A Neural Network Approach: Combining Green Energy Production and Electricity Price Forecasts to Support Green Decision Making

Brian Bak Laursen, 20071275

Kristan Barret, 20073457

Master's Thesis, Department of Computer Science, ICT Product Development

March 2013

Advisor: Niels Olof Bouvin

AARHUS
UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

Abstract

►in English. . . ◄

Resumé

►in Danish. . . ◄

Acknowledgements

►. . . ◄

Brian Bak Laursen,
Aarhus, March 13, 2013.

Contents

Abstract **iii**

Resumé **v**

Acknowledgments **vii**

1 Introduction **3**
1.1 The Problem 3
1.2 The Method 5
1.3 The hypothesis 6

2 Related Work **7**
2.1 Machine Learning 7
2.2 Green Energy Production 9
2.2.1 Prediction 9
2.2.2 Applications 11
2.3 Electricity Prices 11
2.3.1 Prediction 11
2.3.2 Applications 16
2.4 Electricity Demand Prediction 16

3 Neural Network **19**
3.1 Unsupervised learning 21
3.2 Supervised learning 21
3.3 Problems to be avoided 23

4 Conclusion **25**

Primary Bibliography **25**

Secondary Bibliography **29**

Chapter 1

Introduction

1.1 The Problem

Renewable energy has become increasingly important. Energy suppliers make it possible for their customers to choose between and brown energy. Companies and persons promote themselves with green profiles to get a certain kind of image and it has become a choice of the individual company or home to be green. This is only one part of the picture, because the increased attention on green energy is influencing the market forces behind the scenes where acquisition of energy begins [20]. Various trading companies buy electricity on the deregulated energy market and since the amount of renewable energy in this market is increasing and will do for the years to come[11, 20] the traders need to consider it carefully when buying and selling. The most essential task and basis for any decision making in the power market is to forecast the electricity prices [18].

The electricity market consists of two instruments in order to facilitate trading between producers and consumers of energy: the pool which works as an online marketplace, and a framework to make physical contracts between both sides possible [22]. The producers can make their electricity available and the consumers can bid correspondingly. An auction is made every hour to decide the market clearing price and which bids are accepted for that hour. The consumers and producers need to predict the hourly clearing prices to make plans for which bidding strategies to use. If a trader has a precise day-ahead prediction of the prices it will be easier to maximize profit by using the best possible strategy for exactly those conditions [22]. The Electricity traders will attempt to buy electricity in a low-price market and sell it when the price has come up like in the commodity market [**?**]. With a precise prediction they know when to buy and when to sell.

Approximately 25% of the total power production in Western Denmark was covered by wind power in 2008 and could therefore influence the prices on the power market [20]. The influence is seen in lower spot-market prices when the production of wind power is higher. Since the amount of green energy in the entire Danish liberalized market is targeted to increase to 30% by 2025 which

is almost a doubling since 2008 [20] the influence will only become greater. The price impact from windmills will be even greater in strong wind periods and in areas with congestions in the power transmission capacity [20]. The increase in renewable energy makes the ability of predicting green energy production and the corresponding market price more vital. It is not trivial to predict wind production because green energy by nature is unpredictable, e.g. wind turbines only generate energy when the wind is blowing. If the traders are able to predict wind production when doing actual price forecast it will give rise to a huge advantage in the market when deciding to buy or sell - be aware that the price forecasting itself besides from wind production contains many factors that need to be considered equally [22].

Traders buy and sell in real-time, intra-day or day-ahead [**?**]. This puts constraints on a price prediction algorithm - but what constraints depend solely on the type of trade. If the trader wishes to buy in real-time or hour-ahead the algorithm must perform and deliver a result within minutes, seconds or perhaps even milliseconds. If it is day-ahead then the time-interval can be increased. This might result in a compromise on the margin of error because fast results make less time for analysis and computation. Another fact is that the closer you get to the time of the trade the more accurate the weather predictions will be which directly impacts the price prediction algorithm using it. The ability to make both short- and long-term forecasts is very important in the deregulated competitive electricity market because it helps the trader to reduce risks in terms of under/over estimating the revenues from potential sales and this of course also makes it easier to manage risk[22].

Demand has a huge influence on the electricity price. No demand, no electricity price, but if these were the only two variables to consider, a linear regression model could be used for establishing a relationship and thereby a prediction. Other dynamic elements have an impact on the price and therefore the linear regression approach would result in the presence of serial correlation in the error [22]. For example will overestimated energy prices for one year most likely lead to overestimates the next year because it does not account for all the dynamic parameters. It is necessary to carefully consider all variables and characteristics of the price that we are trying to predict in the electricity market and then use a model that handles correlated errors based on the this. The most noticeable characteristics are [22]:

- High frequency;
- High volatility;
- Unusual prices in times of very high demand.
- Calendar effect (weekend, holidays)
- Multiple seasonality

Price forecasting that takes these characteristics into consideration can be modelled with a neural network that embodies crucial information about any decision making [8]. The forecast will be based on a neural network that uses highly interconnected processing units for the different parts of the calculation. This approach models in some way how the human brain itself solves a specific task [8][1]. One unit for temperature, one for wind speed, one for electricity demand, etc,.[6]

When dealing with prediction of green energy prices it is necessary to look at the weather in more detail. In this thesis we will be focusing on energy that originates from windmills.

Wind energy prediction can be divided into two areas [2]: 1) time-series analysis of power data and 2) wind speed prediction and conversion to power. What needs to be utilized in wind energy prediction is wind speed and wind direction which of course has a big influence on how much energy a wind turbine generate. An approach to analysis and prediction of wind power generation is presented in [17] where an Artificial Neural Network is used to predict the production. The prediction is based on weather data such as wind speed, relative humidity but also for how many hours the windmill is generating power.

We will base our work on neural networks and the resilient backpropagation (RPROP) algorithm used to train the network in order to predict electricity prices and wind power generation. It is faster than its predecessor Backpropagation [34, 25]. The RPROP algorithm is a learning algorithm that uses weights to analyze the input dataset and to configure further learning [21]. It is commonly used on the feedforward architecture and is the most used and implemented algorithm [30, 21].

1.2 The Method

The prediction algorithm will be based on historical data which potentially could be a very large dataset. The more historical data included in the algorithm the more data needs to be processed. We need to investigate how much of this data is necessary for precise prediction and perhaps make tradeoffs to ensure performance for decision support.

Based on the dataset that is available and the nonlinear nature of price forecasting and energy price prediction we want a system that is able to develop and learn from the past. Artificial Neural Networks can be categorized as Machine Learning [12] and are networks that imitate the behaviour of the human brain [8]. This gives us the power to be able to forecast energy prices based on how the prices have evolved in the past and therefore we chose to use neural networks to base our model upon. Our decision is based on the nature of neural networks and how it is used in machine learning. It is often used in non-linear statistical analysis [32] and has been used to predict the prices of brown energy [1][6]. We have chosen to use a multi-layered feed forward architecture since

this is one of the most used and widely implemented in open-source frameworks [21].

1.3 The hypothesis

In this dissertation we take our outset in the growing need for predicting green energy prices. It is our intent to show that we can predict green energy prices by gathering weather data from online sources and combine it with historical price development data in a prediction algorithm based on a Back Propagation Neural Network. In our concrete case the prediction itself is to be used as decision support by energy traders in their attempt to beat the electricity market. We will be focusing on 1) building a Back Propagation Neural Network that is able to predict prices; 2) creating a web service for real-time fetching of the predicted prices.

The goal of the thesis is to investigate whether or not Back Propagation Neural Networks(BPNN) are a proper way to predict energy prices. The BPNN shall as a minimum be able to predict energy prices based on historical data like in [1] and weather data. Based on this minimum requirement we will explore if 1) the prediction algorithm can come "some percentage" close to real-world data (the percentage needs to be defined!!); 2) the algorithm can be expanded to also include prediction of green energy prices; 3) real-time fetching of prices.

Chapter 2

Related Work

2.1 Machine Learning

Machine learning algorithms can figure out how to perform tasks by generalizing and where the goal is to generalize beyond the examples of a training set [12]. The programs using the algorithm learn from data and as more data becomes available, more complex problems can be handled. Systems such as web search, spam filters and stock trading uses this approach.

Many learning algorithms exist and they are all based on combinations of three components [12]; 1) the **representation** component makes an attempt to discover how to represent the input during training of the algorithm. What kind of information expressed in the data decides what representation algorithm to use; 2) the **evaluation** component has the objective of distinguishing between the best and the worst output from the representation component and let the next component decide which choice is the best; 3) finally, the **optimization** component needs to search for the highest scoring output from the evaluation and then deliver it as a result. A greedy search could be an example of such a search (see figure 2.1).

Representation	Evaluation	Optimization
Instances	Accuracy/Error rate	Combinatorial optimization
K-nearest neighbor	Precision and recall	Greedy search
Support vector machines	Squared error	Beam search
Hyperplanes	Likelihood	Branch-and-bound
Naive Bayes	Posterior probability	Continuous optimization
Logistic regression	Information gain	Unconstrained
Decision trees	K-L divergence	Gradient descent
Sets of rules	Cost/Utility	Conjugate gradient
Propositional rules	Margin	Quasi-Newton methods
Logic programs		Constrained
Neural networks		Linear programming
Graphical models		Quadratic programming
Bayesian networks		
Conditional random fields		

Figure 2.1: The three components of learning algorithms [12]

Machine learning cannot rely on data alone. Every learner must make assumptions beyond the data that is given to generalize beyond it [12]. It is not possible to map the real world uniformly to all possible mathematical functions. For example we could have a lot of knowledge about what makes two examples similar and thereby making an assumption that they are both of the same type. We can't say with 100 percent certainty.

It can be the case that the system is not at all able to deliver a correct answer. This could result in the system guessing to the best of its ability or simply randoming an answer. This problem is called overfitting [12]. Overfitting can be divided into bias and variance, where bias is when the learner consistently learns the same wrong thing over and over again. Variance is the learners tendency to learn random things and ignoring the possibility of a more correct answer. See figure 2.2 for an example with dart throwing.

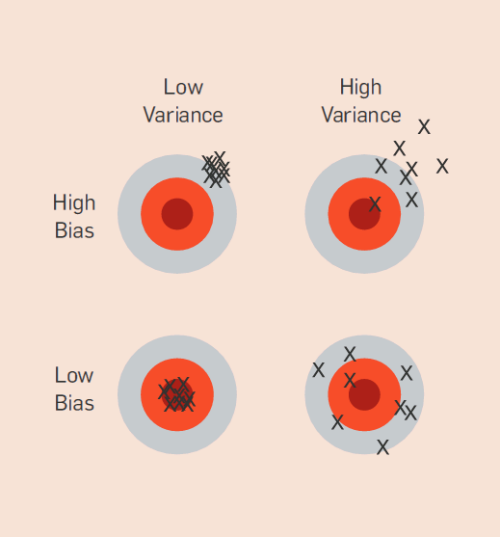

Figure 2.2: Example of bias and variance in a dart throw [12]

Machine learning is not only about the technical stuff but it also relies a great deal on intuition, creativity and black art [12].
A way to achieve machine learning is by using a Neural Network. One way to model a neural network is to base it on the Levenberg-Marquardt algorithm [24, 23, 28]. It is a least squares algorithm that, opposed to Backpropagation algorithm[34], is very fast at calculating the results. It is a curve fitting algorithm mostly used on nonlinear problems with a lot of unknowns[Need citation]. The algorithm can be used with neural networks and information approximation[34] and it can be used in combination with the backpropagation algorithm to be used on a feedforward architecture of the neural network[15].

2.2 Green Energy Production

2.2.1 Prediction

The identification of rich wind resources have become important together with the increasing focus on green energy [17]. It is important to analyse and predict the wind power generation at a certain location before placing actual windmills. Wind speed, relative humidity and generation hours of the windmills are used as input for a Artificial Neural Network in [17]. It can come as no surprise that the meteorological factors like wind speed and air density have a huge impact on the wind power generation. Figure 2.3 shows how the monthly energy generation increases with the monthly average wind speed.

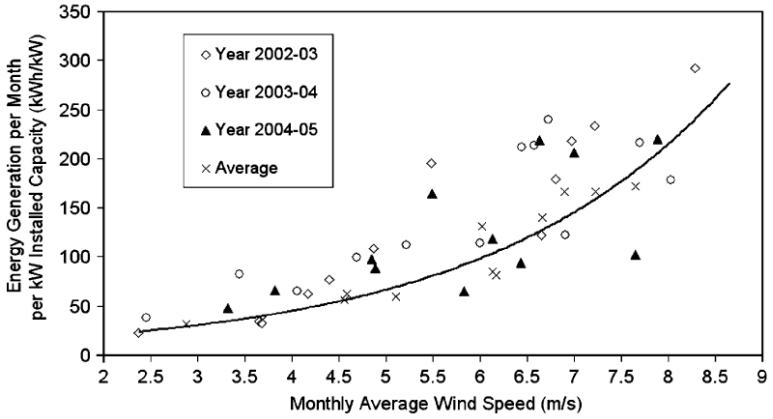

Figure 2.3: The influence of wind speed on the energy generation [17]

The more "heavy" the air, the more energy is received by the windmill turbine. The humidity increases as as air density increases and because wind energy is proportional to air density the prediction algorithm needs humidity as input because it also accounts for temperature and pressure [13]. Moist air is lighter than dry air because water molecules are less dense than the molecules in dry air such as oxygen and nitrogen. This basically means that the more air molecules like oxygen and nitrogen the more wind energy.

The last parameter in their prediction algorithm is generation hours which is the period in which the turbines produce power. The number of hours are influenced by f.x mechanical breakdowns, scheduled maintenance and low wind speeds. It is clear the the more generation hours the more energy is produced as seen in figure 2.4. The generation hours are hard to predict but can be calculated from past years up-time together with the expectations of the company delivering the windmills.

The Artificial Neural Network trains 3-year dataset containing the mentioned input parameters. The input parameters are during the training compared to the output variable which is the wind energy output of the turbine. See figure 2.5 for the architecture.

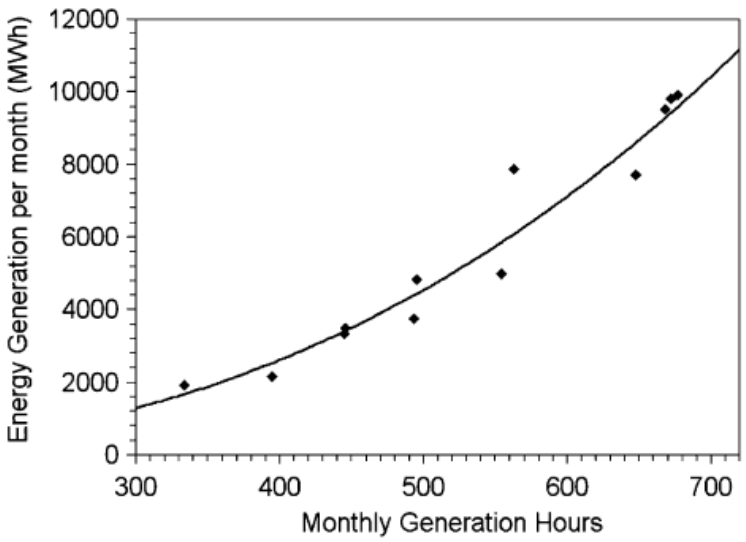

Figure 2.4: The influence of generation hours on energy production [17]

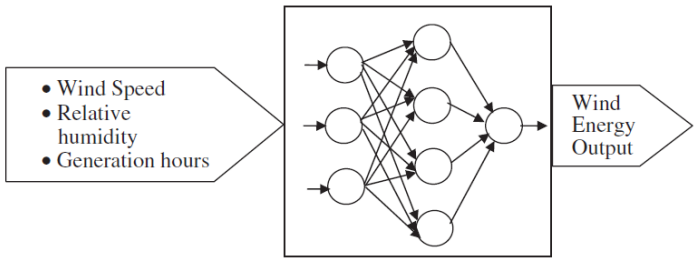

Figure 2.5: Artificial Neural Network architecture from [17]

Another approach is seen in [2] where the algorithm produces weighted nearest-neighbor (NN) tables to generate wind power curves based on available wind speed and direction from an online weather-data source. The weighted approach allows the algorithm to adapt to seasonal changes by weighting newest results highest, and the power curves makes it possible to use the algorithm on different wind farms. This prediction is used to schedule jobs in data centers which is described in the Applications subsection. The vitality of the forecasts are also seen when it comes to placement of wind farms. Before investing billions of dollars in a new wind farm it is of utmost importance to put it in the best possible location in relation to wind statistics, e.g. force of the wind and how often the winds change in power and direction at the position. In [19] they use MCP to predict wind statistics based on large amount of geographical-, weather- and historical data so that the farms can be placed best possible. This knowledge can in some way be transferred to our specific purpose but because we are dealing with decision making in real time we cant use as heavy algorithms that do calculations for several weeks and months before giving a result. We must deliver the most accurate estimate without delaying the traders

work. There is also a big difference in using billions of dollars on windmills. The companies have to be 100 percent certain that they can produce energy where you put them. When doing decision support we only aid the trader in his daily work by giving him a real time estimate for the next hour or perhaps the next week - he/she still needs to take the final decision based on her/his experience and knowledge. In addition, we can also assume that the wind mills are placed in a the best possible location where we only need to predict the energy production based on the current weather.

Another source of green energy is solar panels. Solar prediction algorithms are very accurate when the weather conditions are steady but in changing weather a large error margin is seen. To predict power output from a solar farm the time-series prediction algorithms and Estimated Weighted Moving Average (EWMA) models can be used [2].

2.2.2 Applications

It is often necessary for data centers to run long-running batch jobs where performance is measured in number of completed jobs and throughput. In [2] they design an adaptive job scheduler that utilizes prediction of solar and wind energy production to scale workload (see introduction). Earlier work has focused on using immediate available green energy and then cancelling and rescheduling jobs thereafter. The point in [2] is instead to scale the number of jobs to the expected availability of green energy production by predicting it beforehand. This helps reducing the number of cancelled jobs as the jobs are then scheduled for whenever the energy is available. If the amount of green energy production is not sufficient for an immediate or emergency job the remainder will be covered by brown energy. The system reduces the amount of wasted green energy and increases the overall throughput of the data center.

2.3 Electricity Prices

2.3.1 Prediction

The Auto Regressive Integrated Moving Average (ARIMA) model is a time series model where the ARIMA processes analyse time series with a class of stochastic processes [3, 10]. The model has been applied to forecast of commodity prices such as oil, gas and electricity [10].

The success of the presented ARIMA model is dependent on the linear relationship of the underlying data generating process, whereas the Neural Network can handle non-linear relationships [8]. The Neural Networks are simple but a very powerful tool when it comes to forecasting, provided that the training set contains enough data and that enough computational resources are available. In [8] the Neural Network outperforms the ARIMA model in terms of both time consumption and accuracy of the predicted price as shown in 2.6. where the error percentage to the actual price is shown.

Comparative MAPE results between the neural network approach, the ARIMA technique and the naïve procedure

Market	Week	Neural networks	ARIMA	Naïve
Spanish	Winter	5.23%	6.32%	7.68%
	Spring	5.36%	6.36%	7.27%
	Summer	11.40%	13.39%	27.30%
	Fall	13.65%	13.78%	19.98%
Californian	Spring	3.09%	5.01%	6.98%

Figure 2.6: Comparison between Neural Network and ARIMA in terms of error [8]

The ability of Neural Networks to forecast is seen in [29]. The stock price movements have similarities with electricity prices in terms of its non-linearity and chaotic/dynamic nature. Investors must take into account these factors when handling time series that are non-stationary, noisy and have structural breaks. In addition macro-economical elements significantly influence stock prices, e.g the economy in general, politics, bank rates and expectations of investors could be examples of such influences.

The solution for stock price forecasting in [29] is a hybrid forecasting model called Wavelet De-Noising-based Back Propagation (WDBP) Neural Network. In brief, the network is based on a wavelet transformation function that analyses non-stationary characteristics of the time series by decomposing the original data. The function produces a signal that is further decomposed into a low-pass and high-pass filter for every neuron in the network. The low-pass filter reflects main characteristics of the signal, opposed to the high-pass filter that represents noise. The purpose of the decomposition is to separate stock price characteristics from noise so that prediction will have better chances of accuracy when all undesirable has been discarded. The signal without the noise is then propagated through the Neural Network using back propagation.

A day-ahead forecasting algorithm that predicts electricity prices in the market based on Neural Network (ANN) and Similar Days Method (SDM) is described in [18]. The purpose is to give close estimates for several days to come. The estimates can be used by electricity traders in their decision making but also by transmission companies for different purposes. The companies can use it for scheduling a short-term generator outage in order to predict where it is most inexpensive. It can also be used by actual producers of energy to strategically bid into the market to increase prices. The price estimate itself plays a huge role in decision making in all of these examples.

The combination of ANN and SDM is an attempt to simplify the ANN and make the prediction more accurate. The algorithm forecasts by using a ANN that modifies price curves obtained by averaging five similar price days corresponding to the forecast day, i.e. The ANN corrects the received output from the similar days approach [18]. In other words the technique takes into

consideration the influence of the most similar days and their price development in relation to the day we wish to forecast.

The ANN is trained with only 45 days from the day before the forecast and 45 days before and after the forecast day in the previous year [18]. Results can be seen in table 2.1

Year 2006 #1	ANN (Avg. MAPE [%]) #2	ANN (FMSE [$/MWh])
January 20	6.93	4.57
February 10	7.96	6.12
March 05	7.88	5.39
April 07	9.02	5.87

Table 2.1: Results of forecasting from [18].

An explanation for the choice of days is not discussed in the paper. As mentioned in [12] the more data the more complex problems can be handled

Artificial Neural Networks (ANN) has also been used for electricity price forecasting. In [27] they use an ANN to predict the half-hourly price of electricity of 24 hours. They differentiate between three different kinds of days: Normal trend price, Price with small spikes and price with large spike. They present a prediction for each of these days and present us to the mean absolute error and the root mean square error, which are standard measures for how accurately the prediction is done. The neural network is fed with 13 inputs as follows [27]:

- Day of week
- Time slot of Day
- Forecasted Demand
- Change in demand
- Price (one day ago) - 3 inputs
- Price (one week ago) - 3 inputs
- Price (two weeks ago) - 1 input
- Price (three weeks ago) - 1 input
- Price (four weeks ago) - 1 input

These inputs are fed into a 4-layer neural network: one input layer, two hidden layers and an output layer. The analysis of their ANNs shows that neural networks make a very precise prediction on normal trend price days but have difficulties forecasting the price with small and large spikes. They argue that if the reasons to the spikes in the price were taken into account as inputs in the network maybe the network would be better at forecasting the spikes prices. Also they argue that fuzzy logic, neural networks and dynamic clustering together will provide more efficient forecasting of the prices.

Since price forecasts in electricity markets are such a volatile operation because of the shifting tides, price demand, holidays etc. that affects the price it can be a cumbersome problem to model. In [4] he proposes a new method based on

neural networks and fuzzy logics to predict the electricity prices. He calls the new network approach a new fuzzy neural network. Fuzzy logic is basically a logic that has many values or many correct answers. Opposed to binary logic sets, where the answer can be only true or false, in fuzzy logic we have several grades of what we define as true, thus making it harder to decide what is really the truth but also makes us available to have a way larger scale of the data we are looking at.

The fuzzy logic is used within the nodes in the hidden layer to do evaluation of the data inputs. That is the activation function of the neurons in the hidden layer contains a fuzzification function that creates a square of the inputs compared to sinus activation functions that normally takes the sums of the inputs. This square over the inputs is used to classify the inputs into hyper cubes (input spaces) and then calculating how close they are together. Calculation of how far the inputs are from each other are used to calculate the output of the functions. By this we will get an upper and lower limit that the inputs can range between and thus we get a input that has the characteristics which turns out to give a better result than ARIMA, wavelet-ARIMA, MultiLayered ANNs and Radial Basis based ANNs. The data however is not optimized in this paper and they claim this to be future work. They also state that the better performance is based on a limited dataset.

Support vector machines (SVM) can be used for forecasting in the commodity market. In [35] they use SVMs for predicting the crude oil prices. SVMs are by nature a linear learning machine which means SVMs always use linear functions to solve the regression analysis but they can be expanded to be able to solve nonlinear problems. This is done by mapping the data into a high-dimensional feature space using a nonlinear mapping and then using the linear regression on this space. The SVMs undergo four different phases before being able to make the prediction. These can be seen in figure 2.7 Data sampling is done daily but due to inconsistencies in these data they adopt weekly and monthly data as alternatives. Data preprocessing is done by transforming the data appropriate for learning. This can be done by using logarithmic transformation. The training and learning step is used for determining the architecture and the parameters of the SVM. There is no criterion for deciding this other than just trail-and-error. Out-of-sample forecasting is done on new data and the prediction is made. As for evaluation they use the Root Mean Square Error (RMSE) to describe the estimates deviations from the real values. Their results and analysis shows that their SVM performs better than both ARIMA and Back Propagation Neural Networks (In most cases). They argue that SVMs gives better predictions than ARIMA and BPNNs in most cases but neural networks might perform better with data that is optimized for neural networks. Also the neural network outperforms the SVM in one of their sub-period comparisons.

SVMs have also been used for load forecasting of electricity demand. In [9] they use SVMs to make short-term predictions such as one-day ahead predictions. The goal of this study was to win a competition held by the EUNITE (EUropean Network on Intelligent TEchnologies for Smart Adaptive Systems).

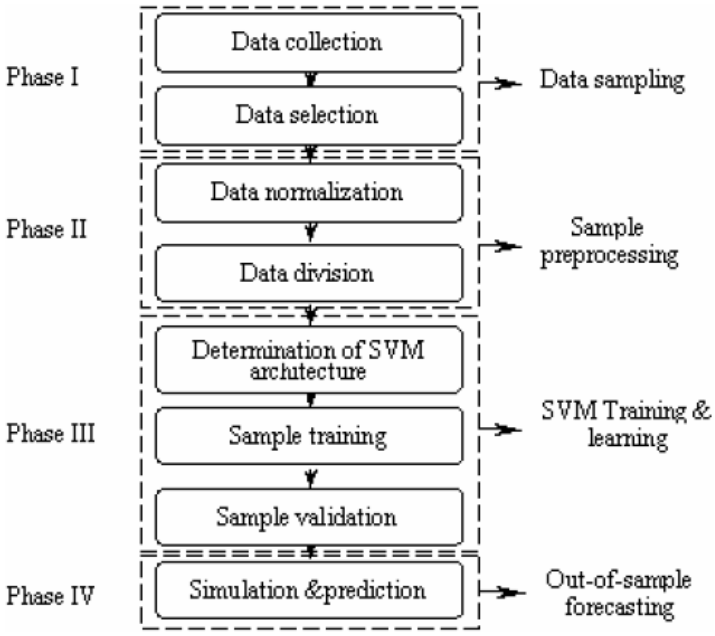

Figure 2.7: The steps taken to create a Support Vector Machine

This was the winning proposal in the competition. They first examined the data which was half-hourly load demand recorded from 1997 to 1998. They figured out that in the wintertime load demand was higher than in the summertime thus indicating a connection to weather data. The weekends could also be separated from the weekdays since the weekend load was lower than regular weekdays. When analysis was done they started setting up the model. First of all they prepare the data and select the data needed for the prediction. They select calendar attributes to map the holidays and which day it is to account for lower demands. Temperature is included in the vectors used for prediction of the electricity load demand and historical data is incorporated as well. The data segmentation in the steps of preparing a SVM allows them to take only a subset of the data since most of it can be generalized in the data analysis thus making it easier to do computations. They argue that their model for forecasting demand loads are the best possible model used in this competition and to give a more varied view on this they try out other methods among these Neural Networks. They configure a neural network that first performs work on the same data as their Support Vector Machine and it provides less than satisfactory feedback with an error margin of 6-8%. If they take the basic values used in the competition without doing any precomputation on it they receive a much better result of 3.64%. From this they conclude that their SVM is performing better than Neural Networks in the specific problem and that neural networks performance depends mainly on what data you use as input.

2.3.2 Applications

Prediction algorithms are not only used by electricity traders but can also be a part of various applications. In [14] they introduce an intelligent electricity broker (IEB) that is integrated into Smart Grids where it; 1) provides provision of en energy storage; 2) attempts to lower the electricity bill, and 3) optimally utilizes electricity during peak and low-peak energy production periods. The prediction algorithm is used by a decision algorithm to locate points in time where it is most feasible for the owner of the smart grid to either sell stored energy or buy energy if storage is low. It helps the system to lower the total amount of energy costs to the owner and also utilize the energy in the most intelligent way.

2.4 Electricity Demand Prediction

Weather conditions have a huge impact on the electricity industry in terms of network infrastructure and electricity consumption. In [16] they describe a multiple regression model that accurately predicts the monthly electricity demand based on weather and sociocultural conditions. The monthly electricity demand from this model shows a clear cyclic pattern which reflects the temperature changes during the year [16]. Besides weather conditions, social and economic factors also affect the monthly demand, e.g. the demand was decreased in Denmark during the financial crisis in 2009 [7].

Parametric multiple regression is preferred in [16] as opposed to a Neural Network used in this thesis. From a statistical error indices point of view the Artificial Neural Network (ANN) is a valid choice for prediction (see figure 2.8) in comparison with a plain socioeconomic model (SE) and a Box and Jenkins model (B&J). The argument for multiple regression is the simplicity in adjusting input values for each analysis of electricity demand in the prediction model and in the end they show similar results to ANN. To make a demand predic-

COMPARISON OF STATISTICAL ERROR INDICES

Error Type	ANN	S-E	B&J
ME	10026.79	25861.33	13764.58
MAE	13860.71	38012.25	27455.42
MSE	8.53×10^8	2.84×10^9	1.45×10^9
MAPE	4.77	14.29	10.98
SDE	9801.00	26878.72	19413.91
R^2	0.94	0.89	0.91

ME: mean error, MAE: mean absolute error, MSE: mean square error, MAPE: mean absolute percentage error, SDE: standard deviation of error, R^2: coefficient of determination.

Figure 2.8: Shows ANN compared to B&J and SE [16]

tion based on weather conditions a relationship between the two needs to be established. To get an understanding of this relationship, a plot of the average monthly demand as a function of Central England Temperature (CET) can be made [16]. The plot (see figure 2.9) shows an inverse relationship where it can be seen that a lower temperature in general results in an increased load consumption. Winter gives rise to lighting and heating load which is consistent with lower temperatures, and conversely in the summer with temperatures above 18 degrees the consumption tends to increase again due to the need for cooling and air-condition [16]. The nonlinear relationship between temperature

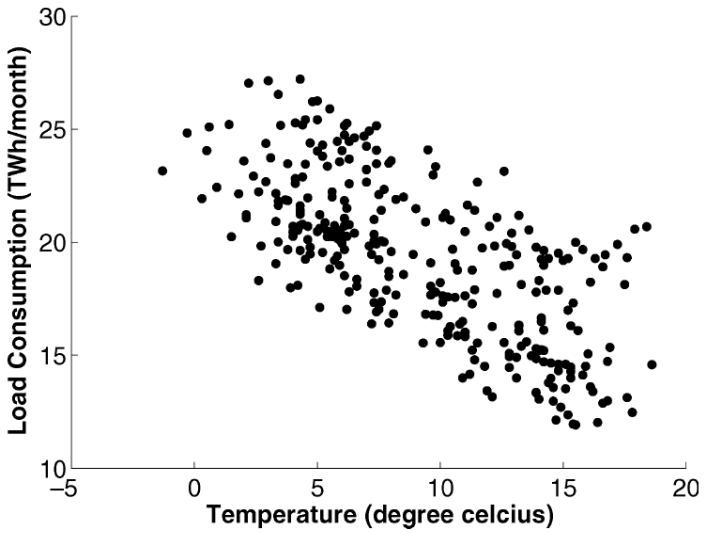

Figure 2.9: Function of monthly CET as a function from 1970 to 1995 [16]

and load consumption is turned into a concept called degree days. They introduce two categories of days; 1) Heating Degree Days (HDD) that is used to quantify when heating is required; 2) Cooling Degree Days (CDD) which is then used to quantify the need for cooling. The days are calculated from the CET data and give a more indicative picture than the temperature-load relationship [16]. A simple explanation of the calculation follows.

$$CDD = \sum_{d=1}^{N_d} (\gamma_d)(T_{dm} - T_{base_C})$$

where $T_{base_c} = 20^\circ$ is the base temp and T_{dm} is the mean daily temperature. $\gamma_d = 0$ if $T_{dm} - T_{base_c} < 0$ and $\gamma_d = 1$ if $T_{dm} - T_{base_c} > 0$. In other words if the temperature if above 20° the day can be characterized as CDD and there is a need for cooling.

$$HDD = \sum_{d=1}^{N_d} (1 - \gamma_d)(T_{base_H} - T_{dm})$$

where $T_{base_h} = 15.5^\circ$ is the base temp and again T_{dm} is the mean daily temperature. $\gamma_d = 1$ if $T_{base_h} - T_{dm} < 0$ and $\gamma_d = 0$ if $T_{base_h} - T_{dm} > 0$. This means that on a day with temperatures below 15.5 degrees there is a need for

heating. In both cases where $\gamma_d ! = 0$ a big difference has a greater impact on the demand.

Temperature is the most influential factor but other weather conditions can also be used when calculating the electricity demand, e.g. wind speed and rainfall can impact heating and lighting demand whereas direct sunshine can decrease the need for heating [16]. The model can be expressed by the relation between all factors by adding them together, e.g. the HDD value automatically increase the total electricity demand if it is not zero and this apply to all other factors in the model seen below.

$$\hat{\mathrm{E}}_A = \alpha_0 + \alpha_1 CDD + \alpha_2 HDD + \alpha_3 ELD + \alpha_4 V_w + \alpha_5 M_s + \alpha_6 M_r$$

Where a_n are constants, ELD is humidity, V_w is windfall, M_s sunshine and M_r rainfall. The calculation model can also include socio-economic factors. Specifically the population growth has a natural impact on the electricity demand over time. The more people the higher the demand gets. This would give rise to an additional factor in the model considering the growth. The prediction model from [16] comes within 3 percentage of the actual demand when it is run on data from 1996-2003 which is fairly close (2.10).

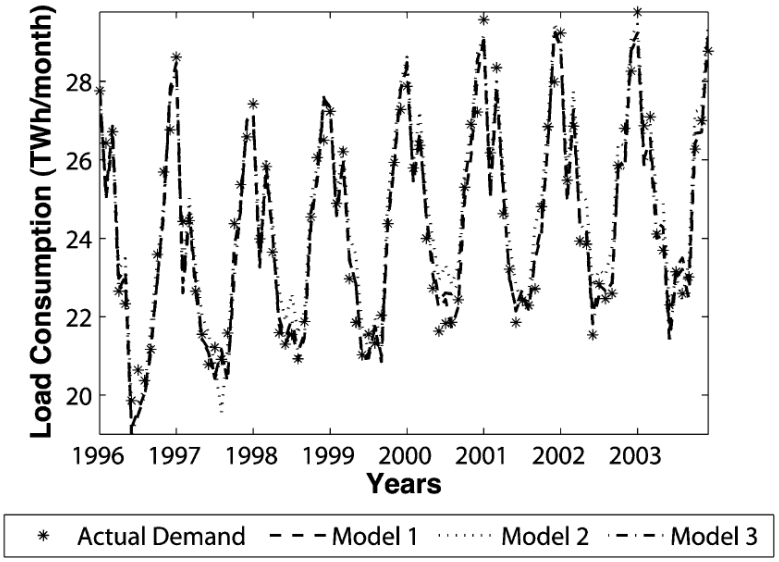

Figure 2.10: The actual values and the predicted demand in a comparison [16]

Chapter 3

Neural Network

This section is based on information from the book AI techniques for game programming[5] chapter 7, 8 & 9. Also from the book Neural networks: a systematic introduction[26] chapter 7.

Neural networks are models that imitate the brains behaviour. They have been created as an option to model artificial intelligence and analyse machine learning. The human brain is made up of billions of neurons that are interconnected in a big grid. They communicate by firing electrical shocks through the network of neurons. The human brain is extremely complex and can calculate vast amounts of data in no time. This is why scientist and mathematicians have been trying to emulate this behaviour to create artificial intelligence.

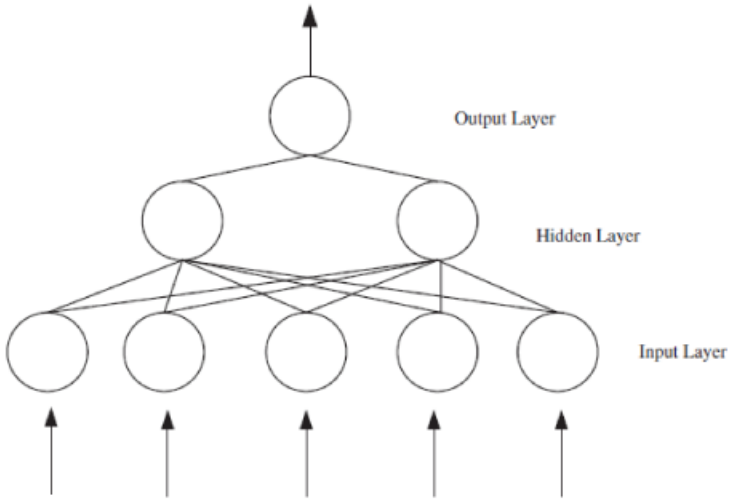

Figure 3.1: A simple neural network with 3 layers. [29]

Artificial neural networks (ANN) are artificial neurons (nodes) that are connected in a network. The network consists of an arbitrary number of layers that are interconnected. The most common structures in these networks are a feed forward structure. This kind of structure has the characteristic that it only flows data from the input layer through the layers to the output layer. There

are no loops in the network thus making it unable to reiterate any information. Normally all of the nodes in the input layer is connected to all of the nodes in the second layer. The same connections are done in the next layers until we hit the output layer. This will give us he sum of all the previous nodes(x_i) and their weights(w_i)($\sum_{i=0}^{i=n+1} w_i x_i$) in every node in the next layer: All of these

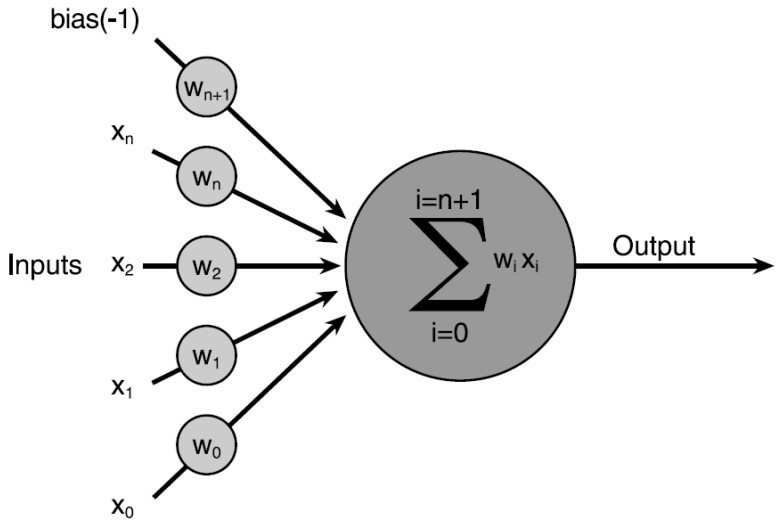

Figure 3.2: How the weight is calculated from one layer to the next

connections carry a weight that dictates how data flows through the network and reflects the relations between the inputs and the outputs of the network. The inputs of the network should be all the factors that has an influence on the output we want from the network. In our example we would want to input; weather data, temperature, demand, availability [22] to get the price as an output from the network.

Every node in our network contains an activation function. This function, when calculated, tells us whether the artificial neuron should fire or not. That is, if the neuron should transmit the data from the current layer to the next layer. There are a lot of different activation functions. The simplest form is the binary step function which either fires or it does not. This depends on the input and gives us a low threshold of complexity which is good in simple neural networks as the relations between the nodes do not need to be that fine grained. In more complex systems we want activation functions with a broader output range than binary. In many cases a sigmoid function is used as the activation function. This is because of the "S"-shape which enables it to compute outputs in a non-linear way. The non-linear nature of the sigmoid functions is what makes the neural network able to compute non-trivial problems in reasonable sized networks. To be able to calculate a non-trivial problem in these kinds of networks we need what is called a training algorithm. This algorithm depicts how the network evolves over time also known as learning. There are two kinds of learning; supervised learning and unsupervised learning.

3.1 Unsupervised learning

Unsupervised learning is when we do not have prior data to base our work on. Instead we have a problem where we want the neural network to try and estimate its behaviour relative to a specific task based on some assumptions we have about the system it performs on. It is commonly used with estimation problems like "Cluster Analysis", which in short is attempting to fit data into clusters of data that have some of the same criteria. This is often done by exploring the dataset and that is what unsupervised learning is good for. It also works with Artificial Intelligence(AI) that have to explore parts of the (virtual)world. In [5] he explains how an unsupervised learning feed-forward artificial neural network trains itself using a genetic algorithm to keep track of the fitness function of the AI. The fitness function is used to tell the AI if it is doing good or doing bad and is used for training the network. It will get a plus score in the fitness function if it encounters what we are looking for and get negative if it hits something that we defined as "wrong". Based on this fitness function it will update the weights of the neural network accordingly to what is most beneficial for the network as a total. After it has been allowed to do a lot of runs it begins to get a sense of what it is exploring and should be able to make better choices for each run.

3.2 Supervised learning

Supervised learning is an algorithm that uses a dataset that contains both the inputs and what the output is expected to be. This dataset is used to train the neural network to make it able to do calculations on data and predict the outcome. An example of an algorithm used for supervised learning is the backpropagation algorithm. It starts out by randomly assigning all the weights on the connections between the neurons. It then calculates the output of the network and compares it to the expected output. From that it calculates the error margin between the expected and the calculated output and adjusts the weights accordingly. This is done for all the hidden layers as well until we hit the input layer. All of these steps are called an epoch. We will repeat as many epochs as we need until the sum of all the errors are within a given threshold. The name of the algorithm originated from this approach where it propagates the error backwards in the network.

Supervised learning can be thought of as learning with a teacher. As an example we can use the XOR table:

In this dataset we have both the inputs and we are given the output that we expect. This gives the backpropagation algorithm a direction to follow when minimizing the error function of the network. We can do this because we get an output that we can compare to the expected output and that allows us to predict the direction that the error correction should take to come closer to the answer. Also the sigmoid activation function helps us closing in on the target output since the sigmoid function always has a positive derivative which ensures that we will always have a direction to follow[26, p. 153].

Input #1	Input #2	Output
0	0	0
1	0	1
0	1	1
1	1	0

Table 3.1: This training set is very simple yet it illustrates a training set for a supervised learning algorithm very well.

In neural networks bias neurons are often added to the layers to help them learn patterns. The bias neurons are added to give the activation functions the ability to change its output even if x is zero. If we look at the graph in figure 3.3 it uses the following activation function: $\frac{1}{(1+e^{(-cx)})}$ where c is the weight.

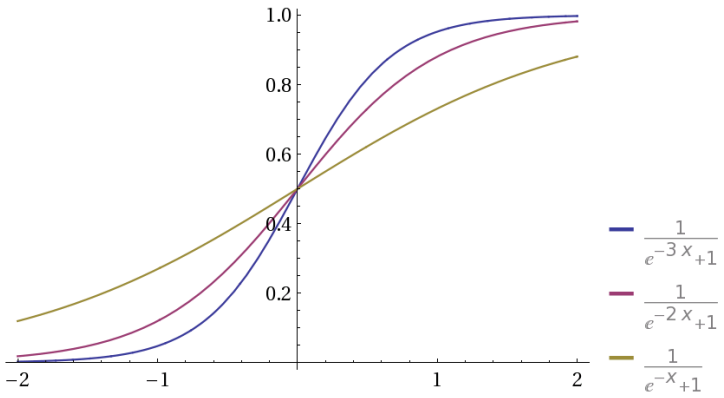

Figure 3.3

The figure shows how the gradient of the function alters with different weight values. Even though the gradient of the function is clearly altered by the weights the function is still outputting the same result for zero thus we cannot alter the output for x equal to zero just by altering the weights. This is what we use the bias for. If we apply a bias of one to all of the neurons we will be able to shift it either to the left or the right. In figure 3.4 we see the same function as before where the weight is set to 2. The difference is that we added a bias (b) to this function: $\frac{1}{(1+e^{(-2x+b)})}$ [26, p. 165] [31]

To calculate the error between input and output we use the mean squared error function. This function is given by: $MSE = \frac{1}{n}\sum_{i=0}^{n}(\hat{y}_i - \bar{y}_i)^2$ where $\hat{y}_i$ is the ideal value and $\bar{y}_i$ is the actual value. We use this error calculation because it incorporates the bias [33]

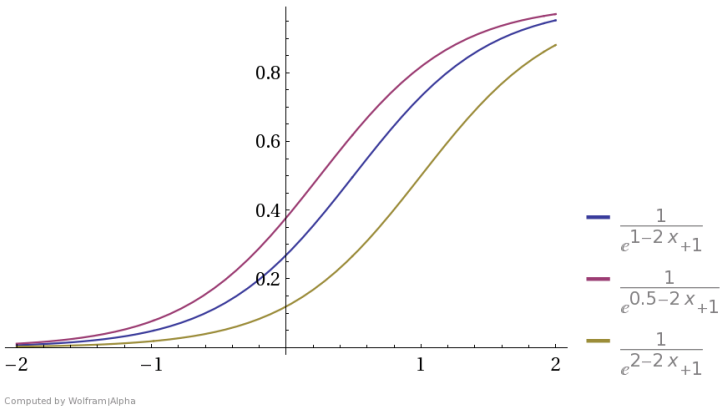

Figure 3.4

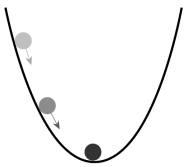

Figure 3.5: At the global minimum [5, P. 318]

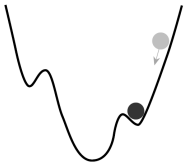

Figure 3.6: Stuck in a local minimum [5, P. 318]

3.3 Problems to be avoided

When we are trying to fit our algorithm and make it recognize patterns we will encounter several possible pitfalls. First of all there is the chance of ending up in a local minima. This is when the the backpropagation algorithm attempts to find the global minimum of the error curve, thus having reduce the error as much as possible. The algorithm works by trying to reduce this error margin a little step at a time. If it encounters a local minima on the curve and thinks it has reached the global minima it gets stuck and we will get inaccurate results.

To avoid the backpropagation algorithm to falsely accept a local minima as the global minima we can give the algorithm momentum. This is done by adding a bit of the last error correction from the earlier layer to the next layers error correction. This way the algorithm, so to say, will scoot right by any small deviations in the error correction face.

Another pitfall when working with neural networks is over fitting the algorithm. This is when the algorithm instead of finding a generalized pattern in the inputs it will find an over-fit pattern that will fit exactly that input. This is better shown in figure 3.7. This can be avoided by some simple techniques. First of all we want to reduced the neurons as much as possible as long as it does not interfere with our performance of the system. This is a trial and error problem and has to be tweaked along with evolving your neural network. We can add noise to avoid this problem. By adding noise(random data values) we prevent the algorithm from fitting the function to closely to the given data.

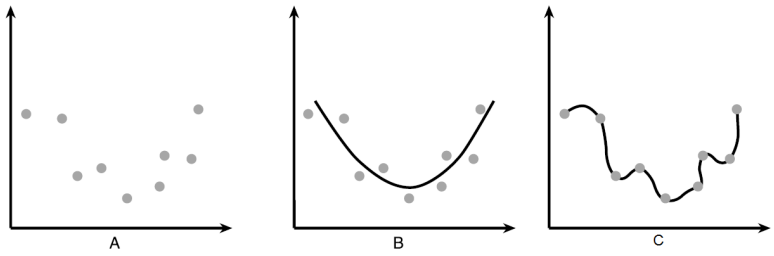

Figure 3.7: A. The plot graph of the input B. The generalized function C. An over-fit function

Thus giving us a more generalized function where it hopefully will be able to fit new data presented to it better. Early stopping is another method to avoid over-fitting. This is only doable with large datasets where you can split it into two equal datasets. The first will work as a training set and the second will work as a validation set. We will keep training the dataset and checking with the validation set until the difference between those two start to increase.

Chapter 4

Conclusion

►. . . ◄

Bibliography

[1] S. K Aggarwal et al. Electricity price forecasting in deregulated markets a review and evaluation. *National Institute of Technology*.

[2] Baris Aksanli et al. Utilizing green energy prediction to schedule mixed batch and service jobs in data centers. *Computer Science and Engeneering Department, University of California, San Diego*.

[3] Hemmati Amjady, Nima. Meisam. Energy price forecasting. *IEE power and energy magazine*, 2006.

[4] Nima Amjady. Day-ahead price forecasting of electricity markets by a new fuzzy neural network. *Power Systems, IEEE Transactions on*, 21(2):887–896, 2006.

[5] M. Buckland. *AI techniques for game programming*. Course Technology, 2002.

[6] D. W Bunn. Forecasting loads and prices in competitive power markets. *Invited Paper*, 1999.

[7] business.com. Finanskrisen saetter elforbrug i bakgear.

[8] JPS Catalao. Short-term electricity prices forecasting in a competitive market: A neural network approach. *Electric Power Systems Research*, 2007.

[9] Bo-Juen Chen, Ming-Wei Chang, et al. Load forecasting using support vector machines: A study on eunite competition 2001. *Power Systems, IEEE Transactions on*, 19(4):1821–1830, 2004.

[10] Francisco. Conejo Anotiono Contreas, Javier. Nogales J. Arima models to predict next-day electricity prices. *IEE TRANSACTIONS ON POWER SYSTEMS, VOL. 18, NO. 3*, 2003.

[11] DEA. Danish energy authority report, 2012.

[12] Pedro Domingos. A few useful things to know about machine learning. *Commnucations of the acm*, 2012.

[13] Zeno Farkas. Considering air density in wind power production. 2011.

[14] Jesper Grode et al. Intelligent electricity broker. *ICEAS 2012*, 2012.

[15] M. T Hagan et al. Training feedforward networks with the marquardt algorithm. *IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 5 , NO. 6*, 1994.

[16] Ching-Lai Hor et al. Analyzing the impact of weather variables on monthly electricity demand. *IEE TRANSACTIONS ON POWER SYSTEMS, VOL. 20, NO. 4*, 2005.

[17] E. Mabel, M. Carolin. Fernandez. Analysis of wind power generation and prediction using ann: A case study. *Renewable Energy 33, 986-992*, 2008.

[18] Senjyu. Naomitsu Urasaki Mandal, Paras. Tomonobu. A novel approach to forecast electricity price for pjm using neural network and similar days method. *IEE TRANSACTIONS ON POWER SYSTEMS, VOL. 22, NO. 4*, 2007.

[19] Maurizio Motta et al. Measure-correlate-predic methods: Case studies and software implementation. *EMD Internation A/S (EMD).*

[20] Poul Erik Munksgaard, Jesper. Morthorst. Wind power in the danish liberalized power market - policy measures, price impact and investor incentives. *Energy Policy 36, 3940-3947*, 2008.

[21] S Nissen. Large scale reinforcement learning using q-sarsa and cascading neural networks, 2007.

[22] Javier Nogales, J. Francisco. Contreras. Forecasting next-day electricity prices by time series models. *IEE TRANSACTIONS ON POWER SYSTEMS, VOL. 17, NO. 2*, 2002.

[23] P Pujol. The solution of nonlinear inverse problems and the levenberg-marquardt method. *University of Memphis, Department of Earth Sciences*, 2007.

[24] A Ranganathan. The levenberg-marquardt algorithm 8th, 2004.

[25] M. Riedmiller and H Braun. A direct adaptive method for faster backpropagation learning: The rprop algorithm. *University of Karlsruhe.Institut for logic, complexity and deduction systems.*

[26] R. Rojas. *Neural networks: a systematic introduction.* Springer, 1996.

[27] Deepak Singhal and KS Swarup. Electricity price forecasting using artificial neural networks. *International Journal of Electrical Power & Energy Systems*, 33(3):550–555, 2011.

[28] T Strutz. Data fitting and uncertainty: A practical introduction to weighted least squares and beyond. *Vieweg+Teubner Verlag*, 2011.

[29] Jian-Zhou Wang et al. Forecasting stock indices with back propagation neural network. *Expert Systems with Applications 38*, 2011.

[30] wiki. Backpropagation.

[31] wiki. Inductive bias.

[32] wiki. Machine learning.

[33] wiki. Mean squared error.

[34] wiki. Resilient backpropagation.

[35] Wen Xie, Lean Yu, Shanying Xu, and Shouyang Wang. A new method for crude oil price forecasting based on support vector machines. *Computational Science–ICCS 2006*, pages 444–451, 2006.