Short-term electric load forecasting in households

Experiments with multiple linear regression

Project Report
DAT300

By

Noräs Salman
Marco Bresch

Chalmers University of Technology
Gothenburg, Sweden 2016

Abstract

An important paradigm shift has appeared in human life since people were able to use electricity in their households. The demand for more electric power has increased over the years, which made researchers and world leaders think about sustainable solutions to reduces this demand and search for alternatives. Researchers and governments are putting their efforts to develop a new solution for the electrical grid, and one of these solutions has been manifested in the smart grid, a computer controlled version of the electrical grid. While the smart grid project is an innovation not fully implemented in the current electricity infrastructure, it has shown the need for studying the variables that lead to the increase of the electric demand in order to be able to reduce the load, thus reducing the cost and the demand for electricity.
Electric load forecasting is one way to predict the load that a certain city, area or household will need in the future. While forecasting is not a new subject of study, the idea of being able to apply it in a consumer household is. One of the more challenging problems is to find the right input parameters that will play the main role in such prediction process.

In this project we present a brief overview of the science behind electric load forecasting, in which we cover definitions and what algorithms could be used. Furthermore, we experiment with short-term forecasting for one household based on an existing data set using a multiple linear regression algorithm where we produce two models and execute them using three different data sets. The experiment aims to compare the output of different variations of input parameters that we considered as features for our prediction model, and presents the results accordingly.

The results shows that it is very important to define the scope of the desired model and what error bound is acceptable. It also shows that there is a trade-off between accuracy and speed of performance when using a sliding window concept.

Contents

List of Figures

List of Tables

1
Introduction

If we try to look up the word forecasting in a dictionary it is *"predict or estimate (a future event or trend)"* [9]. In the modern society, forecasting has been widely used to estimate or predict events and values. For example, forecasting can predict nature causes like weather, earthquake and floods probabilities. It can estimate the performance of a team in the future such as a sales team in a company or a football team playing a game against another one. More importantly it can predict cost and value changes like stock market, housing market and your electricity bill.

Another field of application and the one this project will focus on, is to forecast the electrical load of a small household. The different paradigms of electrical load forecasting are classified into short-,medium- and long-term forecasting and referring to the time period we want to forecast. Short-term forecasting can be used for a period of one hour to one week, medium-term forecasting for a few weeks up to one year and long-term forecasting for up to twenty years.

The input for an electric load forecasting model, is determined by factors that should be taken in consideration when designing a model, the algorithms that could be used for this purpose and their problems. The concept of the sliding window, could be used to focus and isolate the forecast for a specific time frame.

In this project we use the multiple linear regression algorithm to perform an hour ahead short-term load forecasting in a household. The algorithm could be used in a small household to predict the electric load in the next hour and use the output of it for a scheduling algorithm to scheduled the consumption thus the cost of the electricity to show its relation to the humongous field of the smart grid, the computer controlled version of the electrical power grid. The experiment is limited to compare two models we considered, based on previous conducted work. For the purpose of learning and how to design,implement and execute the multiple linear regression we followed the lecture notes for the machine learning course from Stanford university [8], together with a set of papers [7, 1, 10, 5, 2] to project on how to use it for the task of load forecasting in the best way possible.

This report is organized as follows. In section 2, we give a theoretical background for the subject *electric load forecasting*. We then give a short abstract explanation of the system model in section 3. Section 4, gives a comprehensive explanation about how the steps of building a model using the multiple linear regression algorithm. In section 5, we present two models for the purpose of *one hour a head electrical load forecasting*. We also present three data sets we constructed, execute the two models and present the results, followed by an analytical discussion of the results and finish with possible future work. Finally, in section 6, we give our conclusion and describe the learning outcome.

2
Theoretical Background

2.1 Electric Load Forecasting

If we look at the power systems design, we will notice that the system begins with power generators that are the source for electricity to further distribute it to the grid. The generators are distributed across the grid and designed with two main functionalities. The first one is to reduce the power loss on the cable that is the result of the transmission for long distances. The second one is to provide redundancy, so if one of the generators fail, another one that has not reached its maximum capacity can take over. These operations and decisions require the knowledge of the demand in the future and forecasting is used for this purpose to forecast the peak demand, typically for short-term[5]. One of the hardest tasks in the power grid is the requirement to directly and always meet the load requested instantaneously. Putting too much pressure on the generator forces it to slow down, thus affecting the system's performance. And due to the remarkable variation in load between different time periods, it is very important for the operators to know what the demand is going to be in the next few hours or weeks.

From an industry point of view, electrical load forecasting is an important technique for producing the right amount of energy to meet the demand during a given time frame. Furthermore, electrical load forecasting can be used in a single household by combining the acquired knowledge from the forecasting with efficient scheduling algorithms, which can lead to economical benefits by reducing the consumption according to the actual demand. For example, if there was a single person living in a small household, we can identify different patterns of activity. For instance, one of these periods is the work schedule of the person. During the working week the person is usually at work during the majority of the day and not using any devices. If we can predict these different periods using forecasting and combine a scheduling algorithm, we can decrease the amount of energy while the person is at the work, thus reduce the consumption and the cost.

2.1.1 Types of electric load forecasting

Electric load forecasting is divided into three major categories: Short-term, Medium-term, and Long-term with respect to the time frames it can predict in the future. Each type of forecasting is used for different purposes in the industry. Table 2.1 gives a short description for the different types of load forecasting and examples of usage as mentioned by Kyriakides et. al [5].

Table 2.1: Types of Electric Load Forecasting

Type	Cover Period	Usage
Short-term forecasting	One hour to one week ahead	**(Household)** can help to estimate load flows. **(Provider)** make decisions that can prevent overloading.
Medium-term forecasting	Few weeks up to one year	**(Electricity Provider)** used for scheduling maintenance, scheduling of the fuel supply and minor infrastructure adjustments.
Long-term forecasting	Twenty years	**(Investors and Governments)** for planning purposes such as constructing new power stations, increasing the transmission system capacity, and in general for expansion planning of the electric utility.

2.2 Short-term electric load forecasting

In this project we are focusing on short-term electric load forecasting. As described in Table 2.1 short-term forecasting can prevent overloading as we described in section 2.1 and predict the consumption for the next minute, hour or week.
In the following sections we list the algorithms that could be used and describe the factors which are affecting the forecast in the short-term period together with their known problems that have been mentioned in the studied literature.

2.2.1 Influencing factors for short-term load forecasting

When constructing a short-term electric load forecasting model we have to take in consideration by what factors the load is affected. The main factors are categorized as following:

1. Historical data, including time of the day, day of week and the load at that time.

2. Weather, including indoor and outdoor temperature, wind and humidity.

3. Social factors, such as income, mealtimes during the day that may differ from a country to another, work hours and the house owners class.

4. Random factor, as there is always unpredictable peaks that appears in a random manner.

5. Other factors, such as the space and the type of property, national holidays the amount of green energy generated locally.

If we take a look at the models described by the papers in which we investigated this topic, we see that Kyriakides et al. [5] mention factors used in general and the system parameters, weather, and historical data, they also present Figure 2.1 that shows an example of a short-term load forecasting system with its' input parameters. We can see how the short-term load forecasting model uses the offline collected historical, weather and systems parameters data as an input. Rothe et al. [10] uses three parameters for the weather wind speed, humidity and temperature together with the load at a certain time and the loads before that. Tuaimah et al. [7] shows that in their model that weather has very important role and the make two model instead for one generic model, taking in consideration that they are modelling for the use in a hot country. Amral et al. [1] divide to type of buildings to residential, commercial,Industrial and public and construct a model for each type of the buildings. Fernandez et al. [2] say that we should make a distinction between weekday and work day and use what so called "*work calendar*" by making a model for each day, and set of hours has a independent model.

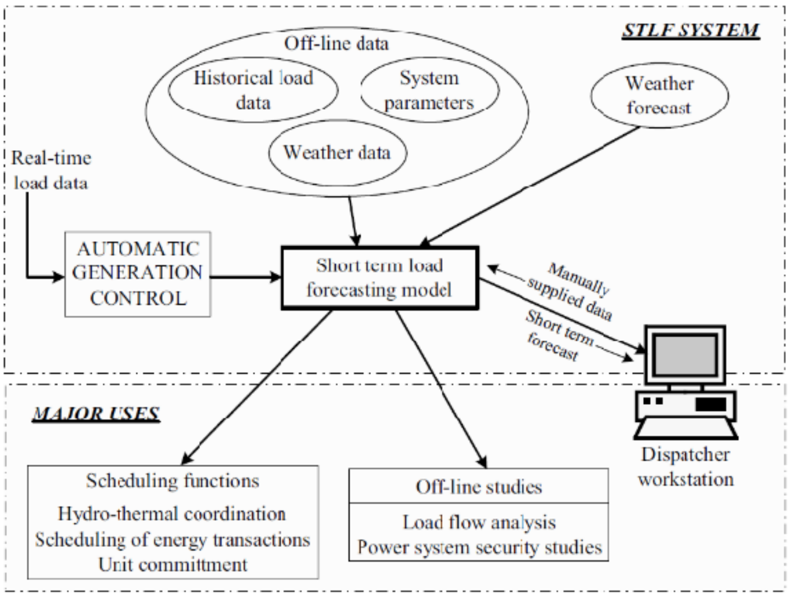


Figure 2.1: Shot-term load forecasting parameters example by [5]

2.2.2 Methods and algorithms

There are a lot of algorithms that have existed for a long time when it comes to forecasting. Kyriakides et al. [5] survey a set of different methods specifically for short-term load forecasting and classify them into classical and intelligent methods. The classical methods include the methods that are based on statistical numerical equations and series such as linear regression and time series. The other set of algorithms are more complex and using artificial intelligence methods such as neural networks, support vector machines , expert systems and fuzzy-logic systems.

Each algorithm has its advantages and disadvantages. For example, time series is is used when periodical data is taken in particularly time intervals and is suitable if no new changes appear to the variables that affect the load, such as the environmental variable or the social variables. Moreover, it requires significant computational time because of the amount of data required for training, thus can not be used in systems that need to perform a forecast in a very short time period and adapt to new changes at the same time. Regression on the another hand, does not require a lot of data for the training thus takes less time for training. However, the resulting forecast is less accurate compared with other algorithms. Neural networks provides a good black box that adapts according to the internal flow of information through the network during learning. Despite the fact that neural networks give good forecasting results, it is hard to use with known periodical information as an input, because it requires continuous training data in addition to the fact that it also requires longer training time and can not be used on-line.

In order to choose the right algorithm, the scope of the system that we want to apply the forecast on should be taken in consideration. Armal et al.[1] classify the short-term load forecasting from another prospective into two models. The *peak load models* and the *load shape models*. In peak load models, the daily or weekly load is modeled as a function of weather and time does not play a role in such models. Where the load shape models uses data with time intervals and is mostly based on historical data.

When it comes to performance, there are certain algorithms that perform better than others. Fernandez et al. [2] present a comparison study for evaluating the accuracy of four chosen algorithms and list a set of problems for each algorithm, the problems will be referred to in the next section.

2.2.3 Restrictions of short-term load forecasting algorithms

Fernandez et al. [2] explain a set of known problems when it comes to load forecasting algorithms. Among the problems mentioned, there is no generic way to perform short-load forecasting. It is very hard to have a single algorithm with a set of input parameters that will perform well in any environment, which makes the property of adaptiveness to other different systems a hard task and requires to construct a new model for each problem targeted.

Feature selection is another problem, as each model requires a different model based on the factors we mentioned in the previous section, not to mention that adaptiveness to new changes and trends or sudden changes in the system is hard to perform.

Last but not least, algorithms that have a learning phase suffers from problems such as, the need to require a big amount of data for learning, and the fact that they tend to *over-fit* i.e be too good from learning of the training sets provided in the learning phase resulting in inaccurate predictions for values outside the training sets.

While it is very hard to over come all of these problems, in our project we picked the linear regression for the fast performance it provides in the learning phase. In addition to that, we bypass the over-fitting by testing the resulting models using foreign data which is unknown by the models.

3
System Model

3.1 Design of a forecast model

Generally speaking, when we want to construct a prediction model for a system, we have a group of components that will be used during the process. We will describe this by using the input-process-output(IPO) model [3] and introduce the following definitions.

Definition 1. The set of input values given to the system denoted x_i, is transformed to produce the output value y that represent the predicted value.

Definition 2. The hypothesis $h(x)$ transforms the input values x_i to the output y.

The main difference between different algorithms is how to define the form of the *hypothesis*. This can be very simple as in linear regression where the hypothesis is defined as a linear equation, or in time series where it is defined as a set of numerical series. However, sometimes it could be not obvious in advance, but a result of *learning process* like in neural networks.
The learning process is the result of a training phase that requires a large set of data that contains previously known correct values of x_i and y, this set is called the *training set*. A *learning algorithm* will optimize the values of the training set is repeated until reaching convergence. Figure 3.1 illustrates the general process of prediction with the different components mentioned above.

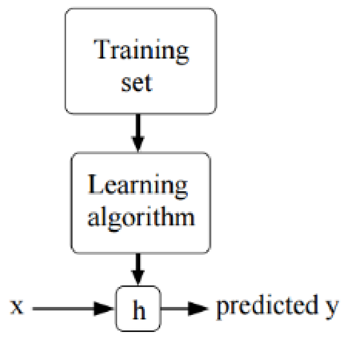


Figure 3.1: General process of prediction [8]

3.2 Data

In this project we are using a data set named *"Individual household electric power consumption Data Set"* [4] which is a part of the UCI machine learning repository project [6].

The data set contains 2,075,259 records gathered every one minute during almost 4 years, to be exact in a (47 months) period with a corresponding time-stamp. It has consumption values for two rooms and a thermo-control system. The first room is the kitchen and contains a microwave, oven and a dishwasher. The oven operates on gas so it has no effect on the value presented. The second room is a laundry room that contains a washing-machine, a tumble-drier, a refrigerator and a light source. The third value represents the consumption for an electric water-heater and an air-conditioner.

We chose this data set because of the huge number of records it contains. This allows us to use a complete subset of the data with the less number of missing values for more accurate experiment results (see section 3.2.1) . In our project, all the records will be added and treated as one value, which is the total load. The time-stamps will be also be transformed to a useful feature, we show this in a more detailed way in section 3.2.2.

3.2.1 Challenges

The data set we mentioned previously has a list of problems that will affect the accuracy of our forecasting model. First, and as mentioned in the dataset's documentation page [4] there are some values missing on a periodical basis from the measurements. This will have an obvious effect on the output in our model. Second, we mentioned on the background section when we talked about the factors that have a strong effect on the forecasting result is the weather information, especially when it comes to predicting peaks. Finding a dataset with weather information was a task near to impossible when looking at the time frame assigned for this report to be produced. For this reason we will train our model using winter data like the paper by Tuaimah et.al [7] where they have two models, a winter model and a summer model for forecasting data.

3.2.2 Data pre-processing

In order to make the data that we have usable, we have to perform several transformation operations on it and transform it into useful features. Since we have our hand only on historical data for the electrical load with date/time-stamps, we transform them to usable integers in our linear equation.

Since we do not have any weather data, we propose a winter model like presented by Tuaimah et.al [7] and assume that the weather has a constant effect during winter. This will affect the accuracy of our model as weather information is used to determine sudden changes in the load, such as peaks, in load shape models [1].

We chose to transform the date string to one integer that represents the day of the week. The values range from Monday=0, Tuesday=1,... ,Sunday=6. The time stamp will be converted to one integer that represents the hour ranging from 1 to 24. The actual load, we will not only be used as a target value in the training process but also be used in a model that is *based on prior knowledge*. This means that the values from previous hours will be used to predict the load at hours that will follow them, this method is also used in [10, 7]. Note that the data set contains a record for each minute, in our case we minimize the number of records and perform sum all the record that is within the range of one hour. For example, all records between 00:00 and 00:59 are aggregated into one value at the hour value of 1.

In order to compare accuracy when running the algorithm with the different features, we create three different data sets which we present in table 3.1. For this reason we use *Python* to transform the existing dataset to the datsets *D1,D2,D3* presented.

D1 contains the least information compared to the other dataset. The motivation behind choosing this dataset in its simple form, is to show the level of accuracy it can give using only simple time information.
In D2, we added one extra variable to investigate the effect of using a variable that gives information about the load for one previous hour. Whereas D3, we added, six more variables that gives information about the previous six hours to show the importance of knowing the load shape pattern for more than one hour.

Table 3.1: Pre-processed data sets

Dataset Name	Features
D1	• x_1 = Day of the week *[0..6]*. • x_2 = Time of the day *(Hour)*. • y = Current demand *(Watts)*.
D2	• x_1 = Day of the week *[0..6]*. • x_2 = Time of the day *(Hour)*. • x_3 = Load on previous hour *(Watts)*. • y = Current demand *(Watts)*.
D3	• x_1 = Day of the week *[0..6]*. • x_2 = Time of the day *(Hour)*. • x_3 = Load on previous hour *(Watts)*. • x_4 = Load on previous second hour *(Watts)*. • ... • x_9 = Load on previous sixth hour *(Watts)*. • y = Current demand *(Watts)*.

4
Experimental Method

4.1 Introduction

In this section, we give a comprehensive explanation about the multiple linear regression algorithm we used. Moreover, we introduce the sliding window concept to extend the algorithm and acquire more accurate results. We finish the section by presenting the methods we considered for forecast performance evaluation .

4.2 Linear regression

Given a training set as an input, containing the features $x_i \in X$ and a target output $y \in Y$. The goal of linear regression is to learn a function $h_\theta(x)$ so that $(h : X \to Y)$ gives predictions for the value y for a set of unknown values x_i [8, 7, 10, 1]. Where the function $h_\theta(x)$ is the hypothesis. To chose the learning algorithm we have to first define the form of our hypothesis $h_\theta(x)$. In linear regression, the hypothesis is decided to approximate y as a linear function and can be defined as shown in equation 4.1 bellow.

$$y = h_\theta(x) = \theta_0 + \theta_1.x_1 + \theta_2.x_2 + ... + \theta_n.x_n \tag{4.1}$$

The θ_i values, also called *weights*, parametrize the space of our linear function. The summation of the values θ_i where $i \in [1..n]$ gives the slope, that represents the consistent change between values x_i where $i \in [1..n]$ and y.

The intercept term θ_i which is a constant is used to represent the difference between the target value y and the summation of $\theta_i.x_i$ where $i \in [1..n]$ as shown in equation 4.2

$$\theta_0 = y - \sum_{i=1}^{n} \theta_i.x_i \tag{4.2}$$

4.2.1 Determine the intercept term

Since we are aiming to predict the shape of the load curve, predefining the intercept is a difficult task. Thus, the best way to determine the intercept value is to use the intercept which results during parameter estimation when using the training data. To accomplish this, we introduce $x_0 = 1$ to our hypothesis then we can represent it as shown in equation 4.3 bellow:

$$\sum_{i=0}^{n} \theta_i.x_i = \theta^T.X \tag{4.3}$$

4.2.2 Stochastic gradient descent

Gradient descent is an optimization algorithm [8]. The main goal of gradient descent is to find the optimal values for the θ constants in our linear equation. This is done by reducing the error value $(y^{(i)} - h_\theta(x^{(i)}))$ which is the difference between the original value and the predicted value of our hypothesis function.

In order to do that we need to adjust our θ values in every learning step taken to minimize the *cost function* as much as possible to reach convergence. For our cost function we considered the *Least Mean Square (LMS)* and presented in equation 4.4 bellow.

$$J(\theta) = \frac{1}{2}\sum_{i=0}^{n}(h_\theta(x^{(i)} - y^{(i)})^2 \tag{4.4}$$

LMS gives a way to measure the sum of square errors i.e accuracy. The motivation for choosing LMS lies behind the ease of implementation even for thousands of features and the simple way of intercepting the resulting value. However, LMS is prone to have difficulties in performance when having too many variables, as well as it gets affected by outliers in the training set e.g. some points in the training data may have excessive larger or smaller values than others.

Putting the mentioned above together, we define our *update rule* for the θ values as shown in equation4.5.

$$\forall j \in N; \theta_j := \theta_j + \alpha\sum_{i=0}^{m}(y^{(i)} - h_\theta(x^{(i)})).x_j^{(i)} \tag{4.5}$$

Where α is the learning rate. In our program we define the *number of iterations* which correspond to how many times the optimization should be done during the training process.

4.2.3 Feature scaling (Normalization)

Feature scaling is the process of standardizing the range of independent variables [8]. When we have multiple values with no relation that affects the prediction , we need a way to scale those values to fit one space . Consider the following example. Imagine we want to predict the annual electricity consumption for a household based on two factors. 1- the space of the house 2- the number of bedrooms that it has. In the figure 4.1 you can see that we have a three dimensional space. X the number of bedrooms, Y the space and Z being the price. In order to scale it to a two dimensional space X,Y we have to perform what so called feature normalization.

Feature normalization is done in two steps. First we calculate the mean value (μ) if each feature from the data set and subtract it from its value. Second we divide the values we got from first step by their respective standard deviation (σ). Equation 4.6 shows the process of feature scaling.

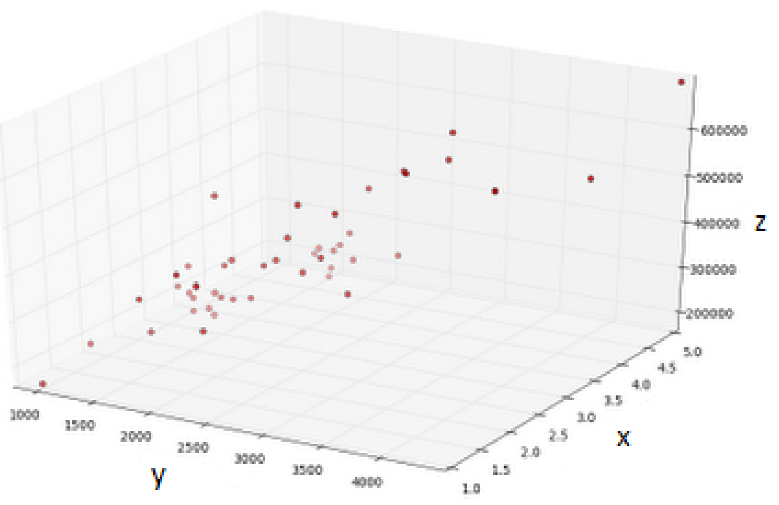


Figure 4.1: Feature Scaling example

$$x' = \frac{x_i - \mu}{\sigma} = \frac{x_i - mean(x_i)}{max(x_i) - min(x_i)} \tag{4.6}$$

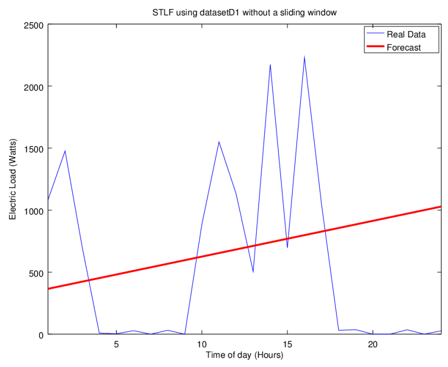
4.2.4 Linear regression - Full algorithm

To summarize the section we present the steps for the linear regression algorithm we used in our project as following:

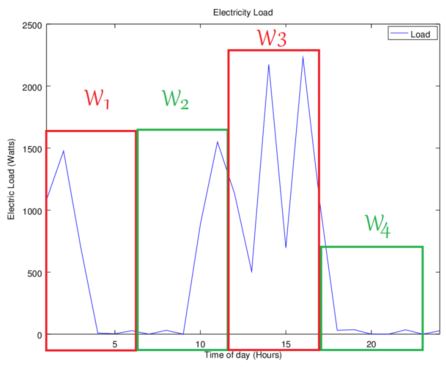
1. By calculating (μ) and (σ), scale the the values for each x_i ; where $i = 1..n$
2. Push the value $x_0 = 1$ to be able to learn the intercept term from the training.
3. Set initial values for $\theta_i = 0$; where $i = 1..n$
4. Training: Run gradient descent and repeat it with respect to (number of iterations).
   (a) Calculate the cost function $J(\theta)$.
   (b) Adjust the next value of θ_i by the α rate specified.
   (c) Go back to 4(a).
5. Prediction: Using new *unknown* values for the model x_i.
   (a) Scale the new values using the same (μ) and (σ) from 1.
   (b) Solve the hypothesis $h(x)$ using the resulting values θ_i from step 4.

4.3 The sliding window concept

Due to the non-linear nature of the electric load, applying linear regression on the a data set constructed from historical data (like D1 from 3.2.2) may not be the perfect idea. Applying it directly will result in a straight line prediction estimate which does not make any sense or is useful in anyway like the one presented in figure 4.2 (a). While the execution of the algorithm is not wrong, applying it on the wide range during the different hours is. Therefore we need do divide the day into several time frames which is useful, especially since people tend to have a repetitive consumption behaviour, for example, people tend to eat breakfast in the morning and dinner in the evening around the same time.



(a) Linear regression without sliding window on dataset D1



(b) Separating the day to four windows with 6 hours window size

Figure 4.2: The sliding window concept

If we look at figure 4.2 (b) where we split the day in four windows *(w1,w2,w3,w4)*, we can see that in *w1,w2* and *w4* there is no change of the direction of the predicted load curve i.e not accurate. In *w1* the load goes down, in *w2* it goes up and *w4* there is nearly no consumption. In this case using the linear regression on those windows appears to be a better way to perform the forecast. If we look at *w3* we can see that it may also be a good idea to separate it to several windows because of the changes in the direction of the load line, as we can also divide it into several windows.

To do that, we have to isolate the day from the dataset and divide the day into several windows. This is also similar to the *work calendar* approach mentioned by Fernandez et al. [2] where they have a model for each day and this day is separated to several models as well.

Finally, if we want to add this to the previous section 4.2.4, we need to perform one step of data processing and two steps using the linear regression algorithm.

1. Isolate the data for the day we want to forecast (e.g Mondays) from the current dataset.

2. Split the day to ψ windows, where each window has its own dataset D_{w_i}.

3. For each window w_i; train using the dataset D_{w_i} and use the hypothesis $h_{w_i}(x)$ to perform the forecast.

Notice that such separation will result in number of hypothesises that is equivalent to number of windows denoted ψ multiplied by the number of days in the week as presented in equation 4.7 .

$$\#hypothesises := \psi X 7 \tag{4.7}$$

4.4 Performance evaluation

In order to calculate how well a forecasting model is performing, we need a way to evaluate it. A well known mathematical formula for evaluating forecasting algorithms, is Mean Absolute Percentage Error (MAPE) [1, 2, 7] and is defined as shown in equation 4.8

$$MAPE := \frac{1}{N_h}\sum_{i=1}^{N_h}(\frac{\mid actualload - forecastedload \mid}{actualload})X100 \tag{4.8}$$

Where N_h is the number of hours the forecast will predict. In the execution of our project we will also use the minimum and maximum error values for the testing datasets defined as following

$$MinError := Min(|y - prediction|) \tag{4.9}$$

$$MaxError := Max(|y - prediction|) \tag{4.10}$$

5

Experimental Study

5.1 Execution

We have two forecast models that perform our forecast, we will reffer to them as Model I and Model II.

The first model, **Model I**, does the training using D1,D2,D3 separately (see 3.2.2) and performs the forecast using the algorithm presented in section 4.2.4. For example, the execution using the training set D1 is done by giving the model the dataset D1, then the linear regression is performed and outputs three values θ_i where $i \in [1, 2, 3]$.
For evaluation, we calculate the MAPE value for each execution together with the minimum error and maximum error values for full same training set and for a previously unknown single day. The single day evaluation will be used later to compare it with the output of Model II.

The next model, **Model II**, performs a sliding window forecast on a chosen day using the whole data set as an input as we described in section 4.3. We execute this model on the training sets D1,D2 and D3 separately, using 2,4,8 and 12 windows for each training set. Since we do not have any features in our data sets that gives a priority to a day than the other, we perform this on a single day and chose to do it on the first day in the week which is Monday.
For evaluation we calculate the MAPE value for each execution together with the minimum error and maximum error values.
For the task of deciding the best windows size we execute the forecast on a full week and compare the MAPE values for each window count to investigate the relation between the accuracy and the window count.

The linear regression settings that were used for both models are defined as following:

1. Number of iterations = 400 iterations .
2. The learning rate $\alpha = 0.01$.

5.2 Results

In this section we show the result of executing our models using the three data sets we introduced earlier.

5.2.1 Hardware performance

We chose linear regression because of the fast performance that it provides. Training and producing new values for the hypothesis can be done online when not performing any prediction. As we chose to perform an hour a head forecast every hour, by testing we found that, for Model I, the whole training process did not take more than 15 seconds when running it on a computer with Intel Core 2 Duo with 4 GB memory. The same result was provided when running it on a Raspberry Pi model B. As for Model II the training took 20%-25% more time than Model I, when using 2 windows and increases linearly with the increase if the window count. This is expected as Model II requires $[7\psi - 1]$ more training times than Model I, performed on subsets of the training set, where ψ is the number of windows.

5.2.2 Model I

Since the dataset D1 does not include features that help to determine when the load changes (lower or higher) while applying the linear regression without a sliding window i.e *Model I*, the prediction values when using D1 will be not accurate. If we look at Figure 5.1 that represent an execution of Model I on the data set D1,D2 and D3 for a Monday, we will notice that the result of executing the model on D1 produces a straight line which does not look any close to the actual load. When executing Model I on D2 and D3 on the other hand, shows a better accuracy that is closer to the original values.

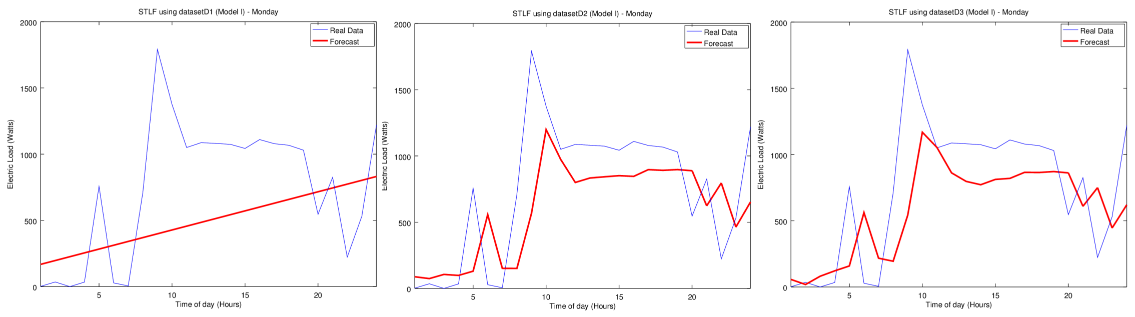


Figure 5.1: Model I execution with D1,D2 and D3. From left to right

Table 5.1 shows the MAPE results for a single day. We can see that the MAPE for D2 is lower than for D3, but if we look at the different values for the minimum error, we can see that D3 provides an better value than D2.

Table 5.1: Model I execution- Single day evaluation

Dataset Name	MAPE	Max Error (Watts)	Min Error (Watts)
D2	27.167	1223.5	39.185
D3	27.603	1248.5	3.2331

Table 5.2 shows the MAPE results that we got for using Model I on all the values (three months) in the data sets D2 and D3. It is noticeable that the MAPE

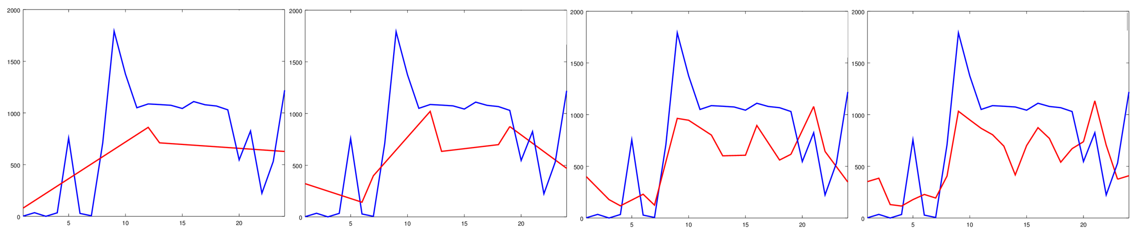
result for D3 is slightly lower than for D2, but the minimum and maximum error is higher than for D2.

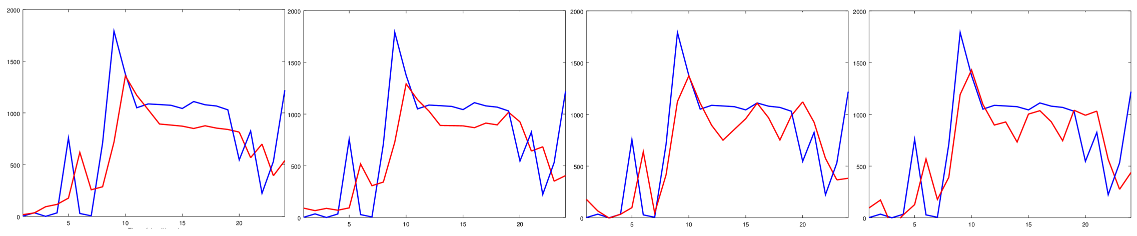
Table 5.2: Model I execution - Full dataset evaluation

Dataset Name	MAPE	Max Error (Watts)	Min Error (Watts)
D2	27.595	2840.6	0.63397
D3	27.572	2857.4	0.90668

5.2.3 Model II

The result of executing Model II using the data sets D1,D2 and D3 and different windows sizes for each dataset is shown in figure 5.2. When looking at the plot when using the data set D1, we can see the effectiveness of performing the sliding window and the difference in the plot from the one in figure 5.1 where we had one straight line that is increasing. When using the data set D1, we can see how the forecast curve started to change direction by using two windows. As for the data sets D2 and D3, we see that the forecast curve is getting smoother and reducing the error difference when comparing figure 5.1 and figure 5.2, especially when using more than two windows in Model II.


(a) Using dataset D1


(b) Using dataset D2

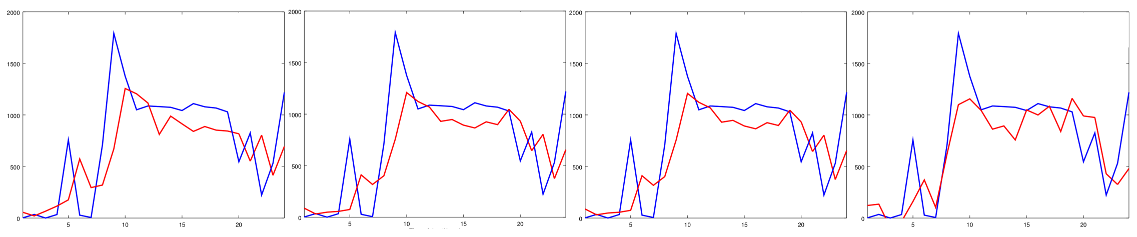

(c) Using dataset D3

Figure 5.2: Execution of Model II on D1,D2 and D3 using 2,4,8 and 12 windows (from left to right) on the day *Monday*. Blue is *actual load* and Red is the *forecast*

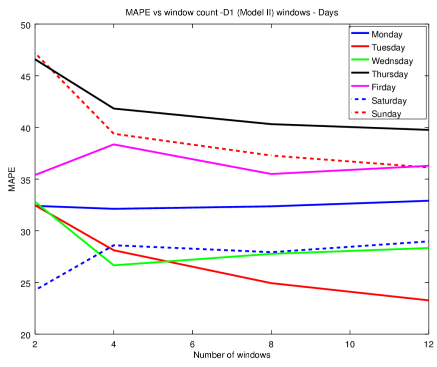
Table 5.3 summarizes the different results we got for executing MODEL II for a single day, with different window counts. The first observation is, that the MAPE percentage is gradually decreasing when moving from D1 to D2 to D3 with respect to the windows count in a non linear way.

If we take a closer look at the MAPE for D1 and the different window sizes, we can see that in contrast to the overall decrease, it increases again, with an increasing size of the windows. In order to finding out why this is happening and to decide which window size is the most effective, we are executing MODEL II on the different data sets again, but this time for a whole week, which can be seen in figure 5.3.

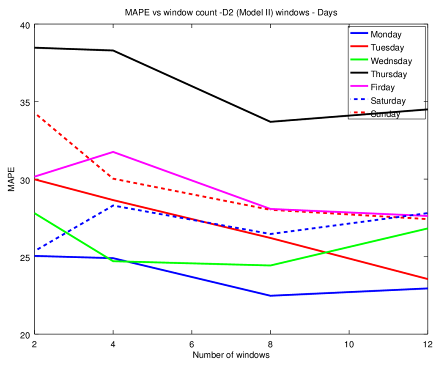
Table 5.3: Model II execution- Single day evaluation

Dataset Name	Window Count	MAPE	Max-Error (Watts)	Min-Error (Watts)
D1	2	32.408	1143.3	77.818
	4	32.123	1144.2	19.644
	8	32.364	870.51	34.797
	12	32.904	810.11	81.703
D2	2	25.045	1072.7	1.7515
	4	24.899	1063.8	12.741
	8	22.475	836.94	0.079652
	12	22.949	782.04	6.4399
D3	2	25.100	1120.1	14.427
	4	23.036	1034.9	3.5694
	8	21.198	663.57	2.0952
	12	20.575	739.49	4.9826

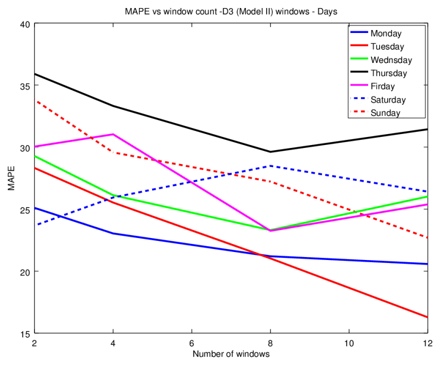
We have noticed that most of the time when we are increasing the number of windows we were getting better results when looking at the MAPE values. With figure 5.3, we performed the forecast for each day using our three data sets, and for each data set we calculated the MAPE values for 2,4,8 and 12 windows. If we look at figure 5.3 (a) we can see a all the days except for the Friday the MAPE value are decreasing when increasing the window count. In figure 5.3 (b),the first half of the MAPE values for the days have the same behaviour as the in (a) and the second half decrease until 8 windows but increase again at 12 windows, where the majority in 5.3 (c) increase at 12. In general we can say that the best window size we can have in this case is 8 windows not 12.



(a) Using dataset D1



(b) Using dataset D2



(c) Using dataset D3

Figure 5.3: Execution of Model II on D1,D2 and D3 showing MAPE percentage according the window count in a full week

5.3 Discussion

We used the knowledge we got from the papers, to construct a winter model for a small household by generating a set of different training sets were each one has different types of data and used the same error measurements that the papers use i.e MAPE. In addition to that we used the minimum error and the maximum error values to give us a little more in depth look of the error committed by the model for each hour predicted.

We can clearly see the using a data set like D1 with Model I is not the best choice as it does not produce a result that is near to the desired result.Since the linear regression output is a linear function and the nature of the electric load is not, we had to apply a sliding window concept where we perform the prediction only on a window of time in a day to see if the the average load is increasing or decrease thus detect sudden changes.
When applying the sliding window using Model II using D1, we saw that we could predict the shape of the load curve in a more accurate way. This is because each window help with defining a slope that is diffrent for other windows, thus determine whether the load in this window is increasing during this time window or not. Using

D1 with Model II has its disadvantages e.g a higher error, but we could identify some advantages when using this dataset, e.g it requires less storage for the prediction, because we only using two different values.
When looking at D2 and D3, we can see that both data sets are more accurate than D1. The main reason for this observation is, that both of them performing the forecast based on some previous knowledge. Using these data sets in Model I, proofs not significant difference in the performance. But looking at the results for D2 and D3 with Model II we can see an increase in the MAPE values. In contrast to D2 the maximum error values for D3 where decreasing.

If we compare the performance of the two models we can see that Model I requires less processing than Model II, because Model II requires identifying in which window the forecast should be performed first to determine which hypothesis to solve in order to give a forecast value. Looking at them from another aspect, Model II is more accurate than Model I in terms of forecast performance and produces less minimum error and maximum error.

Finally, in Model II, we saw how the MAPE values got better when increasing the number of windows used, but this appeared to have a limit, too many windows does not necessarily means a better forecast performance.

5.4 Future Work

In this section we list a set of problems that can make use of our project that we presented. We first want to point one more time that the data set used in this project has missing values and has not weather information. As we presented in the beginning of this project, weather information is tightly coupled to the accuracy of the forecasting model, therefore new data set with such data will definitely improve the accuracy of the algorithms presented. No need for big adjustments to the code when doing that because the code we wrote adjusts dynamically to any data set as long as:

1. It has the first feature as the day of the week.

2. The last feature as the actual load that is targeted for the prediction.

Scheduling is a hot topic when it comes to lowering peaks and the cost of electricity. Such algorithms can make use of short-term forecast for the demand and act accordingly. This requires training a model then only use the *theta values* we explained in the algorithm to solve the *model's hypothesis*.

Privacy is another concern when it comes to the future smart grid. The ability of performing forecasts gives the attacker to observe and predict users behaviour of using the electricity. Furthermore, the attacker may for example determine whether the household owners are at home or not, or when they eat their meals. This project can be used for such study as well.

6 Conclusion

There exists a wide variety of algorithms that can be used for electrical load short-term forecasting, ranging from classical methods like linear regression, which we have used for this project, to computational intelligence methods, e.g. neural networks. When and how a certain algorithm should be used, depends on the use-case and the given input data.

Even though the algorithms are quite different in their nature, one thing the algorithms have in common is, that the main challenge is how to define a correct training set, that can be used by these algorithms.

Another consideration to keep in mind, is that the electric load is non-linear in nature and changes in the load in hard to predict.Therefore , a need for a value in the training that points to a previous time (minute, hour, day) can help the model to change the direction (to lower or to higher), and having more than one is even better as the model can learn a pattern rather than having only one indication.
In the same context, we demonstrated how the sliding window concept can be applied. The designed algorithm showed an increase in performance when dividing a day into several hour windows, where each window in that day of the week performs the forecast after training, using only the data records that are similar properties. We also showed that the windows size for such model requires a careful test to determine the best number of windows.

Finally, it is important to have a certain diversity in your chosen input, e.g. historical-, social- and weather data. In this context our prediction model can be greatly improved by adding weather and temperature data to the historical data that we have used.

Bibliography

[1] N. Amral, C. S. Ozveren, and D. King. Short term load forecasting using multiple linear regression. *2007 42nd International Universities Power Engineering Conference*, 2007.

[2] I. Fernández, C. E. Borges, and Y. K. Penya. Efficient building load forecasting. In *IEEE 16th Conference on Emerging Technologies & Factory Automation, ETFA 2011, Toulouse, France, September 5-9, 2011*, pages 1–8, 2011.

[3] A. Goel. *Computer Fundamentals*. Pearson Education India, 2010.

[4] G. Hebrail and A. Berard. Individual household electric power consumption data set. `https://archive.ics.uci.edu/ml/datasets/Individual+household+electric+power+consumption`, 2016. [Online; accessed 24-Sep-2016].

[5] E. Kyriakides and M. M. Polycarpou. Short term electric load forecasting: A tutorial. In *Trends in Neural Computation*, pages 391–418. 2007.

[6] M. Lichman. UCI machine learning repository. `http://archive.ics.uci.edu/ml`, 2013. University of California, Irvine, School of Information and Computer Sciences [Online; accessed 24-Sep-2016].

[7] F. M.tuaimah and H. M. A. Abass. Short-term electrical load forecasting for iraqi power system based on multiple linear regression method. *International Journal of Computer Applications IJCA*, 100(1):41–45, 2014.

[8] A. Ng. Cs229 lecture notes. `http://cs229.stanford.edu/notes/cs229-notes1.pdf`. Standford University [Online; accessed 20-Sep-2016].

[9] Oxford. Forecast meaning. `https://en.oxforddictionaries.com/definition/us/forecast`, 2016. [Online; accessed 08-Oct-2016].

[10] J. P. Rothe, A. K. Wadhwani, and S. Wadhwani. Short term load forecasting using multi parameter regression. *CoRR*, abs/0912.1015, 2009.