13: Formula for calculating R^2

3.5.2 Linear Regression

In statistics, linear regression is a linear approach to modeling the relationship between a scalar response (or dependent variable) and one or more explanatory variables (or independent variables). The case of one explanatory variable is called simple linear regression. For more than one explanatory variable, the process is called multiple linear regression (Freedman, 2009). Since in our case we have several feature vectors, multiple linear regression is used.

Multiple linear regression (MLR), also referred to as multiple regression, is a statistical technique that uses several explanatory variables (features) to predict the outcome of a response variable (label). The goal of MLR is to model the linear relationship between the features variables and the label. Basically, multiple linear regression is the extension of ordinary least-squares (OLS) regression that involves more than one feature vector.

Figure 3.14 shows the formula for MLR.

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip} + \epsilon$$

where, for $i = n$ observations:

y_i = dependent variable

x_i = explanatory variables

β_0 = y-intercept (constant term)

β_p = slope coefficients for each explanatory variable

ϵ = the model's error term (also known as the residuals)

Figure 3.14: The formula for multiple linear regression (Kenton, 2019)

Scikit-learn provides a Linear Regression model which accepts several parameters that were left intact in this experiment. Linear Regression models require training in advance to derive coefficients for each feature vector. The datasets were split into training and test sets with 80% to 20% ratio.

All feature vectors of the initial datasets were used for model training. The aforementioned dataset split ratio was selected experimentally. For Dataset 2, *Temperature* feature vector was not used as there is no correlation between the temperature outside and the energy consumption in the apartment with no internal heating. Figure 3.15 demonstrates how the Linear regression model is used in Jupyter environment.

```

from sklearn import linear_model
import pandas as pd

mlr = linear_model.LinearRegression()

dp = pd.read_csv("Dataset-1.csv")

features = dp.drop(columns=['kWh'])
targets = dp['kWh'].values

train_features = features[:int(len(features)*0.8)]
train_targets = targets[:int(len(targets)*0.8)]

test_features = features[int(len(features)*0.8):]
test_targets = targets[int(len(targets)*0.8):]

mlr.fit(train_features, train_targets)
predicted = mlr.predict(train_features)

```

Figure 3.15: Using scikit-learn Linear regression model

3.5.3 Support Vector Machines

Support Vector Machine (SVM) (Shoosmith et al., 2006) is a supervised machine learning algorithm which can be used for both classification or regression challenges (Gunn, 1998). Even though it is mostly used in classification problems, SVMs are often used for regression of continuous values as well. Support Vector Machines used for regression are often referred to as Support Vector Regression (SVR).

In SVM, each data item is plotted as a point in n -dimensional space (where n is number of feature vectors) with the value of each feature being the value of a particular coordinate. The goal of the SVM algorithm is to find such a line, hyper plane, that will split different classes from each other with the maximum margin.

SVMs achieve non-linearity by employing a non-linear mapping function $K(x,x)$ that transforms the input x into an N dimensional non-linear output. It allows us to construct a linear model in this new feature space. The linear model in this feature space is given by:

$$f(\mathbf{x}, \mathbf{w}) = \sum_{n=1}^N w_n \cdot K(\mathbf{x}, \mathbf{x}_n) + w_0$$

In this equation, w_0 stands for the bias term and $K(x, x_n)$, $n=1,2,\dots,N$ represents a set of non-linear transformations. SVR uses a loss function $L(t, f(x, w))$ to evaluate the quality of the estimation. The loss function is called ε -insensitive loss function:

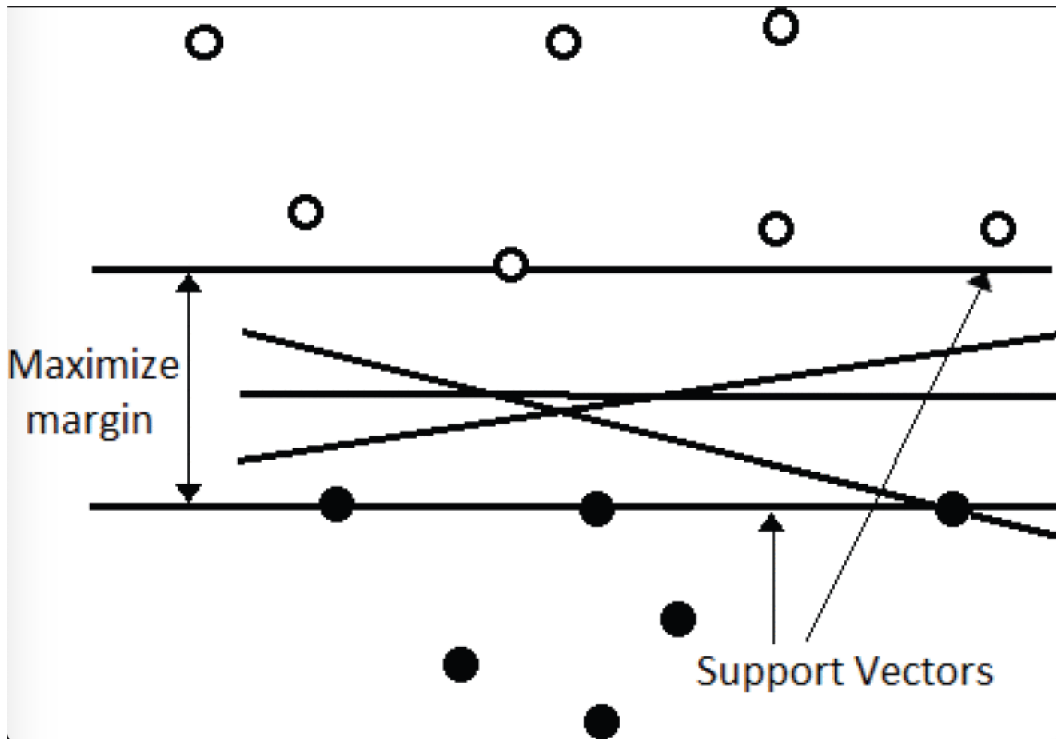


Figure 3.16: Linear SVM Model (Gupta and Rathee, 2016)

$$L(\mathbf{t}, f(\mathbf{x}, \mathbf{w})) = \begin{cases} 0 & , |\mathbf{t} - f(\mathbf{x}, \mathbf{w})| - \varepsilon \leq 0 \\ |\mathbf{t} - f(\mathbf{x}, \mathbf{w})| - \varepsilon, & otherwise \end{cases}$$

SVM employs ε -insensitive loss for linear regression in the high-dimensional feature space while reducing model complexity by minimizing $\|\mathbf{w}\|^2$. This is done by introducing slack variables $\xi_n, \xi_n^*, n = 1, 2, \dots, N$ to measure the deviation of training samples outside the ε -sensitive zone. So SVR is formulated as minimization of the following function:

$$\min \frac{1}{2} \mathbf{w}^2 + C \sum_{n=1}^N (\xi_n + \xi_n^*) \quad (1)$$

The following conditions apply:

$$\begin{cases} t_n - f(x_n, \mathbf{w}) \leq \varepsilon + \xi_n^* \\ f(x_n, \mathbf{w}) - t_n \leq \varepsilon + \xi_n \\ \xi_n, \xi_n^* \geq 0, n = 1, \dots, N \end{cases}$$

The constant $C > 0$ determines the trade off between the flatness of f and the values up to which deviations greater than ε are tolerated. The solution to (1) is obtained by transforming it into a dual optimization problem. Finally the regression function is stated as:

$$f(\mathbf{x}) = \sum_{n=1}^N (\alpha_n^* - \alpha_n) K(\mathbf{x}, x_n)$$

Where the kernel function

$$K(\mathbf{x}, x_n) = \sum_{i=1}^N g_i(\mathbf{x}) g_i(x_n)$$

does the non-linear mapping of the linear input space to nonlinear output space.

Scikit-learn provides an Epsilon-Support Vector Regression model that accepts the following parameters:

- **kernel**: Specifies the kernel type to be used in the algorithm. Set to 'rbf'.
- **C**: Penalty parameter of the error term. Set to 100.

The model accepts other parameters as well but they were set to default values in this experiment. For this experiment, we split the datasets into training and test sets in the ratio of 80% to 20%. It was also experimentally deducted that days of the week worsen the performance of the model, thus feature vectors $[Monday, \dots, Sunday]$ were not used in this experiment. The aforementioned parameters and the dataset split ratio were selected experimentally as well.

Figure 3.17 demonstrates how the SVR model is used in Jupyter environment.

```

from sklearn.svm import SVR
import pandas as pd

ds = pd.read_csv("Dataset-1.csv")

train = ds.values[:int(len(ds) * 0.8), :]
test = ds.values[int(len(ds) * 0.8):, :]

train = np.delete(train, np.s_[3:9], axis=1)
test = np.delete(test, np.s_[3:9], axis=1)

train_targets = train[:, 0:1]
test_targets = test[:, 0:1]

train_features = train[:, 1:]
test_features = test[:, 1:]

svr = SVR(kernel='rbf', C=100)

model = svr.fit(train_features, train_targets)
predicted = model.predict(test_features)

```

Figure 3.17: Using scikit-learn SVR model

3.5.4 Gradient Boosting

Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. (Wikipedia, 2019).

The name Gradient Boosting comes from combining Gradient Decent and Boosting algorithms. A simple example of a gradient decent algorithm would be ordinary least squares that is used in Linear Regression. Boosting is a technique of combining multiple weak learning algorithms into one more powerful algorithm. The combining process is basically adding new models to the ensemble sequentially (hence the name, ensemble algorithms). At every iteration, a new weak learner is trained upon the output of the previous weak learners. Gradient boosting builds the model in a stage-wise fashion, as other boosting methods do, and generalizes them by allowing optimization of an arbitrary differentiable loss function (Scikit-learn, 2019).

Scikit-learn provides a Gradient boosting regressor model that accepts the following parameters:

- **n_estimators**: The number of boosting stages to perform. Gradient boosting is fairly robust to over-fitting so a large number usually results in better performance. Set to 1000.
- **max_depth**: Maximum depth of the individual regression estimators. It limits the number of nodes in the tree. Set to 10.

- **random_state**: Seed used by the random number generator. Set to 13.

The model accepts other parameters as well but they were set to default values in this experiment. For this experiment, we split the datasets into training and test sets in the ratio of 80% to 20%. For Dataset 2, *Temperature* feature vector was not used as there is no correlation between the temperature outside and the energy consumption in the apartment with no internal heating. Figure 3.18 demonstrates how Gradient Boosting Regressor is used in Jupyter environment:

```
import pandas as pd
from sklearn.ensemble import GradientBoostingRegressor
import numpy as np

dp = pd.read_csv("Dataset-1.csv")

dp.drop(['Date'], axis=1, inplace=True)
X = dp.drop(columns=['kWh'])
Y = dp['kWh'].values

features_train = X[:int(len(X)*0.8)]
features_test = X[int(len(X)*0.8):]

targets_train = Y[:int(len(X)*0.8)]
targets_test = Y[int(len(X)*0.8):]

gbr = GradientBoostingRegressor(max_depth=10, random_state=13, n_estimators=1000)

gbr.fit(features_train, targets_train)
predicted = gbr.predict(features_test)
```

Figure 3.18: Using scikit-learn Gradient Boosting Regressor model

3.5.5 Long Short-Term Memory

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture (Hochreiter and Schmidhuber, 1997) used in the field of deep learning. Unlike regular feedforward neural networks (i.e. networks where nodes connections do not form a cycle), LSTM has feedback connections that make it a “general purpose computer” meaning that it can compute anything that a Turing machine can (Siegelmann and Sontag, 1995).

LSTM is extremely powerful for processing entire sequences of data. Its modern applications include handwriting (Graves et al., 2009) and speech (Sak et al., 2014) recognition. LSTM is even considered “arguably the most commercial AI achievement, used for everything from predicting diseases to composing music” (Vance, 2018). Indeed, many

companies use LSTM networks in their products: Google uses LSTM for speech recognition in smartphones (Sak et al., 2015) and for Google Translate (Wu et al., 2016); Apple uses LSTM for its "Quicktype" feature and for Siri (Smith, 2016); Amazon uses LSTM for Alexa (Werner, 2016).

What makes LSTM interesting for our research is that it is well-suited to classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series. LSTMs were also chosen over regular RNNs for this experiment because of its ability to deal with the exploding and vanishing gradient problems that can be encountered when training regular Recurrent Neural Networks.

A common LSTM unit consists of a cell, an input gate, an output gate and a forget gate. The cell remembers values over set time intervals and the three gates regulate the flow of information into and out of the cell.

The cell keeps track of the dependencies between the elements in the input sequence. The input gate controls how a new value flows into the cell, the forget gate controls how a value remains in the cell and the output gate controls how the value in the cell computes the output activation of the LSTM unit. The activation function of the LSTM network we use in this research is hyperbolic tangent ($\tanh(\alpha)$) and the recurrent activation function is hard sigmoid which is a segment-wise linear approximation of the regular sigmoid.

$$S(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{e^x + 1}.$$

Figure 3.19: Sigmoid function is the special case of the logistic function

There are connections into and out of the LSTM gates, a few of which are recurrent. The weights of these connections, which need to be learned during training, determine how the gates operate.

LSTM used in this experiment is provided by TensorFlow library as a part of its implementation of Keras API (Keras Team, 2019). Because of its nature, the usage of LSTM is significantly different from the other ML algorithms.

First of all, it was experimentally established that the LSTM performs best when we use

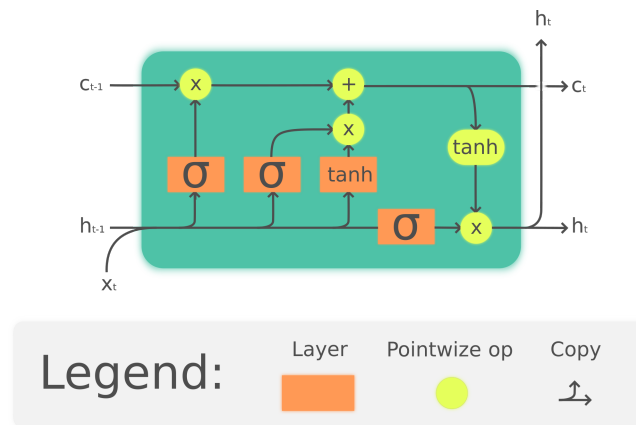


Figure 3.20: The LSTM cell can process data sequentially and keep its hidden state through time

a label vector $[y_0, \dots, y_{i-1}]$ to predict label y_i . Thus, LSTM does not rely on the dependencies between the features and the label, instead, it is trying to learn on the label vector itself. This approach requires additional modification of the dataset: for each label y_i we have to create a vector $[y_0, \dots, y_{i-1}]$ that will be used for training. We will refer to the number i as *Window size*.

To make the vectors creation process simpler, we have developed a separate function `create_datasets()` that takes a dataset and a *Window size* value as parameters and returns a new two dimensional array (a vector of vectors $[y_{i-Window\ size}, \dots, y_i]$ where $i = Window\ size, \dots, n$) and a label vector.

```
def create_datasets(data, window_size):
    x, y = [], []
    for i in range(window_size, len(data)):
        a = data[(i-window_size):i, 0]
        x.append(a)
        y.append(data[i,0])
    return np.array(x), np.array(y)
```

Figure 3.21: Reusable function for creating training data for LSTM

Secondly, it is generally recommended (Zhang et al., 2012) to scale data before using it as an input for an RNN. It is a trivial task when using scikit-learn as it features a module called *MinMaxScaler* that allows us to scale any array within any range. We scale our input array in the range between 0 and 1.// Also, unlike the previous ML algorithms, with LSTM we split our dataset into training and test sets in 60% to 40% ratio respectively.

A big limitation of LSTM in this experiment is that it performs the best when predicting

only a few steps ahead. Naturally, the highest prediction accuracy is observed when predicting only one step ahead, i.e. using $n-1$ values to predict the value n . Even though it may be considered unacceptable for other problems, it is still a valid solution for this research. The reason will be explained in the next chapter.

As it was explained above, LSTMs consist of LSTM units, also called *nodes*. Same as with regular RNNs, the nodes are combined in layers. It is an experimenter's task to pick the right number of nodes and layers. Obviously, higher the number of nodes higher the computational cost.

TensorFlow provides an “empty” model as an instance of **Sequential** class, to which additional layers can be added via **add()** method. The method accepts an instance of **Layer** Class. In TensorFlow, LSTM is a sub-class that is inherited from **RNN** class which in its turn is inherited from the base **Layer** class. Therefore, we pass an instance of the LSTM class to the **add()** method to add an LSTM layer to our model. In this research we use 3 LSTM layers, 100 hidden units each. In TensorFlow, an LSTM layer instance accepts the following parameters:

- **units**: Positive integer. Dimensionality of the output space. Set to 100.
- **return_sequences**: Boolean. Whether to return the last output in the output sequence, or the full sequence. Set to **true**.

It accepts other parameters as well but they were left default in this experiment.

After 3 LSTM layers we add an output layer that is an instance of **Dense** class. We can set how many output values we want. In our case, it is set to 1.

The model has to be configured before training. This is done with **compile()** method. Here we pass two arguments:

- **loss**: loss function to be used during training. Set to "mean_square_error".
- **optimizer**: optimizer to be used during training. Set to "adam".

Upon training the model we also specify the number of epochs i.e. number training iterations. The number of epochs impacts highly the computation time. It was experimentally

established that after 20 iterations the accuracy of the model does not improve enough to justify higher computation cost. Therefore, the number of epochs used in this research is set to 20. Figure 3.22 demonstrates how LSTM networks are used in Jupyter environment.

```
import tensorflow as tf
import pandas as pd
import numpy as np
from sklearn.preprocessing import MinMaxScaler

dataset = pd.read_csv("Dataset-1.csv")
data = dataset.iloc[:, 1:2].values
scaler = MinMaxScaler(feature_range=(0,1))
ds = scaler.fit_transform(data)
ws = 72

def create_datasets(data, window_size):
    x, y = [], []
    for i in range(window_size, len(data)):
        a = data[(i-window_size):i, 0]
        x.append(a)
        y.append(data[i,0])
    return np.array(x), np.array(y)

def fit_model(train_X, train_Y, window_size = 1):
    model = tf.keras.models.Sequential()
    model.add(tf.keras.layers.LSTM(100, input_shape = (1, window_size), return_sequences=True))
    model.add(tf.keras.layers.LSTM(100, return_sequences=True))
    model.add(tf.keras.layers.LSTM(100))
    model.add(tf.keras.layers.Dense(1))
    model.compile(loss = "mean_squared_error",
                  optimizer = "adam")
    model.fit(train_X,
              train_Y,
              epochs = 10,
              batch_size = 1,
              verbose = 2)

    return(model)

train = dataset[(len(dataset)*0.6), :]
test = dataset[(len(dataset)*0.6), :]
train_x, train_y = create_datasets(train, ws)
test_x, test_y = create_datasets(test, ws)
train_x = np.reshape(train_x, (train_x.shape[0], 1, train_x.shape[1]))
test_x = np.reshape(test_x, (test_x.shape[0], 1, test_x.shape[1]))
model = fit_model(train_x, train_y, ws)
predicted = model.predict(test_x)
```

Figure 3.22: Using TensorFlow LSTM network

3.6 Results and discussion

Table 3.3 demonstrates the results that were obtained during experimenting with the ML algorithms in the settings described in the previous subsection.

It can be seen that Dataset 2 consistently produces less accurate models than Dataset 1. One reason for this difference is the presence of internal heating in the house that Dataset 1 belongs to. From Figure 3.1 that shows high correlation between energy consumption

Table 3.3: Performance evaluation results (R^2 scores) obtained during the experiments

ML Algorithm	Dataset 1	Dataset 2
Linear Regression	0.50	0.16
SVM	0.41	0.07
Gradient Boosting	0.31	0.16
LSTM	0.74	0.21

of the house with outside temperature, we can conclude that internal heating is a large contributor to overall energy consumption. The correlation itself is an important factor in ML models training. Feature vectors that correlate well with the label greatly improve a model’s accuracy. As a result of this correlation, models trained on Dataset 1 are more accurate.

It does not, however, explain why the LSTM model trained on Dataset 1 performs better as well. As it was described in the previous subsection, we did not use other feature vectors apart from the label vector itself for training LSTM models. In this case, correlations between the features and the label do not affect the model’s accuracy. LSTM learns from trends in the label vector itself. Obviously, stronger the trends in the dataset - better, more accurate the resulting model. In other words, predicting data that changes randomly is difficult, if not impossible. The question is, what makes Dataset 2 more “random”? Again, it can be partially explained by the absence of internal heating. That makes residents’ actions the main and only cause of changes in household’s energy consumption. Basically, for Dataset 2, the problem shifts from predicting energy consumption to predicting human behaviour.

Human behaviour is chaotic and thus highly unpredictable. Predicting human behaviour is an important problem in many research areas: from marketing, where predicting human purchase behaviour (Valecha et al., 2018) has been one of the main challenges in the last decades especially with the relatively recent rise of online shopping, to education (Chen et al., 2019), transportation (Leonhardt and Wanielik, 2018) and healthcare (Amimeur et al., 2018). In order to solve the problem of human behaviour’s chaotic nature, the researchers used data that explained human’s activity in corresponding context with very high detail. For example, Chen et al. (2019) captured all actions that users performed on an online education platform to build a prediction model. Calvert and Brammer (2012) on the other hand, used a completely different method to predict human behaviour: they took a “mind-reading” approach by applying machine learning to functional magnetic

resonance imaging (fMRI) data. Obviously, the datasets used in this research are not detailed enough to achieve the same level of insights.

While in the context of energy consumption there were advances in predicting energy consumption behaviour on an individual or household level using sub-metering systems (Rajasekaran et al., 2017), the prediction accuracy achievable under the constraints of this research does not satisfy the required level of accuracy for using it for the system we are building. Thus, it was decided not to proceed with Dataset 2 further in this research.

Dataset 1, unlike Dataset 2, is not completely dependant on residents' energy consumption behaviour. As we know, Dataset 1 belongs to a large house that features internal heating, 2 garages and is generally larger. For that reason, the house's baseline energy consumption is higher and random human actions affect less the overall energy consumption. That makes the dataset less chaotic and better fit for predictive modeling. With that provided, we observe a paradoxical situation: the goal of the research is to provide residents with a better understanding of their energy consumption with the aim of changing their energy consumption behaviour towards better conservation however, under the limitations of this work, it is only feasible to build a reliable predictive model in an environment where human behaviour is not the key influencer of the overall energy consumption. Nevertheless, the LSTM model built using Dataset 1 provides predictive accuracy high enough to reflect changes in energy consumption caused by residents' actions. As a result, LSTM model was considered for further validation to evaluate the consistency of its prediction accuracy.

Firstly, the LSTM model was tested using k -fold cross validation method (Stone, 1974). In k -fold cross-validation, the original dataset is partitioned into k chunks of equal size. Out of the k chunks, one is used as a test data while other $k-1$ chunks are used as training data. The cross-validation process is repeated k times, with each of the k chunks used as the test set. The k results are then averaged to produce the final estimation. We split our dataset into 5 chunks for the cross validation, i.e. $k=5$. Table 3.4 shows the results of each iteration and the final estimation.

The model maintains its performance with small deviations across the iterations. Another way of validating the model would be to test it against different datasets. Even though there were no datasets from other houses available when the experiments were carried

Table 3.4: Results of k -fold cross validation of the LSTM model using Dataset 1

Iteration #	R^2 score
1	0.71
2	0.68
3	0.78
4	0.72
5	0.74
Final Estimation	0.73

out, we had datasets containing consumption data of the same house from the years of 2015 and 2017 at our disposal. We tested the model with the two datasets splitting them in the same ratio as the initial one: 60% for training and 40% for validation. The results, R^2 scores, achieved during the experiments were **0.68** and **0.71** for 2015 and 2017 respectively.

LSTM model trained on Dataset 1 consistently produces predictions of decent accuracy with an average R^2 score of 0.71 across validation sets. Dataset 1 and the predictions made by the LSTM model were used for the next stage of this research - data visualization.

4 DATA VISUALIZATION

Data visualization is considered as a modern equivalent of visual communication. It involves the creation and study of the visual representation of data (Friendly, 2005). To communicate information clearly and efficiently, data visualization uses statistical graphics, plots, information graphics and other tools. Numerical data may be encoded using dots, lines, or bars, to visually communicate a quantitative message (Few, 2004). Effective visualization helps users analyze and reason about data and evidence. Data visualization makes complex data more accessible, understandable and usable.

4.1 Visualization of energy consumption

First attempts at improving the communication of household energy consumption to its residents started in 1995 when Wilhite and Ling (1995) carried out a set of experiments by providing more informative monthly energy bills to some households in Oslo, Norway. The research showed that the energy consumption in the households that were receiving experimental bills has reduced by about 10%. Moreover, questionnaire and interview data showed that the residents of the households paid more attention to the bills, were more likely to discuss bills with other members of the household, and were positive to continuing with the experimental billing system. The authors have also estimated that the cost of introducing more informative bills is minimal in comparison to savings.

The next breakthrough in visualization of energy consumption has become possible with

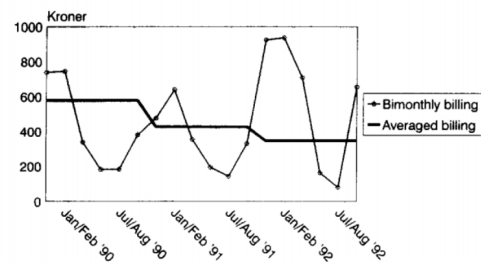
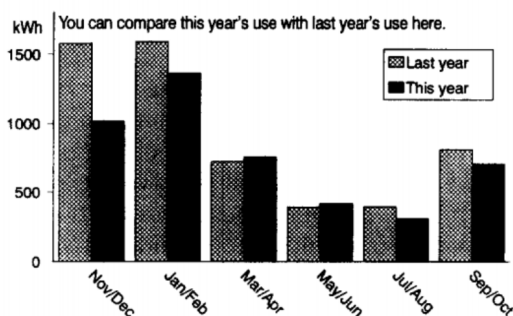


Fig. 4. The household's view of seasonal variation in their electrical consumption.

Figure 4.1: More informative energy bills (Wilhite and Ling, 1995)

the introduction of smart meters that remotely communicated active energy consump-

tion with certain intervals. This resulted in a wave of consumer products, in-home displays (IHDs) that showed energy consumption data in near real-time. Faruqui and Sergici (2010) conducted a research on the impact of IHDs on residential energy consumption. They concluded that direct feedback provided by IHDs encourages consumers to make more efficient use of energy. The authors conducted another research on the topic in 2017 (Faruqui et al., 2017) by investigating how modern energy management tools (EMTs) enabled by advanced metering infrastructure (AMI) affects energy conservation behavior. The major finding of the investigation was that AMI-enabled EMTs reduce does residential electricity consumption.

Despite the large time gap between the experiments and different approaches used, all three aforementioned works agree on the fact that better, easy-to-understand feedback does improve energy conservation behavior on residential level. Another similar aspect is that in all three experiments the main type of visualizing energy consumption were graphs and charts. Even though in their second research Faruqui et al. (2017) describe them as “user-freindly charts”, they still remain classical graphs and charts. We see it as an opportunity for improvement since in their research Herrmann et al. (2018) and Chisik (2011) show that visualization of aggregated energy consumption is unclear to people. They propose alternative visualizations (see Figure 4.2) that, based on their surveys, give better understanding of energy consumption and, what is more important, the consequences of human actions on it. The visualization Herrmann et al. (2018) proposed incorporates the use of special sub-metering systems that that are capable of measuring energy consumption of separate devices.

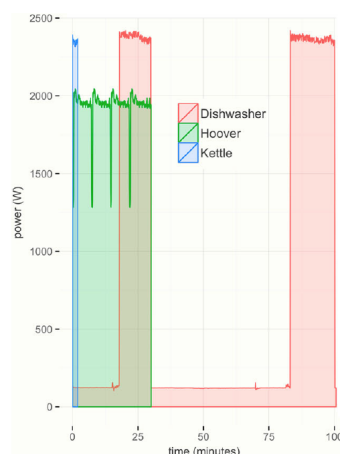


Figure 4.2: Visualizing energy consumption per appliance provides better understanding of total energy consumption (Herrmann et al., 2018)

Nowadays, however, the use of such sub-metering systems is a rarity. Smart meters that communicate total energy consumption on per residential unit basis with certain intervals are more common (Energiavirasto, 2018). Likewise, in this work we are limited to hourly aggregated energy consumption data. As it was described in the previous chapter, we managed to gain better insights from the data by applying modern machine learning algorithms to it. We can predict energy consumption for the next hour at any given time with a decent accuracy. Since the predictive model was trained on historical data, its prediction is nothing more than an expected consumption value (in kilowatt-hours (kWh)). It allows us to compare an expected energy consumption value at a given hour with an actual one. The result of this comparison will show if the actual value is higher or lower than expected. For example, Figure 4.3 demonstrates energy consumption of one day, 24 hours, from Dataset 1 plotted next to the predicted consumption for that day. Thanks to the prediction line, we can clearly see that during the day the energy consumption was lower than usual. Thus, we can translate uninformative (Chisik, 2011) kWh numbers into an easy-to-understand feedback. Our goal is to visualize the feedback in a clearest way possible and find out how it will change people’s perception of energy consumption.

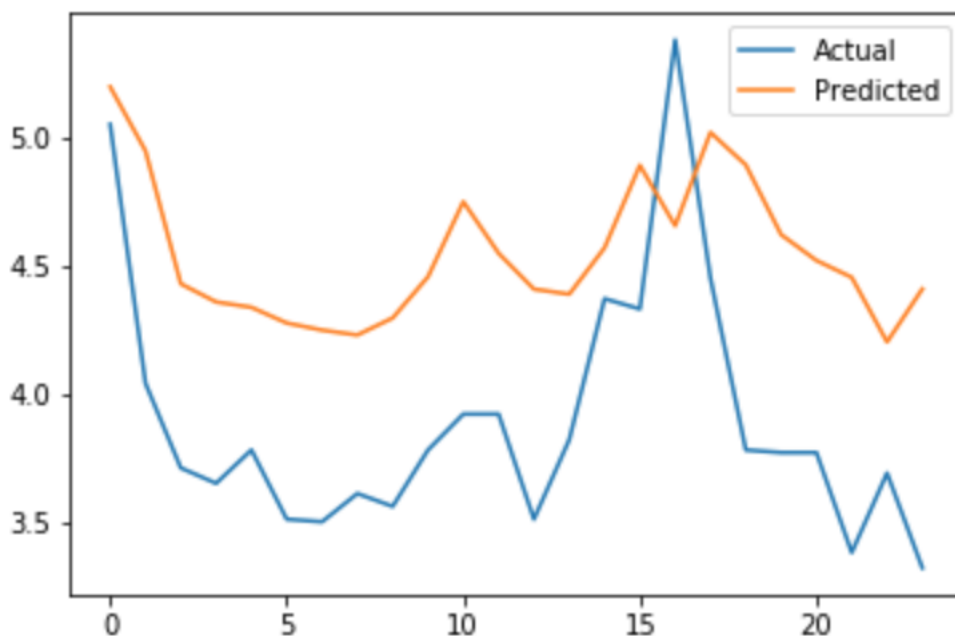


Figure 4.3: 24 hours from the Dataset 1 where the actual consumption was significantly lower than expected

4.2 Feedback visualization

As it was defined in the previous subsection, the feedback that our system provides is based on the comparison between the actual energy consumption value for a given hour and the predicted one.

However, we have not yet defined how we are going to interpret the comparison results. Naturally, we have three base states:

- Actual consumption is lower than expected
- Actual consumption is equal to the predicted/expected value
- Actual consumption is higher than expected

Even though it is our goal to keep the feedback as simple as possible to make it easier for people to understand and interpret, such rough approximation seems not informative enough. As a compromise between simplicity and informativeness, we have decided to add two additional “levels” to the comparison interpretation as well as make the middle, equal, state accept approximate equality as well. After we have established the levels, it was necessary to define what comparison results will fall into the each of them. After experimenting with the dataset, the following rules were set:

- Actual consumption (C_a) is significantly lower than predicted (C_p),
if $C_a < C_p * 0.75$
- Actual consumption is lower than expected,
if $C_p * 0.75 \leq C_a < C_p * 0.9$
- Actual consumption is approximately equal to predicted/expected value,
if $C_p * 0.9 \leq C_a < C_p * 1.1$
- Actual consumption is higher than expected,
if $C_p * 1.1 \leq C_a < C_p * 1.3$
- Actual consumption is significantly higher than expected,
if $C_a \geq C_p * 1.3$

With that set, it was time to decide which kind of visualization to use to efficiently express this 5 levels. As the main goal of this work is to provide better understanding of the energy consumption, persuasive aspect of energy consumption visualization (Morreale et al., 2015) was left aside. Focusing on simplicity, early in the process of choosing the right data visualization we have decided not to experiment with shapes and forms as they create different associations with different people (Kosslyn et al., 2003). That leaves us with, perhaps, the simplest and the most fundamental visualization - color.

An enormous amount of research was conducted on the effects of different colors and color combinations on human emotions (Gilbert et al., 2016), human associations with colors (Griffith and Leonard, 1997), the relationship between colors and human's mental state (De Bock et al., 2013) and so on. Even though the studies show various results on how people perceive colors, there is one historically established set of colors and associations in most of the countries around the world: traffic light colors. Indeed, for most of the people the transition from red to green is associated with the transition from "stop" to "go", from "uncomfortable" to "comfortable", from "bad" to "good". In terms of energy consumption, being "green" is also associated with using less energy. Therefore, we assume that people will associate the green color with an energy consumption behaviour that is good in terms of energy conservation and the red color with a behaviour that is bad in terms of energy conservation. This transition from green to red, consisting of colors red, orange, yellow, light green, green, was used for visualizing the 5 feedback levels. Figure 4.4 illustrates the colors chosen for each level.

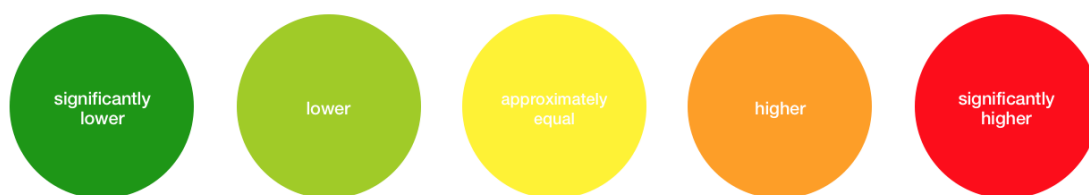


Figure 4.4: Colored circles convey a simple message: green - good, red - bad

With the conceptual design ready, it is important to build a working prototype for our feedback system to a) better understand how it will work in real life scenarios, b) evaluate how well it performs in comparison to regular graphs.

4.3 EcoFeed: a better Energy consumption Feedback system

For convenience, it was decided to give the feedback system we are building a name. 7 minutes of brainstorming have resulted in “EcoFeed”: a frivolous abbreviation of **E**nergy **C**onsumption **F**eedback.

What is EcoFeed? EcoFeed is a near real-time energy consumption feedback system. Why near real-time? Because of the one hour granularity of the data we are operating with in this work, there will be a one hour lag in the system. Nevertheless, the point is that the system constantly provides a feedback on energy consumption displaying one of the 5 colors depending on how energy consumption at a given hour differs from energy consumption predicted for that hour by our predictive model. As the feedback is continuous in its nature, we consider always on in-home displays as the most efficient way of communicating the feedback to users. Combined with the simple visualization we use, IHDs can deliver a clear idea of the households’ current energy consumption at a glance. Obviously, web interface is also a convenient and accessible way of communicating the feedback to the users but it may not be as efficient since it requires the users to access the web app manually. This is why, in our opinion, IHDs are a better communication medium for EcoFeed.

Smart in-home displays are getting more popular nowadays as companies like Facebook (Facebook Inc., 2019) and Amazon (Amazon.com Inc., 2019) enter the market with their solutions that combine voice assistants, video calling and other Internet related features. Integration of EcoFeed with such devices in an unintrusive manner is simple. Figure 4.5 shows how it can be achieved on Amazon Echo Show device by simply adding our feedback circle to the device’s stand-by screen. Another interesting, cheaper and more sustainable way of communicating the feedback can be a smart light bulb like Philips Hue (Philips, 2019). Since we use only colors for visualization, it is possible to make the light bulb efficiently deliver the feedback by changing its color.

At the prototype stage however, we assume that EcoFeed is displayed on a regular IHD with no additional info on it. That will make the prototyping process more straightforward. We have decided to build a web application for displaying EcoFeed and after we can make a survey based on that web application so the survey takers will be able to see it running in real-time and not just as static images. The latter is very important since the whole system is designed around an on-going feedback delivery.



Figure 4.5: EcoFeed can be easily integrated with modern smart IHDs

4.3.1 System Architecture

Before we start developing the prototype, it is necessary to design an architecture of the actual system. The main and only major difference between the prototype and the actual system is that in the real system the energy consumption data will be continuously fetched from the smart meters and in the prototype the data is statically stored within the system.

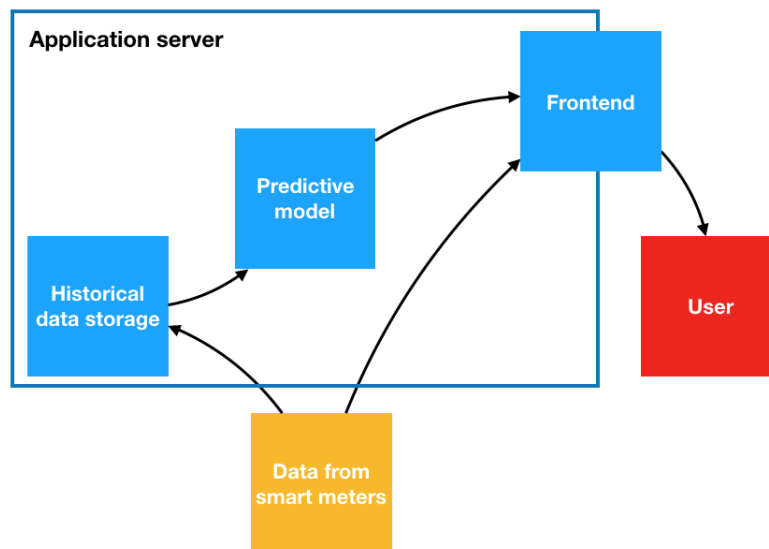


Figure 4.6: System architecture

The simplicity of the architecture makes it very flexible. Any database can be used in the architecture as long as it feeds the predictive model with data sequences in a format

that it can consume. Likewise, various technologies, frameworks and mediums can act as frontend. Even for the application server itself there are many options that will fit this architecture.

4.3.2 Developing a prototype

As it was previously described, we are building a web application as a prototype of EcoFeed. We decided to use ExpressJS (Express.js, 2019) as an application server framework. ExpressJS is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications. The framework allows us to quickly build the prototype from ground up. For frontend it was decided to use HTML + CSS + jQuery combination.

While the choice of HTML and CSS is obvious, jQuery (The jQuery Foundation, 2019) was chosen over other frameworks such as React or Angular for its simplicity and lightweightness. Creating a colored circle that changes its color based on some value does not require anything else. Whole logic can be written in JavaScript and jQuery and the circle itself is a simple div-block with css properties making it look like a circle. We further simplified the prototype by using hardcoded energy consumption and predicted values on the frontend. Thus, we have two arrays, one with “actual” values and one with “predicted” values. Then, we have written a method that takes one value from each of the arrays, compares them and based on the comparison changes the color of the circle. Simple. We have also made the method to repeat with a certain interval, therefore making the circle to change color dynamically over time. Just like it would in the actual system, but the interval in the prototype is, of course, not one hour but two seconds.

Now we have a colored circle that changes its color every two seconds based on the comparison result. To make it better convey the notion of time, we have added a label that shows what hour the consumption values belong to. Now it does actually look like a near real-time feedback system. Figure 4.8 shows how the circle changes color as states hours at which the given consumption level was observed.

Having the EcoFeed prototype ready, it is important to test it against the regular energy consumption visualization - a graph. Thankfully, there are many solutions for adding

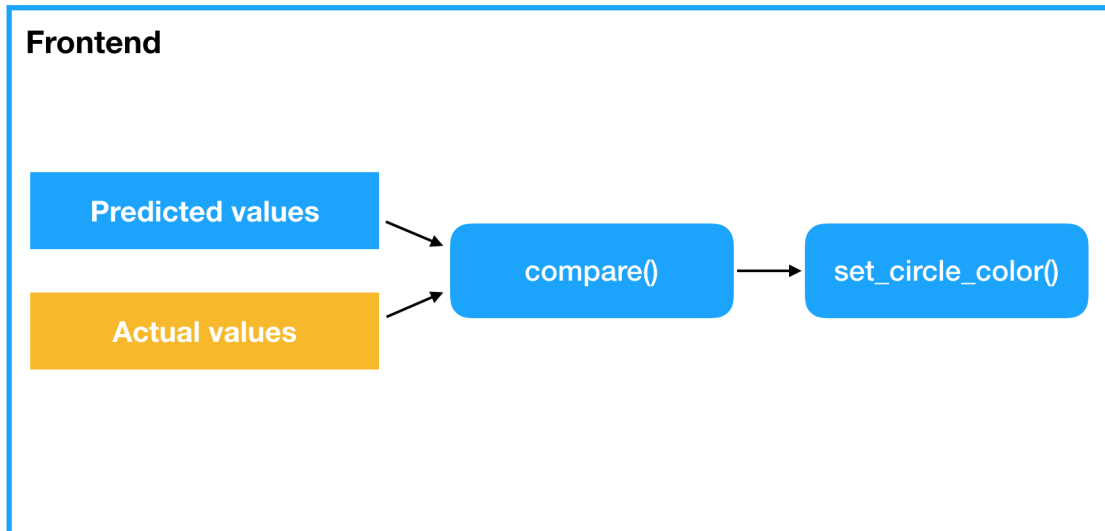


Figure 4.7: Schematic depiction of actions performed every 2 seconds in the prototype frontend



Figure 4.8: The circle changes color and time value in the center

graphs to JavaScript projects. One of the most popular of them is Highcharts (Highcharts, 2019). Highcharts provides an API which developers can use to add various charts and graphs, configure them, change their appearance, add animations and even control their behavior. Using Highcharts and the same array of actual energy consumption values that we have used for the circle, we created a graph that generally looks the same as the regular energy consumption visualizations one can find on energy providers' online platforms or other near real-time feedback solutions employing IHDs like Google PowerMeter. Figure 4.9 demonstrates the visual similarity between Google PowerMeter's energy consumption graph and the graph we have created. Note how both charts communicate the same kind of information: kWh values over time. The only difference is that Google PowerMeter shows the energy consumption pattern over 48 hours while our graph shows 24 hours only.

Thus, we can assume that the graph is similar enough to existing energy consumption feedback solutions so we can use it for evaluation of EcoFeed.

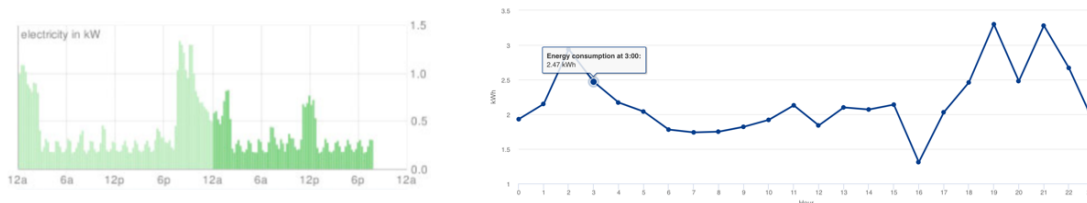


Figure 4.9: Left: Google PowerMeter, right: our graph

In order to fairly evaluate if EcoFeed provides better understanding of energy consumption to people than existing feedback solutions - graphs, it is necessary to create a survey where real people will observe both solutions and based on their responses we can find out which visualization performs better.

4.3.3 Creating a survey

The first thing that has to be done when creating a survey is to pick the right survey type. It depends on what we are trying to find by surveying people. Our problem can be formulated as following: we want to find out which of the two feedback solutions performs better. Thanks to our software development background, we know that A/B testing, or bucket testing, works perfectly for this kind of problems.

A/B testing is a randomized experiment with two variants, A and B (Kohavi and Longbotham, 2017). As the name implies, two versions (A and B) are compared, which are functionally identical but differ in non-functional aspects that might affect a user's behavior. In our case, survey participants will be randomly presented either with the graph or the circle. Both will show an energy consumption pattern over 24 hours. At the end, we will ask the participants to evaluate the energy consumption pattern in terms of energy conservation. The variant with more correct responses will be considered as a better performer.

The problem is, how do we get data that reflects good and bad energy conservation behaviour? Thanks to our predictive model, it is easy. We can just traverse through the

dataset and find a day, meaning 24 consecutive entries that start with 00:00, with the most number of entries that are below/above the predicted values. As a double check we can also manually verify that the entries are indeed lower or higher than average at the given period. Using this approach we managed to find 5 days in the dataset that fit the scenarios we need:

- 2 days with energy consumption much lower than expected
- 1 day with energy consumption about the same as expected
- 2 days with energy consumption much higher than expected

All 5 days were manually verified to actually be lower/same as/higher than average of preceding 14 days in terms of energy consumption. Each of the 5 days kWh values were saved into arrays. Now we can create a two dimensional array consisting of the 5 arrays and use it for visualization. Same was done to the predicted values for the given days. Then we had to add a functionality that will randomly pick one of the visualizations and display them to a survey participant.

It is important to note that in both visualizations the presented data will be the same and will come in the same order as following:

1. A day with energy consumption much lower than expected
2. A day with energy consumption much higher than expected
3. A day with energy consumption about the same as expected
4. A day with energy consumption much higher than expected
5. A day with energy consumption much lower than expected

Now we have a web application that upon opening a page decides which visualization to show and renders either the circle or the graph. It's a survey, we want people to evaluate the visualized energy consumption patterns in terms of energy conservation. We decided to use a simple scale from 1 to 5, where 1 - excessive use of energy/bad energy conservation and 5 - moderate use of energy/good energy conservation. We use a slider as an input for user responses. This way, no validation of responses is required.

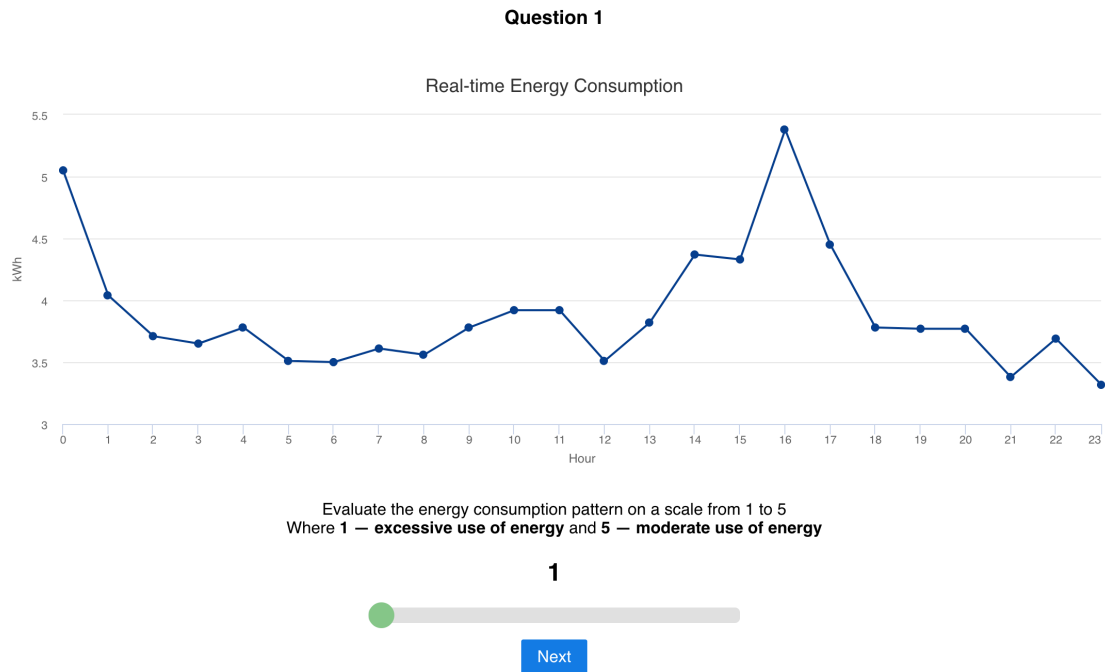


Figure 4.10: Survey question with the graph

Next, responses has to be stored somewhere, so we have used MongoDB (MongoDB Inc., 2019) for that. MongoDB is a cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with schema. MongoDB does not require hard rules for stored data, it is lightweight and easy to deploy and use. For the aforementioned characteristics, it was chosen over other solutions.

Apart from responses, it makes sense to collect basic information about the survey participants. Since we are not looking for any specific categories of people, we only ask the participants to state their sex and age group. This may help during responses analysis.

Since the main functionality was done and ready at this point, we asked several people to participate in the survey on our local machine. We observed their reaction and asked them to share their thought process. It was clear after three tries that the survey needs a thorough explanation, since the very concept of energy conservation, consumption and what exactly visualizations meant was not clear, especially with the graph. We have added explanations on the welcome page as well as instructions with each visualization showed before the actual questions start. After another round of user shadowing, it was decided

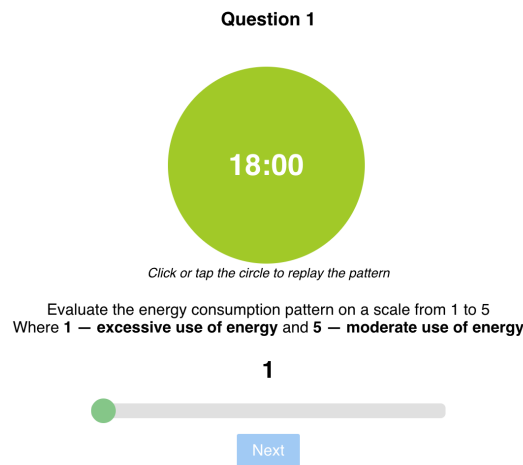


Figure 4.11: Survey question with the circle

to launch the survey. For that, we had to make it publicly accessible.

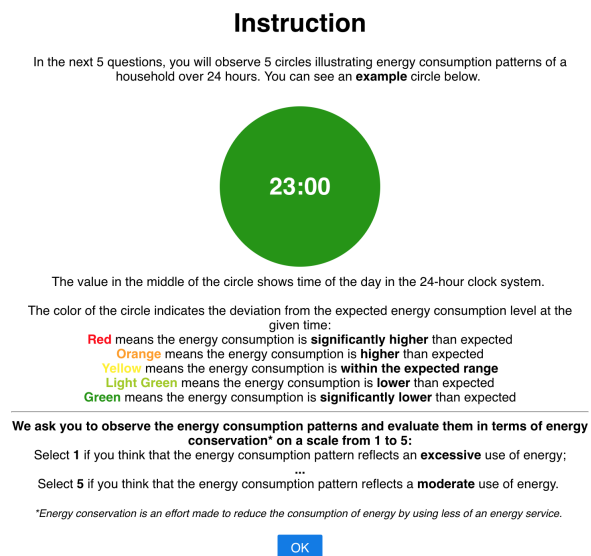


Figure 4.12: Instruction for the questions with the circle

Thankfully, the technologies we used to build the web application are easily transferable so it took less than an hour to deploy the app on an Amazon Web Services (Amazon Web Services, 2019) Elastic Cloud Computing (EC2) instance. Why EC2? First of all, because it has a publicly accessible IP address so people can take our survey online without the need to take it in our office. Apart from that, it's free tier allows us to run one EC2 instance

consistently. Perfect. The survey was ready, publicly accessible so we have shared the link on the internet. Mostly, we asked our colleagues and friends to take the survey but we have also posted the link to the survey on a data visualization and research specific platforms.

4.4 Results and discussion

In this chapter, we have proposed a different approach to providing household residents with a feedback on their energy consumption. The proposed solution is a novel way of visualizing energy consumption made possible by the insights provided by the predictive model we built in the previous chapter. We called our feedback system “EcoFeed” and developed a fully functional prototype. In order to evaluate how well it performs in comparison with well established energy consumption feedback visualization - graph, a survey was developed that assessed how well people understand energy consumption after presented with either of the two types of visualizations.

	Number of responses	Sex		Age group		
		Female	Male	18-24	25-34	35-44
Total	40	13	27	8	29	3
EcoFeed	19	6	13	3	15	1
Graph	21	7	14	5	14	2

Table 4.1: Demographics of the survey participants

Speaking of the survey, we managed to collect 40 responses in total. The number of surveys with the graph and surveys with the circle/EcoFeed was approximately the same 21 and 19 respectively. Table 4.1 describes respondees’ demographics in more details.

The survey results show significant difference in participants’ understanding of energy consumption patterns. Those who were presented with EcoFeed tend to correctly identify the patterns while those who saw the graphs seem to give random and mostly incorrect responses. Figure 4.13 demonstrates survey results broken down by questions. Note how with EcoFeed responses correlate with the actual energy conservation behaviour presented in the questions. Questions 1 and 5, featuring good behaviour in terms of energy conservation, are mostly evaluated as 4 or 5. Same with questions 2 and 4 where the majority of respondees gave 1 or 2. Even the average one, question 3, have mostly received 3s.

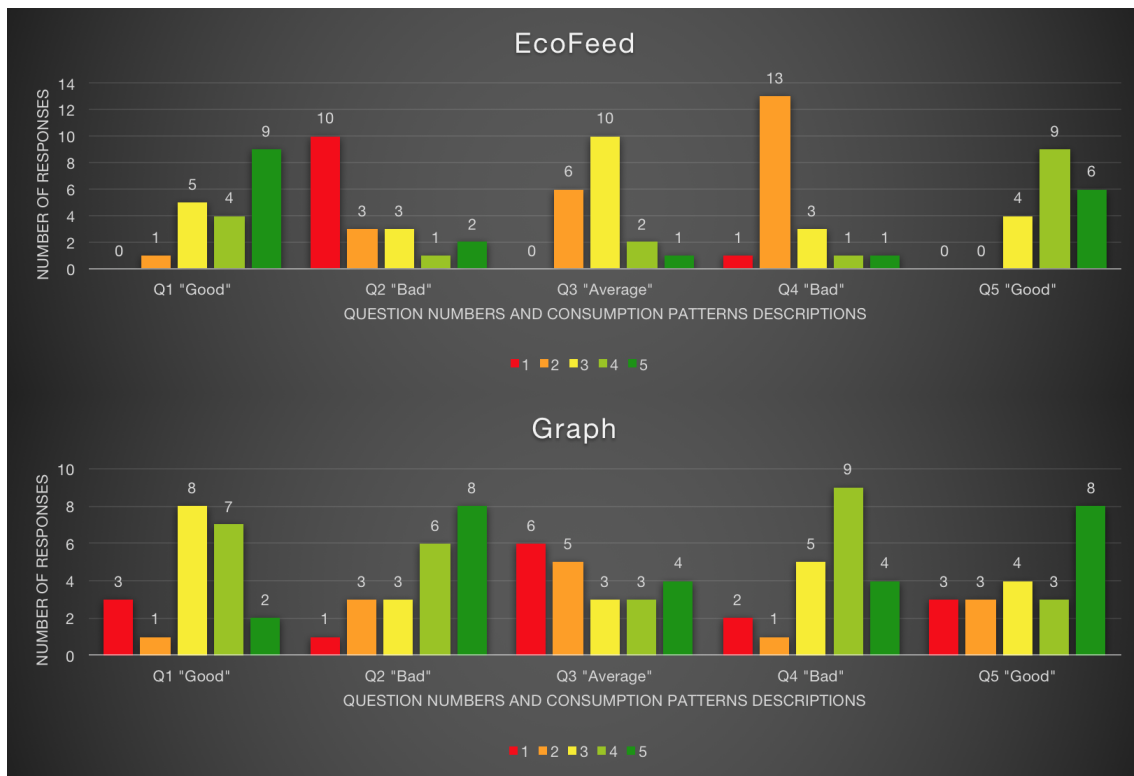


Figure 4.13: Survey results

Graph questions, on the other hand, show either random distribution of answers or even mostly wrong answers. Random answers can be explained easily since even during the participants' shadowing round we have observed that people struggle with graphs, they were asking for context. It was clear that graphs are difficult to read and understand without context. The problem is, current energy consumption feedback solutions do not provide context and present data in the same way we have done it. The reason for wrong responses is that people based their opinions on wrong characteristics. For example, we can see that the question 2 has mostly received 4s and 5s. Question 2 has baseline consumption much higher than Question 1. Because of that, it seems that survey participants assume that energy conservation was worse compared to Question 1. Even though in the instruction to the survey we clearly state that energy consumption patterns were taken from different seasons, days, thus must not be compared with each other. For Question 5, however, we observe many responses with 5s which is correct answer. That is seem to be caused by the shape of the graph, it is descending from 00:00 all the way to 23:00. Thus, we can conclude that with graphs people pay attention to features that do not describe actual energy conservation behaviour. Another important outcome of this survey is, of course, that EcoFeed proved to provide better understanding of energy consumption.

5 DISCUSSION, CONCLUSIONS AND FUTURE WORK

In this chapter we discuss the outcome of this work, how the proposed system can be used in real life scenarios, what could be done better and how it can be improved in the future.

5.1 Discussion

The results of this work show that machine learning based energy consumption feedback system is a viable solution for providing household residents with better understanding of their energy consumption behaviour. We can clearly say that the proposed solution has a lot to improve. We talk more about it in “Future Work” subsection. However, even as it is, it demonstrates great results in comparison to conventional solutions.

It seems that the main reason why EcoFeed was easier to understand for people is a good choice of visualization. While we were looking for something universal, simple to implement and test, it turned out to be efficient as well.

The constraints of this research make this system easily implementable with technologies and data that are available right now. The only thing that is currently lacking is an application programming interface (API) that would allow real-time data fetching from smart meters.

As the main objective of this whole research is to make people more sustainable, it is important to analyze effects that the proposed will incur. We are using a model proposed by Porras et al. (2017) to identify immediate, enabling and structural effects of EcoFeed on different dimensions of sustainability.

Following the model, we first identify immediate effects of EcoFeed. It is clear that introducing a new system, such as EcoFeed, over existing infrastructure requires upfront financial investments and increases its overall carbon footprint. These are negative impacts that must be considered and properly evaluated before implementing the system: will the positive impacts outweigh the negative ones? The only positive immediate effect of the system is that users get to enjoy better, more advanced service. An enabling effect of the better service is better understanding of electricity consumption by the users, which leads to more sustainable consumer behavior. Therefore, it lessens the amount of CO₂ emitted, contributing to a better overall environmental situation. As you can see, the proposed model lets us explore different impacts EcoFeed makes on individuals, so-

ciety, environmental and economic situations. Figure 5.1 demonstrates the sustainability analysis of EcoFeed.

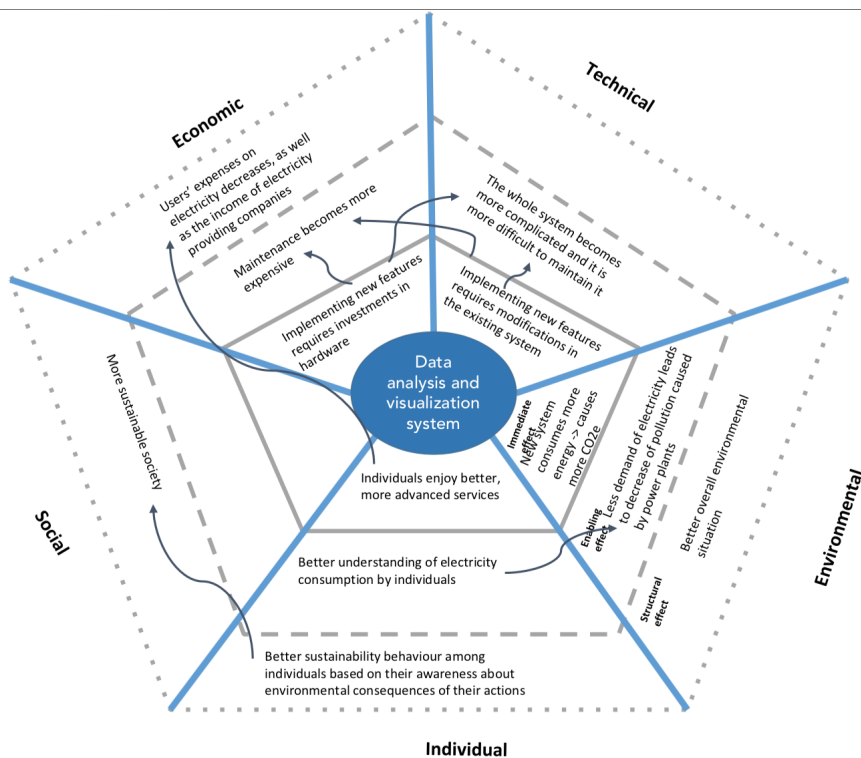


Figure 5.1: Sustainability Analysis of EcoFeed

5.2 Conclusions

In this work, we have collected an energy consumption data of two residential units in Lappeenranta, Finland. A two-storey house with internal heating and an apartment without internal heating. We then investigated how machine learning was used for residential energy consumption prediction in previous studies. After conducting literature review, it was clear that few research have focused on predicting single household energy consumption. Then the collected datasets were prepared for building predictive models. Several of the most used machine learning algorithms (Linear Regression, Support Vector Machine, Gradient Boosting, Long Short-Term Memory recurring neural network) were utilized to build predictive models from the datasets. After evaluating all of them, an LSTM model was chosen as the best performing. Another major finding at this point was that the apart-

ment's data was mostly dependant on human actions and less on other factors such as weather. This made the data very random and thus, poor for training predictive models. For that reason we had to give up the apartment's data and use only the data obtained from the larger house.

Having a model that can predict energy consumption one hour ahead with a high accuracy enabled us to create a novel energy consumption visualization. A prototype was developed in order to evaluate how well the proposed visualization communicates energy consumption to people compared to a conventional solution. We created an interactive A/B survey to compare the visualizations. Survey results showed that the novel visualization performs much better than the conventional visualization.

5.3 Future Work

This work consists of two main parts: machine learning and data visualization. Both parts have plenty of room for improvement.

The predictive model we use can be improved by itself by tweaking certain parameters like number of epochs, optimizer and so on. Completely different algorithm can be employed as well since the domain of machine learning is developing at an enormous pace. One goal can be to improve the initial dataset using open data so it will be able to produce a predictive model accurate enough to be used with smaller residential units as well. As we remember, in this work we had to give up a whole dataset because we could not make an accurate predictions out of it. Even though in this research we purposefully decided to use only open data and data from smart meters, it may worth a try to expand the datasets with detailed features *not* from open sources or make some assumptions that might drastically improve predictive capabilities of resulting models. It could also be interesting to apply our approach to more households of different kinds to see how well it stands over a larger sample.

Speaking of data visualization, results of this work can be used to create easy to understand *and* persuasive visualizations. What if instead of colors we use some forms or objects? The feedback can potentially cause more empathy if the colors will be replaced with a leaf, for example: higher the energy consumption - darker the leaf. Financial

incentive can be employed as well by adding a potential difference in electricity costs that current energy consumption behaviour is causing. Can it make EcoFeed even more efficient? Possibly. That is a topic of a whole new research.

References

- Abdelkader, S.S., Grolinger, K., and Capretz, M.A. (2016). Predicting energy demand peak using M5 model trees. In: *Proceedings - 2015 IEEE 14th International Conference on Machine Learning and Applications, ICMLA 2015*. ISBN 9781509002870.
- Amazon Web Services (2019). *Amazon Web Services (AWS)*. url: <https://aws.amazon.com>.
- Amazon.com Inc. (2019). *Amazon Echo Show*. url: <https://www.amazon.com/All-new-Echo-Show-2nd-Gen/dp/B077SXWSRP>.
- Amimeur, A., et al. (2018). Time-sensitive behavior prediction in a health social network. In: *Proceedings - 16th IEEE International Conference on Machine Learning and Applications, ICMLA 2017*. ISBN 9781538614174.
- Anaconda Inc (2019). *Anaconda*. url: <https://www.anaconda.com/distribution/>.
- Andersen, R.V., Toftum, J., Andersen, K.K., and Olesen, B.W. (2009). Survey of occupant behaviour and control of indoor environment in Danish dwellings. *Energy and Buildings*. ISSN 03787788, doi:10.1016/j.enbuild.2008.07.004.
- Ben Taieb, S. and Hyndman, R.J. (2014). A gradient boosting approach to the Kaggle load forecasting competition. *International Journal of Forecasting*. ISSN 01692070, doi:10.1016/j.ijforecast.2013.07.005.
- Benington, H.D. (1983). Production of Large Computer Programs. *Annals of the History of Computing*. ISSN 01641239, doi:10.1109/MAHC.1983.10102.
- Burkov, A. (2019). *The Hundred-Page Machine Learning Book*, 1st edn. ISBN 978-1999579500, 159 p.
- Calvert, G.A. and Brammer, M.J. (2012). Predicting consumer behavior: Using novel mind-reading approaches. *IEEE Pulse*. ISSN 21542287, doi:10.1109/MPUL.2012.2189167.
- Chen, W., et al. (2019). Early Detection Prediction of Learning Outcomes in Online Short-Courses via Learning Behaviors. In: *IEEE Transactions on Learning Technologies*. ISSN 19391382.

Chisik, Y. (2011). An image of electricity: Towards an understanding of how people perceive electricity. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN 9783642237676, ISSN 03029743.

CSC (2019). *CSC - IT Center for Science*. url: <https://www.csc.fi/>.

De Bock, T., Pandelaere, M., and Van Kenhove, P. (2013). When colors backfire: The impact of color cues on moral judgment. *Journal of Consumer Psychology*. ISSN 10577408, doi:10.1016/j.jcps.2012.09.003.

Diao, L., Sun, Y., Chen, Z., and Chen, J. (2017). Modeling energy consumption in residential buildings: A bottom-up analysis based on occupant behavior pattern clustering and stochastic simulation. *Energy and Buildings*. ISSN 03787788, doi: 10.1016/j.enbuild.2017.04.072.

Dong, B., Li, Z., Rahman, S.M., and Vega, R. (2016). A hybrid model approach for forecasting future residential electricity consumption. *Energy and Buildings*. ISSN 03787788, doi:10.1016/j.enbuild.2015.09.033.

Edwards, R.E., New, J., and Parker, L.E. (2012). Predicting future hourly residential electrical consumption: A machine learning case study. *Energy and Buildings*. ISSN 03787788, doi:10.1016/j.enbuild.2012.03.010.

Energiavirasto (2018). *National Report 2018 to the Agency for the Cooperation of Energy Regulators and to the European Commission*. Technical report. Energiavirasto. url: <https://www.energiavirasto.fi/documents/10191/0/%0DNational+Report+2018>

EU (2018). Greenhouse gas emission statistics - emission inventories. *Eurostat*. ISSN 15518949 (ISSN).

EU (2019). *Energy Efficiency*. url: <https://ec.europa.eu/energy/en/topics/energy-efficiency>

Eurostat (2018). *Eurostat - Statistics explained. Energy consumption in households*.

Express.js (2019). *Express - Node.js web application framework*.

Facebook Inc. (2019). *Portal from Facebook*. url: <https://portal.facebook.com/>.

- Faruqui, A., Arritt, K., and Sergici, S. (2017). The impact of advanced metering infrastructure on energy conservation: A case study of two utilities. *Electricity Journal*. ISSN 10406190, doi:10.1016/j.tej.2017.03.006.
- Faruqui, A. and Sergici, S. (2010). *Household response to dynamic pricing of electricity: A survey of 15 experiments*. doi:10.1007/s11149-010-9127-y, ISSN 0922680X.
- Few, S. (2004). EENIE , MEENIE , MINIE , MOE : Selecting the Right Graph for Your Message. *Intelligent Enterprise*.
- Freedman, D.A. (2009). *Statistical models: Theory and practice*. ISBN 9780511815867.
- Friendly, M. (2005). Milestones in the history of data visualization: A case study in statistical historiography. In: *Studies in Classification, Data Analysis, and Knowledge Organization*. ISBN 3540256776, ISSN 14318814.
- Gajowniczek, K. and Zabkowski, T. (2017). Electricity forecasting on the individual household level enhanced based on activity patterns. *PLoS ONE*. ISSN 19326203, doi: 10.1371/journal.pone.0174098.
- Gilbert, A.N., Fridlund, A.J., and Lucchina, L.A. (2016). The color of emotion: A metric for implicit color associations. *Food Quality and Preference*. ISSN 09503293, doi: 10.1016/j.foodqual.2016.04.007.
- Google Inc. (2019a). *Google Colaboratory*. url: <https://colab.research.google.com/>.
- Google Inc. (2019b). *Tensorflow*. url: <https://github.com/tensorflow/tensorflow>.
- Grandjean, A., Adnot, J., and Binet, G. (2012). *A review and an analysis of the residential electric load curve models*. doi:10.1016/j.rser.2012.08.013, ISSN 13640321.
- Graves, A., et al. (2009). A novel connectionist system for unconstrained handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. ISSN 01628828, doi:10.1109/TPAMI.2008.137.
- Griffith, L.J. and Leonard, S.D. (1997). Association of colors with warning signal words. *International Journal of Industrial Ergonomics*. ISSN 01698141, doi:10.1016/S0169-8141(96)00062-5.
- Grolinger, K., L'Heureux, A., Capretz, M.A., and Seewald, L. (2016). Energy forecasting for event venues: Big data and prediction accuracy. *Energy and Buildings*. ISSN 03787788, doi:10.1016/j.enbuild.2015.12.010.

- Gunn, S.R. (1998). Gunn SVM Support Vector Machines for Classification and Regression. *Image Speech and Intelligent Systems Technical Report*. ISSN 0003-2654, doi: 10.1039/b918972f.
- Gupta, G. and Rathee, N. (2016). Performance comparison of Support Vector Regression and Relevance Vector Regression for facial expression recognition. In: *International Conference on Soft Computing Techniques and Implementations, ICSCIT 2015*. ISBN 9781467367929.
- He, H.A. and Greenberg, S. (2009). Motivating Sustainable Energy Consumption in the Home. *Psychologist*.
- Herrmann, M.R., Brumby, D.P., Oreszczyn, T., and Gilbert, X.M. (2018). Does data visualization affect users' understanding of electricity consumption? *Building Research and Information*. ISSN 14664321, doi:10.1080/09613218.2017.1356164.
- Highcharts (2019). *Highcharts*. url: <https://www.highcharts.com>.
- Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*. ISSN 08997667, doi:10.1162/neco.1997.9.8.1735.
- Humeau, S., Wijaya, T.K., Vasirani, M., and Aberer, K. (2013). Electricity load forecasting for residential customers: Exploiting aggregation and correlation between households. In: *2013 Sustainable Internet and ICT for Sustainability, SustainIT 2013*. ISBN 9783901882562.
- Hunter, J.D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science and Engineering*. ISSN 15219615, doi:10.1109/MCSE.2007.55.
- Jain, R.K., Smith, K.M., Culligan, P.J., and Taylor, J.E. (2014). Forecasting energy consumption of multi-family residential buildings using support vector regression: Investigating the impact of temporal and spatial monitoring granularity on performance accuracy. *Applied Energy*. ISSN 03062619, doi:10.1016/j.apenergy.2014.02.057.
- Kaytez, F., Taplamacioglu, M.C., Cam, E., and Hardalac, F. (2015). Forecasting electricity consumption: A comparison of regression analysis, neural networks and least squares support vector machines. *International Journal of Electrical Power and Energy Systems*. ISSN 01420615, doi:10.1016/j.ijepes.2014.12.036.
- Kennedy, S. (2019). *Astral. Python calculations for the position of the sun and moon*. url: <https://github.com/sffjunkie/astral>.

- Kenton, W. (2019). *Multiple Linear Regression - MLR Definition*. url: <https://www.investopedia.com/terms/m/mlr.asp>.
- Keras Team (2019). *Keras*. url: <https://keras.io/>.
- Kohavi, R. and Longbotham, R. (2017). Online Controlled Experiments and A/B Testing. In: *Encyclopedia of Machine Learning and Data Mining*.
- Kosslyn, S., Gershon, N., Levkowitz, H., and Pearlman, J. (2003). Improving visualization: theoretical and empirical foundations.
- Koza, J.R., Bennett, F.H., Andre, D., and Keane, M.A. (1996). Automated Design of Both the Topology and Sizing of Analog Electrical Circuits Using Genetic Programming. In: *Artificial Intelligence in Design '96*.
- Kuhlman, D. (2012). A Python Book: Beginning Python, Advanced Python, and Python Exercises. *Book*.
- Leonhardt, V. and Wanielik, G. (2018). Neural network for lane change prediction assessing driving situation, driver behavior and vehicle movement. In: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. ISBN 9781538615256.
- Li, D., et al. (2016). Profiling household appliance electricity usage with N-gram language modeling. In: *IEEE International Conference on Industrial Technology (ICIT)*.
- Li, Z. and Dong, B. (2017). A new modeling approach for short-term prediction of occupancy in residential buildings. *Building and Environment*. ISSN 03601323, doi: 10.1016/j.buildenv.2017.05.005.
- Liu, Y., Wang, W., and Ghadimi, N. (2017). Electricity load forecasting by an improved forecast engine for building level consumers. *Energy*. ISSN 03605442, doi: 10.1016/j.energy.2017.07.150.
- Mckinney, W. and Pydata Development Team (2019). *pandas: powerful Python data analysis toolkit*.
- MongoDB Inc. (2019). *MongoDB*. url: <https://www.mongodb.com/>.
- Morreale, P., McAllister, J., Mishra, S., and Dowluri, T. (2015). Turning leaf: Eco-visualization for mobile user engagement. In: *Procedia Computer Science*. ISSN 18770509.

NumPy Devs. (2019). *NumPy*. url: <https://www.numpy.org/>.

Peattie, K. and Belz, F.M. (2010). Sustainability marketing - An innovative conception of marketing. *Marketing Review St. Gallen*. ISSN 1865-6544, doi:10.1007/s11621-010-0085-7.

Pedregosa, F., et al. (2011). Scikit-learn : Machine Learning in Python. *Journal of Machine Learning Research*. ISSN 0021-9355, doi: <https://dl.acm.org/citation.cfm?id=2078195>.

Philips (2019). *Philips Hue*. url: <https://www2.meethue.com/en-us>.

Porras, J., Palacin-Silva, V., Drögehorn, O., and Penzenstadler, B. (2017). Developing a model for evaluation of sustainability perspectives and effects in ICT projects. In: *International Sustainable Ecological Engineering Design for Society (SEEDS) Conference*, pp. 13–14. Leeds, UK.

Project Jupyter (2019). *Project Jupyter*. url: <https://jupyter.org/>.

Rajasekaran, R.G., Manikandaraj, S., and Kamaleshwar, R. (2017). Implementation of Machine Learning Algorithm for predicting user behavior and smart energy management. In: *2017 International Conference on Data Management, Analytics and Innovation, ICDMAI 2017*. ISBN 9781509040834.

Rencheroglu, E. (2019). *Fundamental Techniques of Feature Engineering for Machine Learning*. url: <https://towardsdatascience.com/feature-engineering-for-machine-learning>

Sak, H., Senior, A., and Beaufays, F. (2014). Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In: *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*. ISSN 19909772.

Sak, H., et al. (2015). *Google voice search: faster and more accurate*.

Scikit-learn (2019). *Gradient Boosting*. url: https://scikit-learn.org/stable/modules/gradient_boosting.html

Shoensmith, E., Vapnik, V., and Kotz, S. (2006). Estimation of Dependences Based on Empirical Data. *The Statistician*. ISSN 00390526, doi:10.2307/2988246.

- Siegelmann, H.T. and Sontag, E.D. (1995). On the computational power of neural nets. *Journal of Computer and System Sciences*. ISSN 00220000, doi: 10.1006/jcss.1995.1013.
- Smith, C. (2016). *iOS 10: Siri now works in third-party apps, comes with extra AI features*. url: <https://bgr.com/2016/06/13/ios-10-siri-third-party-apps/>.
- Stat.fi (2018). *Energy consumption in households*. url: https://www.stat.fi/til/asen/2017/asen_2017_2018-11-22_tau_001_en.html.
- Stone, M. (1974). Cross-validated Choice and Assessment of Statistical Predictions. *Journal of the Royal Statistical Society. Series B (Methodological)*.
- The jQuery Foundation (2019). *jQuery*. url: <https://jquery.com/>.
- Tso, G.K. and Yau, K.K. (2007). Predicting electricity energy consumption: A comparison of regression analysis, decision tree and neural networks. *Energy*. ISSN 03605442, doi:10.1016/j.energy.2006.11.010.
- Valecha, H., et al. (2018). Prediction of Consumer Behaviour using Random Forest Algorithm. In: *2018 5th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering, UPCON 2018*. ISBN 9781538650028.
- Vance, A. (2018). *This Man Is the Godfather the AI Community Wants to Forget*. url: <https://www.bloomberg.com/news/features/2018-05-15/google-amazon-and-f>
- Werner, V. (2016). *Bringing the Magic of Amazon AI and Alexa to Apps on AWS*.
- Wijaya, T.K., Vasirani, M., Humeau, S., and Aberer, K. (2015). Cluster-based aggregate forecasting for residential electricity demand using smart meter data. In: *Proceedings - 2015 IEEE International Conference on Big Data, IEEE Big Data 2015*. ISBN 9781479999255.
- Wikipedia (2019). *Gradient Boosting*. url: https://en.wikipedia.org/wiki/Gradient_boosting
- Wilhite, H. and Ling, R. (1995). Measured energy savings from a more informative energy bill. *Energy and Buildings*. ISSN 03787788, doi:10.1016/0378-7788(94)00912-4.
- Wood, G. and Newborough, M. (2007). Energy-use information transfer for intelligent homes: Enabling energy conservation with central and local displays. *Energy and Buildings*. ISSN 03787788, doi:10.1016/j.enbuild.2006.06.009.

Wu, Y., et al. (2016). Google's NMT. *ArXiv e-prints*.

Zhang, B., Miller, D.J., and Wang, Y. (2012). Nonlinear system modeling with random matrices: Echo state networks revisited. *IEEE Transactions on Neural Networks and Learning Systems*. ISSN 2162237X, doi:10.1109/TNNLS.2011.2178562.

Zhang, F., et al. (2016). Time series forecasting for building energy consumption using weighted Support Vector Regression with differential evolution optimization technique. *Energy and Buildings*. ISSN 03787788, doi:10.1016/j.enbuild.2016.05.028.

Zhao, H.X. and Magoulès, F. (2012). *A review on the prediction of building energy consumption*. doi:10.1016/j.rser.2012.02.049, ISSN 13640321.

