TILBURG UNIVERSITY

MASTER THESIS

Forecasting short term load with machine learning

Author:
Loes VAN DER POEL

Supervisor:
Dr. Denis KOJEVNIKOV

A thesis submitted in fulfillment of the requirements
for the degree of Msc. of Econometrics and Economical Mathematics

in the

Department of Econometrics & Operations Research

June 16, 2022

TILBURG UNIVERSITY

Abstract

Tilburg School of Economic and Management
Department of Econometrics & Operations Research

Msc. of Econometrics and Economical Mathematics

Forecasting short term load with machine learning

by Loes VAN DER POEL

Accurate electricity consumption prediction has primary importance in the field of energy planning. It is a key research area for efficient power grid operation which has become more challenging due to rapid developments in the applications sustainable energy sources. This thesis aims to study and compare a variety of short-term load forecasting methods in an attempt to propose a solution which provides the most accurate forecast output for a typical data set. Three models are proposed, their performance is measured against a benchmark model that bases its predictions on naïve persistence. The first model makes use of the Box-Jenkins ARIMA modelling technique. The last two models are respectively a simple feed-forward neural network and a more sophisticated long short term memory (LSTM) neural network. The comparison revealed that the performance of each model depends heavily on the time gap between the available data and the prediction. As this gap grows larger, additional complexity in the model choice results in more accuracy in the prediction, this is reflected by the LSTM model outperforming the other two models.

Contents

List of Figures

List of Tables

List of Abbreviations

TSO	Transmission System Operator
BRP	Balance Responsible Parties
PF	Profile Fraction
SJV	Standardized Yearly Volume
MCF	Measure Correction Factor
STLF	Short Term Load Forecasting
EDA	Exploratory Data Analysis
ANN	Artificial Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short Term Memory
MAE	Mean Absolute Error
MSE	Mean Squared Error
MAPE	Mean Absolute Percentage Error
RMSE	Root Mean Squared Error
ACF	AutoCorrelation Function
PACF	Partial Autocorrelation Function
ARMA	Autoregressive moving average
ARIMA	Autoregressive integrated moving average
ARIMAX	Autoregressive moving average with exogenous input
BPTT	Backpropagation through time
SGD	Stochastic Gradient Descent

Chapter 1

Introduction

Electric load forecasting is commonly deployed by companies to determine the amount of electricity that needs to be generated or purchased in order to closely match demand. Obtaining precise predictions will enable such companies to make more informed and better decisions with regards to their energy purchase management. On a larger scale, accurate forecasts are vital for maintaining stable and efficient energy supply.

Europe's energy sector is changing fast. In the past, large centralised power stations provided the majority of Europe's energy needs. However Europe's energy market has experienced a vast extent of liberalisation, this has been an important starting point in the design of the balancing system in the Netherlands (Tennet, 2022). Furthermore renewable electricity growth is accelerating faster than ever. Besides all the positive effects, renewable energy sources have been adding a lot of variability to the behaviour of energy demand. These facts make the ability to produce a qualitative forecast all the more valuable.

1.1 Balancing the grid

"The high-voltage grid is the backbone of the electricity supply system, it connects generators of electricity to consumers." (Tennet, 2022) The high-voltage grid in the Netherlands has a frequency of 50 Hertz, which is in accordance with European standards (WorldStandards, 2021), in the US the system frequency equals 60 Hertz. The power system operator faces the challenge of keeping this level of frequency constant at all times. In the Netherlands this power system operator is TenneT, they are responsible for preserving balance between demand and supply. At 50 Hertz, and within two hundred milihertz on either side of 50 Hz, the power system safely and securely transmits power from the suppliers to the demand side. Changes in supply and demand for electricity can have a major effect on the frequency of the grid. For instance, if there is a demand surplus the frequency will fall and vice versa. All electrical devices and equipment in Europe are designed to operate at 50 Hz, a deviation of more than two hundred miliherz will cause electrical infrastructure and equipment to fail which could eventually lead to a blackout. This almost happened on January 8th 2021, supposedly when the frequency dropped to 49.74 due to a power supply failure in eastern Europe. The sudden drop in frequency nearly caused a Europe-wide blackout, pumped storage power plants and some gas-fired power plants still available had to be mobilised in order to prevent such a black-out. Still large electricity customers in France had to be disconnected from the grid.

"The statutory role of maintaining the balance between the injection of elictricity to- and withdrawal of electricity from the electricity grid, has been assigned to TenneT TSO B.V. in the 1998 Electricity Act (article 16). TSO is an abbreviation for

Transmission System Operator of the national electricity or gas grid, TenneT is the designated TSO for electricity in the Netherlands and large parts of Germany." (Tennet, 2022) All connections to the grid are allocated to so-called balance responsible parties (BRP), VanHelder is one of those BRP's. "BRP's are financially responsible for keeping their own position (the sum of their injections, withdrawals and trades) balanced over a given timeframe." (*Benchmark of markets and regulations for electricity, gas and heat and overview of flexibility services to the electricity grid* 2019) In order to supervise this process, law obliges each BRP to provide TenneT with a commercial hourly trade schedule on a daily basis. When the quantities estimated by the BRP's differ from the amount that is actually produced or consumed, TenneT has built in safety measures in place that will (automatically) restore the balance on the grid. When the frequency exceeds the upper statutory limit, generator output can be adjusted. When frequency threatens to dive below the lower threshold, load will be disconnected from the grid. This is called automatic frequency restoration reserve, when the grid imbalance has not been resolved after 12.5 minutes, manual frequency restoration reserves can support or replace the automatic measures.

These balancing measures have financial implications for the BRP's. When the demand is underestimated, resulting in a deficit of electricity, TenneT will be forced to purchase potentially expensive peaking power to supplement the shortage of electricity at a price much higher than the market price (Foster, 2020). When the demand is overestimated, this results in a production surplus which leads to unnecessary expenditures. Each BRP is financially responsible for its imbalance and pays or receives the imbalance price for the relevant imbalance settlement period. The imbalance is calculated as the difference between the predicted amounts and the actual electricity usage, the costs are then allocated proportionally to the individual imbalances of the BRP's. Systematic mistakes in predictions made by all BRP's are thus especially costly because they cannot be diversified.

1.2 Measure Correction Factor

A multitude of elements factor in to an accurate prediction of energy consumption. One of the main aspects is knowing the types of consumers that need to be served. As can be imagined electricity consumption patterns of a regular household will vastly differ from those of large companies. Various types of consumers can be identified by consumer profiles, there are 10 consumer profile categories characterized by properties such as their voltage level and whether the connections are remotely readable or not. A specification of the 10 consumer profile categories and their characteristics will be provided in the appendix (Appendix A). The consumption of industrial consumers higher up in the profile categories (E3A and above) can be determined on a daily basis, this segment of the market is therefore generally referred to as telemetry. This thesis however, will be focusing on predicting electricity consumption for the smaller consumer section (E2B and below), which includes both households and smaller businesses. Smart meters are a rising trend, supported by network operators as well as the government. They allow frequent communication of information to electricity suppliers. However, according the central bureau of statistics (CBS), 21 percent of the Dutch households, still only has a traditional meter installed. For this group, suppliers have to rely on periodic meter readings for feedback.

NEDU (Vereniging Nederlandse Energie Data Uitwisseling) issues profiled fractions for each consumer profile category on an annual basis. These load profiles will specify for every 15 minutes of every day in a year the expected fraction of the annual consumption. The sum of those $4 \times 24 \times 365 = 35.040$ profile fractions (PF) will add up to 1. This method takes into account the following determinants: the season, time, national holidays and daylight saving time. The profiled fractions provide the BRPs with a fundamental idea of the energy allocation over the year. Supplier obligations are settled on a subhourly basis, whereas the actual consumption is measured only periodically based on meter reading schedules. The load profiles can be used to convert this periodic data into estimates of subhourly consumption to determine the supplier obligation. The estimates resulting from multiplying a load profile with the expected annual consumption (SJV) (based on the meter readings) can be aggregated for all consumers of an energy supplier. This aggregated amount can serve as a starting point for further calculations to determine the total demand of their customer base, that the BRP needs to communicate to TenneT

The ratio between the actual energy consumption and the calculated 15-minute volume ($PF \times SJV$) is the measure correction factor (MCF) for that 15-minute time interval, its formula is given by (1.1). The Netherlands are divided in grids and a MCF is calculated for every grid separately.

$$MCF = \frac{\text{Real Volume}}{\text{Total profiled volumes}} \tag{1.1}$$

Every supplier will be allocated a part of the actual volume that is calculated in the following manner: $MCF \times \sum_{i=1} PF_i \times SJV_i$, where i represents each connection of the supplier in that grid. The difference between the allocated volume and the volume that this BRP has predicted and communicated will be settled at the imbalance price.

The imbalance costs are minimized when the BRP accurately predicts the MCF for each period and each grid that they serve. Therefore the main goal of this thesis will be to evaluate a variety of methods that can be employed for forecasting the MCF. This will be a short term load forecast problem.

1.3 Renewable energy sources

The use of renewable energy, as a replacement or supplement of more traditional forms of energy, has been massively increasing over the past decades. This is largely due to the growth in electricity generation from renewable sources, adding up to an increase of almost 7% in 2020. From this renewable growth, solar PV and wind are the two major contributors accounting for approximately two-thirds of the growth. This transition to renewable energy sources has been heavily promoted by the European Union, who have targeted for the EU to have 20% of its final energy consumption provided by renewable sources by 2020 (Council of European Union, 2018). For the Netherlands specifically, a national overall target for the share of energy from renewable sources in gross final consumption of energy in 2020 was set to be 14%, whereas the initial share of energy from renewable sources amounted up to 2.4% in 2005. In 2020, renewable energy accounted for 11.1% of total Dutch energy consumption, this is significant leap from the 8.8% in 2019 (Statistics, 2021). In order to achieve the target set by the EU, the Netherlands entered into an agreement to buy 8

to 16 TWh of renewable energy from Denmark, this is called a statistical transfer. The next step on the energy agenda is for the Netherlands to reach a percentage of 16% sustainable energy by 2023, aiming for 100% sustainable energy in 2050 (Council of European Union, 2018).

The Dutch government offers a number of incentives for households and companies to invest in sustainable energy. For instance: companies that either produce renewable energy or apply CO_2 reducing techniques are eligible for the subsidy sustainable energy production and climate transition (SDE). This subsidy ranges from 60 to 300 euros per tonne CO_2 reduced, in 2021 this subsidy had a budget of 5 billion euros available. Another incentive provided by the government is the Energy Investment Allowment (IEA) for sustainable investments, which allows companies to deduct 45.5% of the investment costs from their taxable profit. The budget available for this program in 2022 adds up to 149 million. Lastly, a regulation that benefits small companies as well as households is the net metering arrangement. The net metering arrangement guarantees that all power that a household or small business feeds back into the grid will be deducted from the amount of power it consumes. This way they will pay only for the resulting net balance of the two. Theoretically this means that a household could reduce its costs to zero, but not below that. The net metering arrangement has been the main reason that solar panels have become a very popular investment for households and small businesses. However this arrangement is set to change from 2023 onwards, the government will phasing out of the arrangement by decreasing the amount of energy covered by the net metering law by 9% per year.

Renewable energy, also referred to as clean energy, is energy that comes from natural sources or processes that are constantly replenished. While these sources are inexhaustible in the long run, they are limited in the amount of energy they can generate per unit of time. Generation depends on natural processes such as irradiation for solar energy and wind speed for wind energy. Further research even suggests that these dependencies between weather variables and the amount of renewable energy generated are proven to be quite complex. For instance: solar cell performance does not only vary with radiation but is also observed to decrease in temperature (Dubey, Sarvaiya, and Seshadri, 2013). Furthermore, wind power seems to be affected by both wind velocity and air-density in a non-linear manner (Şen, 2013). Additionally, these new energy sources are not located centrally, as used to be the case for more traditional sources of electric power. Wind farms are often established off-shore, and many small scale wind farms exist throughout the Netherlands. Solar panels can be installed locally at a consumers residency. Generation of electricity from such local resources is unobserved by system operators. Adding to this complexity of dependency and unobservability the fact that predicting weather variables in and of itself is a complicated task, it can be concluded that renewable energy adds a lot of unpredictable variability to the energy demand.

Fluctuation of energy production from renewable sources proposes a significant challenge for energy suppliers, whereas fossil fuels provide a relatively constant supply of energy. This is also the main reason that arrangements such as net metering are necessary. Solar panels will generate a lot of energy at times when it is generally less needed resulting in overproduction, overproduction will be fed back into the grid. However, because the demand for energy at those times is really low the price for electricity will be low as well. At night a solar panel will logically not generate any electricity, energy has to be bought from the grid at a much higher price than

what the overproduction earlier that day was sold at. The difference is now covered by the government. A great solution for this problem would be to store energy overproduction, so that it can be used to balance shortages at a later time. However technology has not yet found a way to implement this on a large scale.

Altogether it can be concluded that the surge of renewable energy is a trend of increasing relevance. As long as energy can not be stored, it will add a lot of uncertainty to energy demand forecasts.

1.4 Goal and structure of this thesis

This thesis will provide an extensive comparison on different forecasting methods. Furthermore this research will also take into account the effect of predicting a number of periods ahead and thus dealing with a delay of the available historical data. The first part of this thesis will focus on the theoretical background. Chapter 2 will feature an in-depth literature review on the topic of forecasting time series with specific attention for predicting short term load forecasting. The following chapter, chapter 3, will handle the theory of classical time series forecasting models, explicitly focusing on ARIMA models and their extensions. Finally, chapter 4 will conclude the theoretical part of this study with a thorough exploration on the theory behind neural networks and some of their extensions, that are specifically tailored to time series forecasting. Chapter 5 dives into the data, it encompasses amongst others an exploratory data analysis and a section on data preparation. The subsequent chapter discusses the methodology that was applied in order to obtain and compare the results presented in chapter 7. Lastly, the thesis is completed with a conclusion and a brief discussion.

Chapter 2

Literature Review

This chapter reviews the literature that has been published on the topic of time series forecasting and the performance of classical time-series methods versus that of neural networks. The first two sections will focus on the academic history of statistical methods and neural networks respectively. The third section will present an extensive summary of related literature on comparing the performance of neural network against that of traditional models, specifically ARIMA. The last section will review the literature on forecasting short term load.

2.1 Statistical time series methods

Time series forecasting is a historically well-studied topic in literature. Theoretical developments in statistical analysis of time series data started as early as 1927 (Yule, 1927). It was during this time that the principle of a autoregressive moving average model was first introduced by Herman Wold (Wold, 1938). These models were suitable to mitigate the problem of periodic fluctuations. However, Wold was not able to define the likelihood function for this model that was necessary to estimate its parameters. This limitation was only supplemented years later by the publication of the book "Time Series Analysis" by G. E. P. Box and G. M. Jenkins (G. Box, 1970). Their approach rapidly became widely acclaimed by academics and practitioners. The appraisal was amplified by a study conducted a few years later which provided arguments in favour of the application of a time series approach as opposed to more complicated econometric models for short term predictions. The research proved that predictions generated by time series techniques were substantially more accurate (Granger and Newbold, 1975). Nowadays, the so-called Box-Jenkins models are possibly the most commonly applied models in time series analysis, and many techniques used for forecasting find their origin in these models.

A critical note is that most classical time series models are linear, whereas real world time series often exhibit non-linear behaviour (Kantz and Schreiber, 2004). This means that these processes would be better described by non-linear models.

2.2 Machine learning methods

The very first idea of a computational neuron was proposed in 1944 by Warren McCullough and Walter Pitts (Fitch, 1944). The neuron consisted of two parts, the first part would take the inputs of the model and perform an aggregation on those inputs. The second part would make a decision based on the aggregated value produced by

the first part. The paper initially received little attention until its ideas were applied by mathematicians such as John von Neumann and Norbert Wiener. The advancement of computers in the 1950s made it possible to simulate neural networks. In 1957 American psychologist Frank Rosenblatt, proposes the first neural network consisting of one single layer (Rosenblatt, 1957). Rosenblatt's perceptron was an improvement of the McCulloch-Pitts neuron, it dealt with some of its major flaws. Rosenblatt's perceptron could assign different weights to each input automatically and could also automatically compute the threshold for the decision making part of the neuron. Additionally the single layer in Rosenblatt's perceptron can process non-boolean inputs. In 1969, (Minsky and Papert, 1969), discussed the bottleneck of a Single-Layer perceptron model; it always implied a linearly separable underlying distribution of the data. This was cause for a stagnation in research on neural network while popularity of other linear machine learning methods prevailed.

In 1974 social scientist and machine learning pioneer Paul Werbos published his dissertation, he was the first to describe the process of training neural networks through backpropagation of errors (Werbos, 1974). Backpropagation is currently the most widely applied tool in the field of artificial neural networks. The discovery of the backpropagation method inspired the proposition of a multilayer neural network (MLP) that avoids the pitfalls raised by Minsky and Papert. (Rumelhart, Hinton, and Williams, 1986) were the first to successfully train a shallow network, their results and training algorithms are published in "Learning Internal Representations by Error Propagation". The successful implementation of a multi-layered model caused a renewed interest in neural networks.

In recent years, machine learning has made important advances and neural networks are recognized as one of the best options for solving problems in many different fields and applications. Facial recognition, stock market prediction and handwriting analysis are only a sample of the many possible implementations of neural networks. "NNs are recognized for their ability to capture subtle relationships in the data, even if they are unknown or hard to describe." (Zemouri and Zerhouni, 2012) NNs make few a priori assumptions about the functional form of the forecasting model, whereas most traditional models have at least a certain extent of reliance on a functional specification (Park et al., 1991).

However, NNs also have some indisputable disadvantages. NNs are often referred to as black-box models because they lack the ability to provide insight on the structure of the function being approximated (Karakurt, 2021). Therefore, it is impossible to derive how a certain output is determined. This problem is specific to NNs whilst other models such as regressions or tree classifiers have a clear representation of the way their results are achieved. Another drawback is that Neural Networks require large amounts of data in order to achieve successful results. The amount of data necessary depends on a number of factors, one of the most dominant factors being the number of parameters of the model. A rule of thumb is that the amount of data needs to be at least 10 times the degrees of freedom of the model (Azoff, 1994). Thus when there is a limited amount of data, there might be restrictions regarding the model choices of the NN.

Recurrent Neural Networks (RNN) (Rumelhart, Hinton, and Williams, 1986) are a derivation of a simple ANN. The focal distinction between them is the fact that

a RNN is designed to process sequential data such as time series data. ANNs assume independence among the input variables, the natural temporal ordering of time series data explicitly violates this assumption. RNNs essentially contain feedback loops that allow information to persist. Previous information can be stored and used as part of the input for the next step. RNN's have been succesfully employed for time series forecasting in various disciplines.

The fundamental weakness of basic RNNs, is that they experience difficulties with handling long-term dependencies in data. (Hochreiter and Schmidhuber, 1997) introduced the Long Short Term Memory (LSTM) model that was capable of dealing with long-term dependencies. The tailored model architecture permits LSTM to bridge larger time lags between relevant inputs, while traditional RNNs already fail to learn when the number of lags exceeds 10.

2.3 Comparing model performance

An extensive research on forecasting with neural networks was conducted by (Zhang, Patuwo, and Hu, 1998). They start of with the presentation of some unique characteristics of neural networks. "ANNs are data-driven self- adaptive methods in that there are few a priori assumptions about the models for problems under study. They learn from examples and capture subtle relationships among the data even if the underlying relationships are unknown or hard to describe." This specific quality makes artificial neural networks an attractive alternative for more traditional forecasting methods. Its performance is however not guaranteed to exceed that of other time series models. Zhang et al. considered a substantial amount of research and found results to be inconclusive as to whether ANNs outperform classical methods.

The comparison of neural networks to traditional methods has been a topic of interest since the rise of Neural nets as a forecasting mechanism. Most of the early and most prominent papers examining the performance of NNs and time series method used data from the M-competition. This competition used 1001 real world time series of various sources and challenged experts to submit forecasts on this data. The results from this very first forecasting competition were later summarized in a paper (Makridakis et al., 1982). They found that considerable gains in accuracy could be induced by discrimination in choice of method depending on several factors such as the frequency of the data, the horizon and the type of series (micro or macro data etc.). Additionally, it appeared that combining multiple forecasting methods significantly improved the accuracy of the forecast as opposed to employing one single forecast method. They concluded with the finding that that statistically sophisticated methods do not outperform simple methods when there is a considerable amount of randomness in the data. This competition has been repeated several times over the last three decades with varying sets of data.

The results of this competition have been replicated and verified with subsets of the M-competition dataset. (Sharda and Patil, 1990) trained an artificial neural network with one hidden layer on two years of historical data for 75 datasets of varying frequency (yearly, quarterly, monthly). They compared its performance to a Box-Jenkins model with default settings, evaluation was done using the MAPE. They showed that there was no significant difference between the forecasting accuracy of both models. They concluded by stating that these results indicate potential

for the application of neural networks as a forecasting tool. Improvement of performance could be achieved by a more sophisticated model architecture or training algorithm. Later they found that Neural Networks outperformed Box-Jenkins models for time series with short memory (Sharda and Patil, 1992). They additionally emphasized the impact of the architecture and training method on the performance of neural networks.

NNs were found to outperform ARIMA for quarterly and monthly data by (Chakraborty et al., 1992), this result is confirmed by a large number of papers investigating the performance on time series from a variety of economic areas. Equivalent but not superior performance was registered for daily data by (Caire, Hatabian, and Muller, 1992). (Palomares-Salas et al., 2009) compare an ARIMA model to a NN for data on wind speed measured at a very high frequency and for a selection of horizons. They deduce that the ARIMA model outperforms the neural networks. Furthermore, some studies argue that NNs generate superior results in forecasting compared to traditional models when the underlying data is nonlinear (Hwang and Basawa, 2001). Conversely ARIMA models closely match the performance of NNs when the data is linear (Zhang, Patuwo, and Hu, 2001). However, there is also evidence of the contrary provided by (Faraway and Chatfield, 1998)

A performance comparison between LSTM and ARIMA was conducted by (Siami-Namini and Namin, 2018), who applied both models on a monthly financial dataset from January 1985 to August 2018. They reported an improvement of 85% by applying the LSTM model compared to the ARIMA model. (Hu et al., 2018) provide a quantitative analysis regarding the performance of ANN and LSTM model on rainfall-runoff modelling. The LSTM model delivers a more stable and significantly more accurate result than the ANN model.

2.4 Literature on Short Term Load Forecasting

Within the realm of forecasting problems, three separate categories can be distinguished by the time horizon of interest. Load forecasts can be short-term, medium-term (MTLF) or long-term (LTLF) (Mocanu et al., 2016). Short term load forecasts (STLF) predict system load over an interval ranging from one hour to one week ahead (Contreras and Santos, 2006), it is commonly used for planning and managing power balance. Medium term load forecasts (MTLF) covers a timespan up to one year ahead, it is suitable for maintenance scheduling and to plan for outages and major works in the power system (Zhang, 2014). Long term load forecast (LTLF) considers a time period from 1 up to 10 years ahead, it is often applied for similar purposes as medium term load forecasts. Furthermore, it is vital to provide reliable power system operation in the future.

Many authors have investigated which models produce an accurate prediction of future load consumption. Time series models are an instinctive approach to forecast short term power load. The majority of time series models are based on statistical methods that predict the future value of the variable by fitting a mathematical function of its previous values. (Hagan and Behr, 1987) present the results of fitting a Box-Jenkins model on a STLF problem. The final conclusion reads that Box-Jenkins models are very suitable for load forecasting issues both with a longer and a shorter horizon. Parenthetically, they remark that Box-Jenkins models are incapable of capturing a nonlinear relationship between load and exogenous variables. A practical

implementation of an ARMA model is described by (Fan and McDonald, 1994a), who integrated the influence of weather conditions on load behaviour. Satisfactory results are reported: the MAPE is primarily less than 2.0% for the forecasts of less than 24 hours ahead and beneath 2.5% for 168 hours ahead, where normal weather conditions are in place (Fan and McDonald, 1994b).

(Amjady, 2001) assess models that incorporate expert knowledge with a traditional ARIMA model, the expert knowledge provides a starting point for the model. The proposed model generates more accurate results than regular ARIMA models and ANNs.

One of the first papers that analyses the potential of the neural network as a time series forecasting mechanism for STLF is written by (Peng, Hubele, and Karady, 1992). The paper demonstrates that neural networks produce favorable results with regard to more traditional models accepted in literature. Their neural network for predicting load one week ahead comprised of one hidden layer, the amount of nodes in this hidden layer is varied from 6 to 12 in increments of 2. The training set consists of one year of historical data which is divided into five subsets corresponding to day types. The model selects training data that is similar to the inputs of the forecast (within a 85%-115% range). Earlier work selected training data from a moving window of two weeks (Peng, Hubele, and Karady, 1990), this procedure proved to produce large prediction errors for special events or holidays. Evaluation criteria for the training data turn out to be instrumental for the quality of the forecast. This paper reports MAPEs between 1.8116% and 2.8342% depending on the day of the week.

(Senjyu et al., 2002) focus on one-hour-ahead load forecasting. This research designed a neural network of one hidden layer with 20 nodes which predicts a correction on similar day data. The average load power of a selection of 10 similar days is taken as a starting point, the MAPE of this similar day prediction without correction is 1.28% which is a vast improvement from a regular linear regression with a MAPE of 13.08%. Inputs of the neural network are defined as deviations between the forecast day input and the average similar day inputs. The neural network is trained on a sample of data on the past 30 days before the forecast day and 60 days before and after the forecast day the previous year. The MAPE of correcting similar day data with a correction predicted by a neural network amounts to 1.18%. This approach was not merely of academical interest but it was also adopted by the Okinawa Electric Power Company in Japan.

A review on the application of neural networks for the STLF problem is provided by (Hippert, Pedreira, and Souza, 2001). They conclude with some critical commentary that, while a lot of positive results are reported on the employment of NNs in the field of STLF, some scepticism is justified. The arguments that they provide for this scepticism are firstly that most models in literature are overparameterized while having very small datasets, such a model would be prone to overfitting. Furthermore they argue that, though results are always tested on real data, the tests were often not systematically carried out.

Recent academic work frequently features LSTM models for time series problems such as STLF. The accuracy of a LSTM model is compared to a selection of alternative models by (Zheng et al., 2017), the alternative models are represented

by: SARIMA (seasonal autoregressive integrated moving average), NARX (nonlinear autoregressive neural network with exogenous inputs), SVR (support vector regression), NNETAR (feed-forward neural network model for univariate time series forecasting with a single hidden layer and lagged inputs.). The electricity data in this study is measured over 15-minute intervals and predicted 96 steps ahead, the training set consists of 904 samples of the last 10 days. The LSTM model outperforms all other models tested in this study. The authors also examine the error metrics of these models for a different, more simple dataset on airline traffic. For this dataset they found the SARIMA model to outperform the LSTM method but they argued that this was most likely due to extensive pre-processing in the form of seasonal differencing.

(Liu et al., 2017) compare hour-ahead forecasts from a LSTM model to those of an Elman neural network and find that the LSTM model outperforms the Elman model. More and more studies validate the success of LSTM models compared to other time series methods for STLF forecasting. The complicated nature of electricity data seems extremely suitable for LSTM models.

Chapter 3

Classical time series models

A time series data set is a series of observations ordered by time, conventionally these successive data points are spaced evenly over time. Commonly known examples of time series are disease rates, stock-market returns and migration data. Time series forecasting aims to develop a model that is capable of predicting future values based on historically observed data. The natural ordering of data is the key characteristic that distinguishes time series data from cross-sectional data. Cross-sectional studies analyse data from a population at a given point in time, it can for instance be applied to investigate the effect of income on health. Regressions are frequently employed to predict the dependent variables in cross-sectional data studies. Even though regressions can be used for time-series data as well, the time-dependent nature of the data allows for a myriad of models that are specifically applicable for time series forecasting.

Two of the most widely used approaches are exponential smoothing and ARIMA

3.1 ARIMA

A powerful model for describing stationary and non-stationary time series is the autoregressive integrated moving average process of order (p,d,q), denoted by ARIMA(p,d,q). The ARIMA model has been the subject of extensive research and was developed by George Box and Gwilym Jenkins (G. Box, 1970). The autoregressive part of the model is designed to predict future values of the dependent as a linear combination of its previous values. Thus an autoregressive model of order p, abbreviated as AR(p), can be represented by an equation in the following way: $y_t = c + \phi_1 y_{t-1} + ... + \phi_p y_{t-p} + \epsilon_t$. Where ϕ_i are the autoregressive parameters of the model that have to be estimated, c is a constant and ϵ_t is the error term. The other defining element of the ARIMA model is a moving average model of order q, MA(q). It specifies that the current observation depends linearly on current and past residual error terms of the model. The equation that accompanies this model is: $y_t = \mu + \theta_0 \epsilon_t + \theta_1 \epsilon_{t-1} + ... + \theta_q \epsilon_{t-q}$. Where $\theta_0 = 1$ and μ is the expectation of y_t. These two models combined compose the ARMA(p,q) model (3.1). Its parameters, θ and ϕ, can be estimated by Maximum Likelihood Estimation (MLE).

$$y_t = \mu + \sum_{i=0}^{q} \theta_i \epsilon_{t-i} + c + \sum_{i=1}^{p} \phi_i y_{t-1} \tag{3.1}$$

3.1.1 Stationarity

An important assumption of the ARMA(p,q) model is that the underlying time series is (weakly) stationary. A stationary time series is one, whose distribution does not rely on time. For weak stationarity it suffices that it has a constant mean and a covariance which only depends on the difference in time between the observations, this automatically implies a constant variance. When this condition is not satisfied, an ARIMA model can transform the dataset in order to make it stationary. The parameter d in this model stands for the degree of differencing that needs to be applied to make the process stationary. Generally first order differencing, $d = 1, \Delta_1 y_t = y_t - y_{t-1}$ or second order differencing $d = 2, \Delta_2 y_t = \Delta_1 y_t - \Delta_1 y_{t-1}$ will be sufficient to obtain a stationary process.

Definition 3.1.1 (Stationarity) *A time series process is called weakly stationary or second-order stationary if for all integers r, t and s the following properties hold:*

$$\begin{aligned} \mu(t) &= \mu < \infty, constant \\ \sigma(t) &= \sigma < \infty, constant \\ \gamma(t,s) &= \gamma(t+r, s+r) \end{aligned}$$

There are a number of tests that can be employed to assess the stationarity of a time series dataset. In this study, the augmented Dickey-Fuller test (ADF) will be used to test for the presence of unit root in the data.

Definition 3.1.2 *[The Augmented Dickey-Fuller test] The ADF test fits an autoregressive AR(p) model to the observed data, it subsequently uses the ordinary least squares (OLS) estimator of γ in order to make a statement about the presence of a unit root in the data*

$$\Delta y_t = \mu + \alpha t + \gamma y_{t-1} + \sum_{i=1}^{p-1} \phi_i \Delta y_{t-i} + \epsilon_t$$

The ADF test proposes a simple t-statistic as a ratio of departure from the estimated value to its hypothesized value. The critical values for this test are tabulated by (Fuller, 1976)

3.1.2 ACF and PACF

Estimating the ARIMA model requires identification of its parameters. The Box-Jenkins approach prescribes that the lag-coefficients can be determined from the autocovariance function (ACF) and the partial autocorrelation function (PACF). The definitions in this section will express the expectation of a random variable Y as $\mathbf{E}(Y)$

Definition 3.1.3 (Autocovariance function) *The ACF assigns to any two time periods $t, s \in \mathbf{Z}$ the covariance between Y_s and Y_t.*

$$\begin{aligned} \gamma_y(t,s) &= Cov(y_t, y_s) \\ &= \mathbf{E}\left[(Y_t - \mathbf{E}(Y_t))(Y_s - \mathbf{E}(Y_s))\right] \\ &= \mathbf{E}(Y_t Y_s) - \mathbf{E}(Y_s)\mathbf{E}(Y_t) \end{aligned}$$

Definition 3.1.4 (Partial Autocorrelation function) *The PACF gives the partial correlation of a stationary time series with its own lagged values.*

$$\alpha(0) = 1$$
$$\alpha(1) = Corr(Y_{Yt-1})$$
$$\alpha(h) = Corr(Y_t - P(Y_t|1, Y_{t+1}, ..., Y_{t+h}), Y_{t+h} - P(Y_{t+h}|1, Y_{t+1}, ..., Y_{t+h}))$$

Where $P(X|1, Y_{t+1}, ..., Y_{t+h})$ *is the best linear prediction of X given* $1, Y_{t+1}, ..., Y_{t+h}$

The value of the ACF for each lag is comprised of both the direct correlation between y_t and y_s and indirect correlations. These indirect correlations are a linear function of the correlations of the observation, with observations at the time steps between the two lags. The PACF removes these indirect correlations, and takes into account only the direct correlation between two observations. Plots of the ACF and PACF provide some valuable intuition about the orders of the autoregressive and moving average elements of the ARIMA model. The PACF plot describes the direct relationship between the dependent variable and its lag. Assuming that the observed data is partly generated by an AR(p) process, it can be expected that the PACF shows no correlation beyond lag p. Thus, we select for p the number of lags that appear to be significantly different from zero in the PACF plot. For the moving average parameter q, select the lag for which the ACF starts declining.

3.1.3 ARIMAX

The ARIMA model considers only past observations of the dependent variable as inputs of its forecast, it thus assumes that future values will depend linearly on its past values and the values of its stochastic shock. However, as discussed before, there are several other elements that are expected to be relevant for the prediction of the MCF. Information about the weather is one obvious illustration of a variable that might be helpful in explaining some of the variation in the MCF.

The ARIMA model can be extended to additionally incorporate exogenous variables into the equation. The ARIMAX model can therefore be interpreted as a multiple regression with one or more autoregressive terms and one or more moving average components. This merge of ARIMA and multiple regression is commonly referred to as a dynamic regression model.

$$y_t = \mathbf{X}\beta + \mu + \sum_{i=0}^{q} \theta_i \epsilon_{t-i} + c + \sum_{i=1}^{p} \phi_i y_{t-1} \tag{3.2}$$

Chapter 4

Neural Networks

4.1 Artificial Neural Networks

An artificial neural network (ANN) is an information processing mechanism structured as an interconnected group of nodes, which is inspired by a simplification of neurons in the brain. Neural networks are comprised of vertically stacked components called layers. Three types of layers can be distinguished:

- **input layer**: passes the initial data into the network for further processing
- **hidden layers** : perform nonlinear transformations on the input variables produced by the preceding layers. Hidden layers allow break down the function of a neural network into particular transformations of the data.
- **output layer**: produces the output of the ANN

Each layer contains a set of neurons, each of which itself maps input data to an output data space. The mapping of input data to an output space is determined by a set of parameters and an activation function. All nodes in consecutive layers are connected, activations in one layer determine the activations in the next layer. These connections, frequently referred to as edges, are weighted to determine the strength of one node's influence on another. Weight $w^h_{i,j}$, for instance, represents the strength of the connection between node i in layer h and node j in layer $h+1$. Additionally each neuron has its own bias, b^h_i, which can be thought of as the intercept in a linear equation. The value of a neuron can be computed as a weighted average of its inputs, shifted by the bias and transformed by a nonlinear activation function. The weights and biases are the parameters of the neural network. These parameters can be 'learned' by performing an optimization algorithm on a training set X_i for which the outcomes y_i are known. This process is called training the neural network.

4.1.1 Single Layer Perceptron

The single layer perceptron (SLP) is the simplest form of a neural network, in addition it was the first proposed neural network created by Rosenblatt (Rosenblatt, 1961). A SLP is capable of classifying linearly separable patterns with a binary output (4.1). The SLP consists of an input layer and an output layer, it starts with a randomly chosen set of initial weights $\mathbf{w} = [w_0, w_1, w_2, ...]$ where w_0 is the bias. $\mathbf{x} = [x_0, x_1, x_2, ...]$ is the set of input variables where x_0 is set to 1 to accommodate the bias. Figure 4.1 displays a very simple view of the data processing in a SLP.

$$f(\mathbf{x}, \mathbf{w}, b) = \begin{cases} 0, & \text{if } \langle \mathbf{w}, \mathbf{x} \rangle \leq 0 \\ 1, & \text{if } \langle \mathbf{w}, \mathbf{x} \rangle > 0 \end{cases} \tag{4.1}$$

The single-layer perceptron is non-differentiable at $x = 0$, this means that a gradient-descent algorithm will not be able to make progress in updating the weights. The initialized weights should be updated each training sample, the weight update can be written formally as $\mathbf{w} = \mathbf{w} + \Delta\mathbf{w}$. Where $\Delta\mathbf{w} = \eta(\mathbf{y}^* - \hat{\mathbf{y}})\mathbf{x}$, η is the learning rate of the model and y^* and $\hat{y}$ are the target output value and the predicted output value respectively. This process iterates until convergence is reached, convergence is only guaranteed if the classes are linearly separable. This deficiency was proved in 1969 by (Minsky and Papert, 1969) who found that a SLP was not able to solve a simple XOR classification problem. An XOR function should return a true value if the two inputs are not equal and a false value if they are equal. XOR input values are not linearly separable and can therefore not be classified by a SLP. The Multilayer Perceptron was designed to solve this problem, it is a neural network where the process from input to output is of non-linear form.

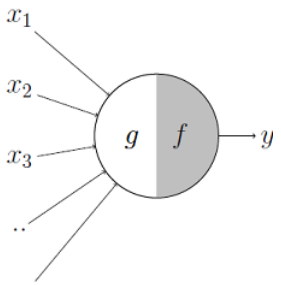


FIGURE 4.1: Visualisation of a single layer perceptron

4.1.2 Multilayer Perceptron

A multi-layer perceptron (MLP) has the same structure as a SLP, but is extended with one or more hidden layers. They fall under the category of feedforward algorithms; inputs are combined into a weighted sum and subjected to the activation function. Each activation is propagated to neurons in the next layer, every layer is feeding the next one with their internal representation of the data. Each hidden layer is fully connected with the previous hidden layer and the output layer has full connectivity with the last hidden layer. Information moves forward, never backwards, through the network. One example of a MLP is given by Figure 4.2.

Let the inputs of the model be defined by $x_i^0, i = 1, 2, ..., n$ where n represents the number of input variables. These inputs enter the model through the input layer, layer $l = 0$. The dimension of layer l is denoted by m_l and the output of its neurons are denoted by $x_i^l = [x_1^l, x_2^l, ..., x_{m_l}^2]$. Layer l will thus make m_l linear connections of the inputs processed by the previous layers. The weight matrix that is applied to construct these linear combinations is given by $W_l \in \mathbb{R}^{m_{l+1} \times m_l}$, with elements $w_{i,j}^l$ representing the weights between node i in layer l and node j in layer $l + 1$. $w_{0,j}^l$ is the bias of node j in layer $l + 1$. Any intermediate activation z of node j in layer $l + 1$ can be described by z_j, which is subsequently transformed by the nonlinear activation function $\varphi : \mathbb{R} \to \mathbb{R}$ of the hidden layer to produce its output x_j^{l+1}.

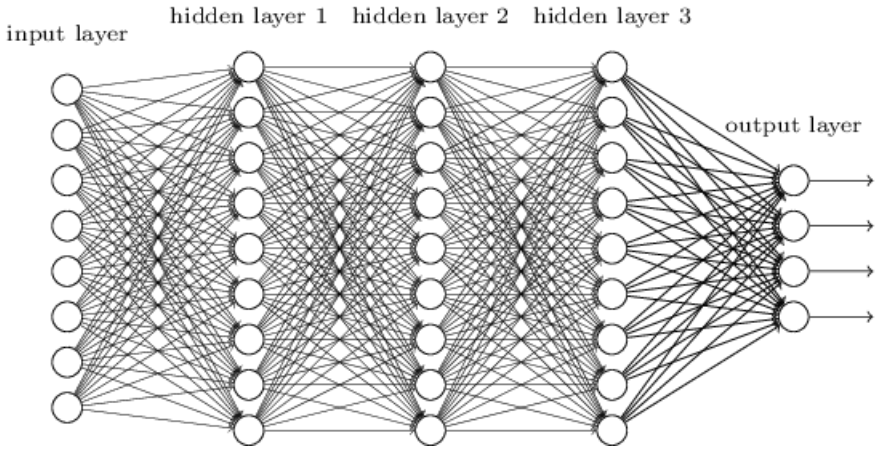


FIGURE 4.2: Architecture of a MLP

$$
\begin{aligned}
z_j^{l+1} &= \sum_{i=1}^{m_l} w_{i,j}^h x_i^l + w_{0,j}^l, \quad j = 1, ..., m_{l+1} \\
&= \sum_{i=0}^{m_l} w_{i,j}^l x_i^l \\
x_j^{l+1} &= \varphi(z_j^{l+1})
\end{aligned}
$$

The number of hidden layers and the number of neurons contained by each layer belong to the set of hyperparameters of the model, they can be tuned to enhance performance. "A feedforward network with a single hidden layer containing a finite number of nodes can approximate any continuous function on a compact subset of $\mathbb{R}^z$, under mild assumptions on the activation function φ." (Hornik, Stinchcombe, and White, 1989) Figure 4.3 visualizes how the data moves through each neuron in the network.

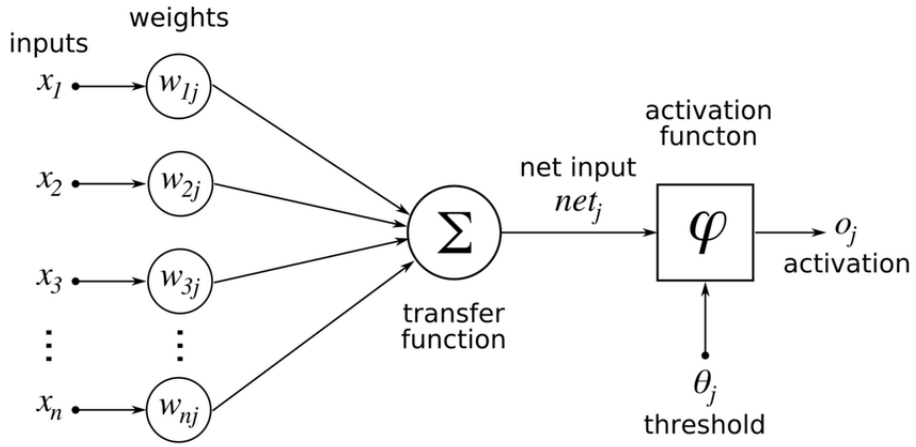


FIGURE 4.3: Data process in a single neuron

4.1.3 Training the network

When the network architecture is structured as desired, the network is ready to be trained. Training involves repeatedly adjusting the parameters to attain an optimal prediction. The optimization process can be regarded as a search for candidates of parameters that minimize an objective function. The objective function is a cost function that evaluates the performance of a candidate set of parameters. One requirement for optimization is that the cost function is continuous. A cost function is usually formulated by a quantification of the deviation of the prediction from its true value, the mean squared error is a commonly used example of such a quantification.

The training process is initiated by initializing the parameters of the model randomly. The cost function evaluates the performance of the network with the initialized parameters. The cost function will be large when the network fails to accurately predict the true output value. The cost function takes the parameters of the neural network as an input and maps those to the defined error metric, the mean squared error. The objective of training is to find inputs of the cost function that minimize the output. The gradient of the cost function will give the direction of steepest ascent, naturally the negative gradient provides the direction of steepest descent. A gradient descent algorithm computes for each step in the process the gradient of the cost function, alters the parameters in that direction and iterates this process until a (local) minimum is reached. The gradient descent algorithm is formulated by (4.2) where α is the learning rate, which determines how much the parameters must be changed each iteration. The learning rate determines the speed at which the model trains. $C(W)$ is the cost function that belongs to the corresponding parameters of the model at that iteration. Gradient descent algorithms are the foundation for most of the established optimization algorithms for neural networks.

$$w_{i,j}^{h} = w_{i,j}^{h} - \alpha \frac{\partial}{\partial w_{i,j}^{h}} C(W) \tag{4.2}$$

Backpropagation is the algorithm that facilitates efficient computation of the gradient for a single training example. A gradient descent step averages the gradients for all training examples. However this method is computationally slow and therefore often infeasible. Instead, the data is shuffled and subdivided in mini-batches, each step can be computed with respect to a mini-batch. Repeatedly going through all the mini-batches is called stochastic gradient descent (SGD).

4.2 Recurrent Neural Networks

A recurrent neural network (RNN) is a type of ANN that is appropriate for processing sequential data. ANNs are assumed to have all inputs (and outputs) independent of each other, RNNs make it possible to loosen that assumption. A RNN relates the computations at a certain time step to its prior history and its memory of the computations at a prior time step. This is achieved via a so-called recurrence relation where an internal state is passed forward through the model across time. The intuition is that the state captures some notion of memory, as a result the output is dependent not only on the input at a certain time step but also on the memory passed forward from a prior time step.

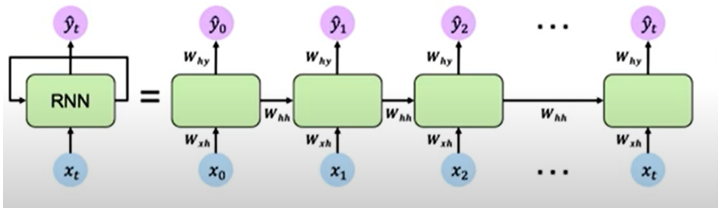


FIGURE 4.4: Recurrent Neural Network

Figure 4.4 exhibits a computational graph of a RNN. The left side shows the unfolded state of a RNN and the right side unfolds that RNN into a network. The key idea is that the RNN maintains an internal state h_t, that is going to be updated at each time step as the sequence is processed (4.3). This updated internal state is then weighted and transformed by an activation function to produce a predicted output (4.4). Every time step uses the same weight matrices W_{hh}, W_{xh} and W_{hy}, therefore, the parameterization cost of an RNN does not grow as the number of time steps increases.

$$h_t = \phi_h(W_{hh}^T h_{t-1} + W_{xh}^T x_t) \tag{4.3}$$

$$\hat{y}_t = \phi_y(W_{hy}^T h_t) \tag{4.4}$$

4.2.1 Backpropagation through time

The algorithm that is employed to update the weights during training is called backpropagation through time (BPTT) and it is strongly related to the backpropagation algorithm for neural networks. Conceptually BPTT works by unrolling the network and aggregating the errors across each time step (4.5). The BPTT moves backwards from the final error through the outputs, weights, activation function and inputs of each layer to calculate the gradients using the chain rule. The longer the sequence, the lengthier the derivation of the gradients. The gradient of the cost function can be calculated with respect to the three weight matrices. Those gradients are then used to adjust the weights in a similar manner as for ANNs

$$C = \sum_{t=1}^{T} C_t \tag{4.5}$$

The gradient of the loss function must be calculated with resepect to the parameters of the RNN model; W_{hy}, W_{xh}, W_{hh}. The derivation of the gradients will be formulated below, once a cost function is chosen that can be filled in, in order to obtain the gradient for RNN. Weight matrix W_{hy} is present only in the function of $\hat{y}$ (4.4). The gradient for this matrix is given by (4.6).

$$\frac{\partial C_t}{\partial W_{hy}} = \frac{\partial C_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial W_{hy}} \tag{4.6}$$

Weight matrix W_{hh} appears in the function of h_t (4.3) directly and then through h_t in the formulation of $\hat{y}_t$ (4.4). (4.7) defines the gradient of this weight matrix, however the final term has to account for the fact that h_t also depends on W_{hh} implicitly through earlier memory states. This dependence is incorporated in (4.8)

$$\frac{\partial C_t}{\partial W_{hh}} = \frac{\partial C_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial h_t} \frac{\partial h_t}{\partial W_{hh}} \tag{4.7}$$

$$\frac{\partial C_t}{\partial W_{hh}} = \frac{\partial C_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial h_t} \left(\sum_{k=1}^{t} \frac{\partial h_t}{\partial h_k} \frac{\partial h_k}{W_{hh}} \right) \tag{4.8}$$

The gradient for weight matrix W_{xh} is defined in a similar manner

$$\frac{\partial C_t}{\partial W_{xh}} = \frac{\partial C_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial h_t} \frac{\partial h_t}{\partial W_{xh}} \tag{4.9}$$

$$\frac{\partial C_t}{\partial W_{xh}} = \frac{\partial C_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial h_t} \left(\sum_{k=1}^{t} \frac{\partial h_t}{\partial h_k} \frac{\partial h_k}{W_{xh}} \right) \tag{4.10}$$

4.2.2 Vanishing gradients

Theoretically a RNN should be able to capture all temporal dependence in sequential data, however it turns out that RNNs experience significant difficulties when facing a long-term past observation. This deficiency is caused by a phenomenon referred to as the vanishing gradient problem. During backpropagation weights are corrected each iteration by the partial derivative of the cost function with respect to the weight. However the gradients of memory state weight W_{hh} and the input variable weight W_{xh} are heavily influenced by their own values because of the chain rules. Weights are often initialized at a relatively small value and to add to that, most popular activation functions are also designed to produce small outcomes. The gradients will thus become very small, which prevents the algorithm from changing the weights in a meaningful way. This will slow down the training process and eventually cause the network to stop training entirely. The exponential decay of gradients is a common occurence in RNN and often referred to as the vanishing gradients problem.

$$\frac{\partial C_t}{\partial W_{xh}} = \frac{\partial C_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial h_t} \left(\sum_{k=1}^{t} \frac{\partial h_t}{\partial h_k} \frac{\partial h_k}{W_{xh}} \right)$$

$$\frac{\partial h_t}{\partial h_k} = \prod_{t \geq i \geq k} \frac{\partial h_i}{\partial h_{i-k}} W_{hh}^T diag \phi_h'(x_{i-1})$$

The derivation above shows the dependence of the gradient on weight matrices. Under certain assumptions on the activation function ϕ it can be calculated for which eigenvalues of the weigh matrices the vanishing gradient problem will be likely to occur.

4.3 Long Short Term Memory

Long short term memory models were created by (Hochreiter and Schmidhuber, 1997) as a solution to the short memory limitation of regular RNNs, they are supposed to tackle the vanishing gradient problem.

LSTM cells are composed of two types of states: the hidden state h_t and the cell state s_t. The hidden state operates exactly the same as in the simple RNN models; it carries past information through the model as a linear combination of the input and the previous state. The hidden state transfers information from the immediate past, this information is overwritten at every time step. The shortcoming of a RNN model is alleviated by the novel component of the cell state. The cell state is the component that possesses the capability of long term memory.

In principle, information can move through the cell state uninterrupted. The LSTM cell has the possibility to alter the information in the cell state by removing or adding information to it. The process of removal and addition is thoroughly regulated by structures called gates. Gates control the way sequential information enters, is stored and leaves the model. A regular LSTM unit contains three gates: the input gate, the forget gate and the output gate. Gates basically consist of a non-linear activation function such as sigmoid (4.11) or tanh (4.12). The sigmoid activation function, produces output values that take on values between 0 to 1 in a non-linear fashion. A value of zero means that no information can pass through the gate, conversely a value of one allows information to pass.

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{4.11}$$

$$tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{4.12}$$

The first gate is the forget gate, represented by f_t with its parameters W_f and b_f. This gate determines what information the model should forget. The inputs x_t and the previous hidden state h_{t-1} are transformed into values between 0 and 1 by a sigmoid layer. These mapped values are then pointwise multiplied with the previous cell state s_{t-1} and thus determine which numbers in the cell state should be kept and which should be thrown away. Figure (4.5) illustrates the procedure of the forget gate. And equation shows the associated formula.

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{4.13}$$

$$\tag{4.14}$$

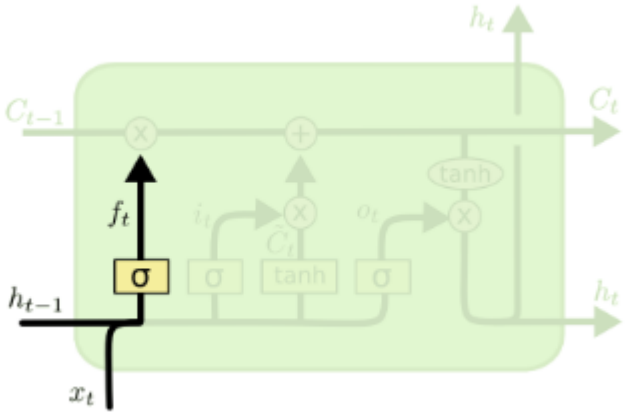


FIGURE 4.5: The Forget gate of a LSTM network
Understanding LSTM networks n.d.

The next gate is the input gate, i_t with weight matrix W_i and bias b_i. This gate consists of two part. The first part (4.15) is again a sigmoid layer which determines what values of the cell state should be updated. The second part (4.16)is a tanh layer that produces a set of candidate values for the cell state $\tilde{s_t}$. The values between 0 and 1 from the first part of the input gate will scale the candidate values by means of pointwise multiplication. These candidate values will then be added to the cell state s_t. The transfers in the second gate are exhibited visually in Figure 4.6

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{4.15}$$
$$\tilde{s_t} = tanh(W_s[h_{t-1}, x_t] + b_s) \tag{4.16}$$

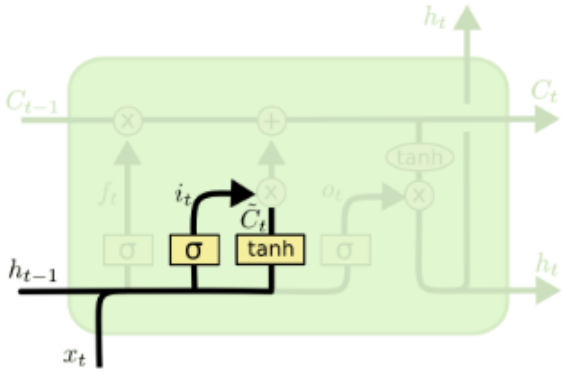


FIGURE 4.6: The input gate of a LSTM network
Understanding LSTM networks n.d.

The new cell state s_t is therefore a combination of the old cell state, multiplied by the vector from the forget gate, and the new scaled candidate values from the input gate (4.17).

$$s_t = f_t \cdot s_{t-1} + i_t \cdot \tilde{s_t} \tag{4.17}$$

The last gate is the output gate, o_t. The output gate decides on the formation of the hidden state. A sigmoid layer decides what parts of the cell state are suitable for output based on the previous hidden state and the input layer (**??**. The cell state is then transformed by a tanh activation function and pointwise multiplicated by the outputs of the sigmoid layer. These values are then transferred to the next time step as the hidden state (4.19). This last gate is visually represented by figure 4.7.

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{4.18}$$
$$h_t = o_t \cdot tanh(s_t) \tag{4.19}$$

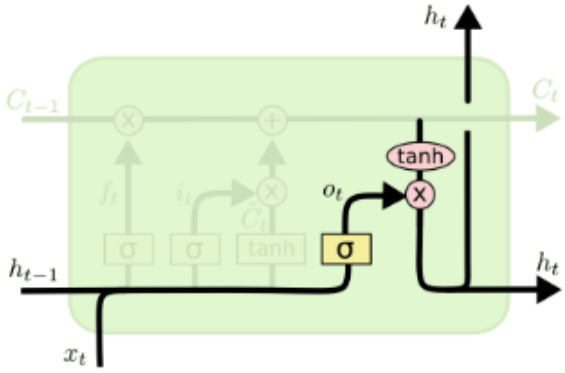


FIGURE 4.7: The output gate of a LSTM network
Understanding LSTM networks n.d.

Chapter 5

Data

This chapter addresses the data gathering process. Section 5.1 will describe the data that is used in this research and mention the various sources from where it was obtained. In section 5.2 there will be an exploratory data analysis on the exogenous and endogenous variables of this project. Data preparation includes the actions that are required to transform the raw data from the databases into clean data that can be used directly as input for the forecasting models in this research. Section 5.3 will feature an elaborated description of the data preparation process. Lastly, section 5.4 will describe the necessity and approach of splitting the data into a training, validation and a test set.

5.1 Data collection

The energy demand of a single supplier is generally described by the term system load, which is the sum of individual demands at all the connections that the supplier serves. Theoretically, the system load could be calculated if each individual consumption pattern at each connection was known. However, the consumption patterns of individual customers display a high level of randomness and diversity. Additionally, not all individual connections are equipped with the appropriate metering techniques to read out sub-hourly consumption data. Therefore it is preferred to predict the consumption patterns of the totality of the individual loads.

System load is traditionally assumed to be determined by a number of separate components, including but not limited to:

- Economic factors
- Weather effects
- Time
- Random effects
- Historical load

5.1.1 Economic Factors

Economic factors have significant influence on electricity consumption patterns. Aspects such as level of industrial activity, area demographics, changes in the farming sector and developments in the regulation of sustainable energy sources have significant impacts on system load growth or decline (Gross and Galiana, 1987). These features do not vary over short periods of time and are generally of the same order in the grid areas of interest. Because of the longer time scales associated with these economic factors they will not be represented in this thesis. However, it might be

useful to account for such factors periodically. Another economic factor of interest is the price of electricity, several researches have revealed the existence of negative price elasticity of demand in the electricity market. Al Faris (2002) presents empirical results that indicate that both income and price have an impact on energy demand. They report an average short-run electricity price elasticity in the GCC (Cooperation Council Countries) countries of -0.09. Additionally, they find significantly larger long-run elasticities. Al-Faris, 2002 argue that elasticities become larger as the horizon increases because: "As time passes, consumers will have more flexibility to curtail their demand, either through conservation or substitution, in response to higher tariffs or taxes.". Most studies find similar or even higher results. Whereas nearly all empirical studies are based on yearly or quarterly data, Lijesen, 2007 considers the real-time elasticity of electricity. A price elasticity of -0.0043 follows from a log-linear specification. It can be concluded that the dependency of demand on electricity prices increases over time, this suggests that the addition of a lagged price variable to the model might expand its explanatory power. This thesis will include a 30 day lag of day-ahead electricity prices as a variable in the predictive models. Trading on the Dutch day-ahead market is centralized at the European Power Exchange (EPEX), the day-ahead electricity prices are thus referred to as EPEX spot prices. The EPEX spot prices in this study are provided by ENTRNCE.

5.1.2 Weather Effects

Weather effects are universally recognised as one of the most important explanatory variables for electricity usage. The effect of weather is most significant for domestic and agricultural consumers, but it can also influence the demand patterns of commercial customers (Fahad and Arbab, 2014). Out of the broad of range of weather variables, temperature is the most commonly used factor. The intuition behind the explanatory power of temperature is quite straightforward: on cold days demand rises due to increased use of electric space- and water heating devices, on hot days demand for electricity will be driven up by the use of air conditioning. From this intuition it can be easily derived that the relationship between energy demand and temperature will most likely be non-linear. Valor, Meneu, and Caselles, 2001 investigate the relationship between air temperature and electricity consumption, they observe a non-linear relationship with some regions of non-sensitivity. In their review paper Hippert, Pedreira, and Souza, 2001 conclude that out of 23 papers, 13 authors only used temperature as an independent variable, 3 papers did not include temperature and the remaining 7 added additional variables. The choice of not adding additional weather variables often depended heavily on the lack of available data.

The importance of weather effects is closely linked to the increase in renewable energy sources, whose output is often dependent on the weather. Because of its strong correlation with the weather, renewable energy generation is extremely variable. Its output fluctuates at timescales ranging from minutes to hours to multiple days. A wide range of weather conditions can affect the generation of energy through renewable sources such as solar panels and wind turbines. Furthermore, these relations between weather variables and renewable sources are complicated and subject to combined effects. Staffell and Pfenninger, 2018 show that year-to-year variability of electricity demand will increase by 80% before 2030. Apart from temperature, a selection of weather variables that is often considered in short term load forecasting consists of: radiation, wind, humidity and cloudiness.

Historical Weather

The historic weather data is collected from a KNMI API, this dataset contains numerous variables ranging from temperature to humidity measured at 47 weather stations located throughout the Netherlands. An extensive overview of all the weather stations, their coordinates and the percentage of available data over the period 2018-2022 is provided in B.1. The weather stations without sufficient available data are removed from the dataset, leaving 34 weather stations. Each grid area will be assigned a weather station from this selection. The matching process of grid areas and weather station is based on their longitudinal and latitudinal coordinates, which are linked to the zip codes in the grid areas. Although the range of variables provided by KNMI is extensive, a lot of the variables report a huge amount of missing data. Three weather variables from this dataset will be taken into consideration as inputs. The choice of these variables depends on the presence of a similar forecasted weather variable for the test data. The three variables of interest are: temperature (T) measured in 0.1 Celsius at a height of 1.5m, Radiation (Q) in (J/cm^2) per hour and cloudiness (N) measured in oktas.

Standard practice in load predicting papers is to use observed values for weather variables in order to assess the performance of forecasting models. In practice, the forecast error on the weather variables introduces an extra element of uncertainty to the prediction. Unexpected events such as a storm can influence weather variables and make predictions based on forecasted values unreliable.

5.1.3 Forecasted Weather

It is essential that the input data of the forecast matches the training input data. Therefore it would be preferred to collect the weather forecasts from the same source as the historic data. However, KNMI does not yet provide a free and accessible way to import their weather forecasts. The forecasts for radiation are obtained from SolCast, which provides solar irradiance data at 30 minute intervals, measured in W/m^2. Temperature and Cloudiness are imported via an API from openweathermap at an hourly rate, they are measured in respectively Kelvin and percentages.

5.1.4 Time

Time elements affect the energy consumption patterns, a distinction can be made between periodicity and special events such as holidays. Periodicity refers to the fact that energy consumption follows a well-defined pattern. In the short term, energy usage follows a daily and weekly pattern, which is shaped by the work-rest cycle of a community. Figure 5.1 displays an example of a weekly load profile from the NEDU documentation. Peaks of energy usage occur early in the morning and later in the evening when people either go to work or are presumably spending time at home cooking or watching television with some lights on. Troughs can be observed between peaks at night and during the middle of the day. Additionally, weekends often have slightly different patterns than weekdays. Changes in the load profile can also occur more gradually over a longer period of time. In the winter days get shorter and colder, causing demand for electricity to rise. Figure 5.1 clearly depicts the distinction between electricity consumption in the winter and the summer.

Some time elements, such as a specific holiday or the change to daylight saving time, have a more immediate effect on the load pattern.

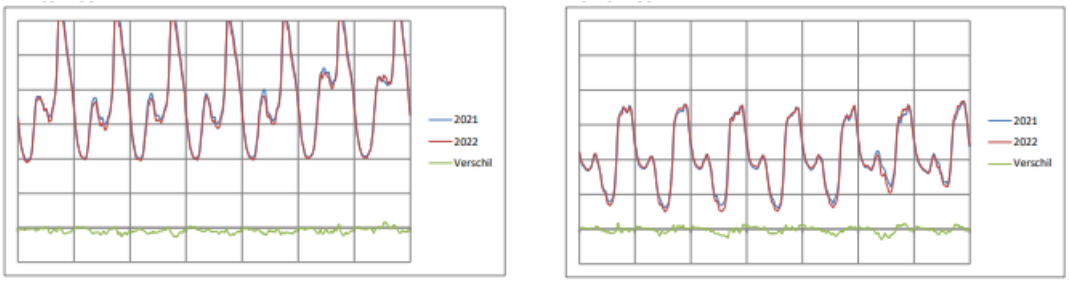


FIGURE 5.1: Example of a load profile in a winter week (left) and a summer week (right)

Time elements are considered a major determinant in predicting system load. However, time elements are already incorporated in the final prediction through profiled fractions. Therefore, it might be pointless to include them in the prediction of the MCF. In section 5.1, a fast fourier transformation is applied to the data to extract frequencies from the data. This will allow for analysis of seasonality in the MCF data as well as the profiled fractions. Initially, time elements will not be considered as an input variable because they are already accounted for in the final prediction. The influence of adding variables corresponding to certain time elements will be incorporated as a robustness check.

5.1.5 Random effects

The energy demand is subject to shocks due to randomness or unobservable events, these shocks to the system load are classified as the random effects. Shocks to the system can be unique to an individual connection, caused by inherent factors. The system load is after all a composite of diverse individual demands. This type of shock can be diversified if the amount of connections in a grid is large enough. If this is the case there remains a systematic disturbance factor. Variation in the energy demand can be caused by unobserved or unpredicted factors that shift the energy demand of all individual connections in the same direction. Some typical examples of random disturbances are: increased usage of television because of a specific program of interest and government compulsory demand side management due to predicted shortfall of electricity. However, the most common source of random disturbance is probably unexpected weather effects on the generation of renewable energy.

5.1.6 Historical load

Electrical load data is recorded per quarter, per grid area. The period for which historical load data is available depends on when VanHelder started to supply energy to active connections in the corresponding area. For the majority of grid areas there is data available from October 2018 onwards. At the time of writing this research, VanHelder is serving connections in 17 grid areas, the latest of which joined in January 2020. Indicating that at least two years of historic data is available for each area. A detailed overview of the grid areas and the period for which data is available on these areas, is provided by Appendix C. The historic MCF per quarter for each grid area is delivered by ENTRNCE on a weekly basis. Informative variables on date and time are extracted from the datetime labels of these historic MCF entries. The

historical data also provides the possibility to construct a lagged dependent variable.

Table 5.1 displays a list of all the input variables of the model, its dimensions and a brief description.

TABLE 5.1: Information on input variables

Variables in Data	Dimension	Description
Year	NA	Year
Month	NA	Month
DayOfYear	NA	day as number of the year (jan 1st = 1)
Weekday	NA	1 if weekday, 0 if weekend
Hour	NA	from 0 to 23
Minutes	NA	in intervals of 15
Temperature	$^{\circ}C$	Temperature in 0.1 $^{\circ}C$ measured at 1.5m
Clouds	Okta	Cloudiness (degree of air coverage)
Price	€/kWh	Spot price of electricity
Radiation	J/cm^2	Global radiation

5.2 Exploratory Data Analysis

Exploratory data analysis (EDA) is the first step in any data-driven problem solving approach. EDA aims to maximize initial insight into a data set and its underlying structure. Firstly, the characteristics of the data will be discussed in part 5.2.1. Section 5.2.2 will visualise the data and take an in depth look at its distributional qualities. The presence of seasonal components will be discussed and analysed by means of a Fast Fourier Transformation Transformation in section 5.2.3. Lastly, the EDA will focus on a simple correlation analysis in order to extract some linear dependencies between the variables in part 5.2.4.

5.2.1 Descriptive Statistics

The dataset comprises of 1.928.444 data points collected from 17 separate grid areas on a sub-hourly basis. A detailed list of the 17 grid areas and the period of available data is disclosed in appendix C. The grid area that was added most recently was added to the dataset on the first of January 2020. This means that for each grid area, at least two years of historical data has been recorded. Every day thus contains 96 observations.

Parameters	MEAN	MEDIAN	SD	MIN	MAX
Entire dataset					
MCF	0.9071	0.9121	0.2280	-7.511	6.11707
Temperature	102.28	95	62.90	-131	402
Radiation	45.25	2	73.33	0	347
Clouds	6.15	8	3.05	0	10
Price	0.0880	0.0499	0.0816	-0.0792	0.7
Year=2019					
MCF	0.9158	0.9142	0.1365	-0.0130	1.7502
Temperature	112.94	107	65.76	-91	402
Radiation	38.13	0	66.98	0	351
Clouds	6.04	8	3.09	0	9
Price	0.0409	0.0471	0.0113	0.0090	0.1215
Year=2020					
MCF	0.9057	0.9143	0.2284	-4.18579	2.41236
Temperature	116.71	109	62.90	-48	359
Radiation	47.06	2	76.35	0	338
Clouds	5.77	8	3.23	0	10
Price	0.0323	0.0317	0.0153	-0.0791	0.2000
Year=2021					
MCF	0.8988	0.9028	0.2924	-7.511	6.1192
Temperature	103.10	101	63.44	-131	327
Radiation	37.27	1	64.77	0	351
Clouds	7.00	8	2.41	0	9
Price	0.1030	0.0783	0.0747	-0.0662	0.62

TABLE 5.2: Descriptive statistics

The descriptive statistics of the variables are recorded in table 5.2. The table also separately considers each full year of data in order to identify a potential time trend. The mean MCF amounts to 0.9071, which suggests that on average the profiled fractions tend to overestimate the energy demand by approximately 10%. Over the years, a slightly declining trend in the MCF is reflected by the diminishing MCF's per year. The median matches the MCF rather closely, which seems to denote no remarkable skewness of the distribution. Another striking observation is the growth of the standard deviation, it more than doubles in three years starting at 0.1365 in 2019 to 0.2924 in 2021, the average over the whole dataset settles at 0.228. The rise in volatility can also be noted from the minimum and maximum values of the MCF that become more extreme as the years go by. The most obvious reason for volatility in the MCF's is the added factor of uncertainty caused by the ever-increasing dependency on renewable energy.

Another source of variability in the MCF's stems from the underlying amount of connections in a grid area. The underlying profiled volumes that partially determine the MCF's can vary immensely depending on the number of active connections at that time in that area. MCF's that are based on a large amount of data are evidently more reliable than MCF's that are based on data from merely a small number of connections. Intuitively, the MCF's of a grid area with only a few active connections will be more susceptible to fluctuations of one single connection. This idiosyncratic risk is diversified in grid areas that serve a larger customer base.

The weather variables do not display a distinctive trend over time. A correlation study in section 5.2.4 will reveal more about the relationship between those variables and the MCF. The market price of electricity is extremely sensitive to changes in demand and supply. Following the shift to renewable energy sources, this might also entail that the market price could become increasingly dependent on some weather variables. This could also be the reason that the volatility of the prices experience a significant increase from 2020 to 2021. The last year has been turbulent for the energy market because of the major increase in prices. This only illustrates the responsiveness of the electricity market to policies and events. For the first three months of 2022, an average price of 0.20593 is recorded. These soaring prices could be a reason for accelerated transition to green energy sources in the future.

5.2.2 Time Series and distributional Analysis

Time Series analysis is a sub-area of mathematics that analyses a sequence of data points that are collected over equidistant intervals of time. Time is a crucial variable because it offers an additional source of information about how the data adjusts over time and a set order of dependencies between the data.

Figure 5.2 displays a line plot of the MCF averaged over the 17 grid areas on the full range of date points. The red line represents the moving average with a one month window. From the descriptive statistics it could already be observed that the MCF experiences a small decline, this effect is hardly noticeable in the graph. The feature that draws attention is the variance of the MCF, the increase of volatility manifested itself in the descriptive statistics and is also apparent from the time series plot. The figure displays more and more extreme values and fluctuations through time, whereas the 1 month moving average stays roughly the same.

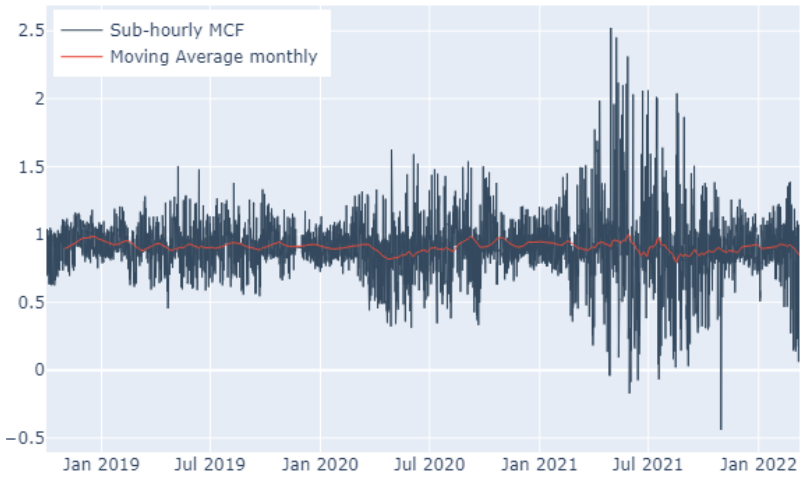


FIGURE 5.2: Time series with moving average (monthly) of MCF averaged over all grid areas

To obtain a better understanding of the behaviour of the MCF, some additional analysis techniques are applied and visualised. "Kernel density estimation (KDE) is a non-parametric method to estimate the probability density function of a variable" (Rosenblatt, 1956). It allows for inferences on the distribution of a variable based on the data sample on hand. The KDE on the MCF's, averaged over grid areas, is presented in Figure 5.3 on the left. The distribution closely resembles a normal distribution, with the mean and variance already discussed in section 5.2.1. The distribution seems quite narrowly concentrated around the mean, however the tails of the distribution reach reasonably far. 90% of the data lies between 0.569 and 1.211. Another option to illustrate the relative frequency distribution of the data is to plot the emperical distribution function (EDF), as depicted on the right side of Figure 5.3. The EDF is an estimate of the cumulative distribution function generated by ordering all the unique observations and calculating the cumulative probability as the number of observations less than or equal than the current observations, divided by the total number of observations. The same conclusions follow from the EDF, the median of the dataset is located at the 0.5 proportion. The steepness of the EDF corresponds to the narrowness of the KDE.

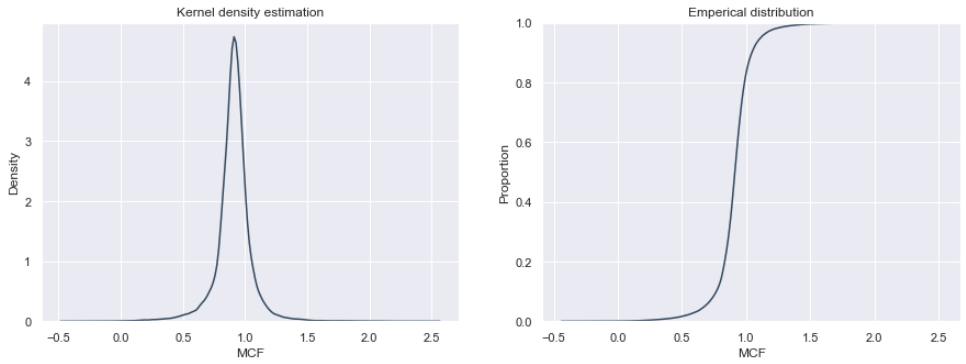


FIGURE 5.3: Kernel density estimation (left) and Emperical distribution function (right) of average MCF's

5.2.3 Fast Fourier Transformation

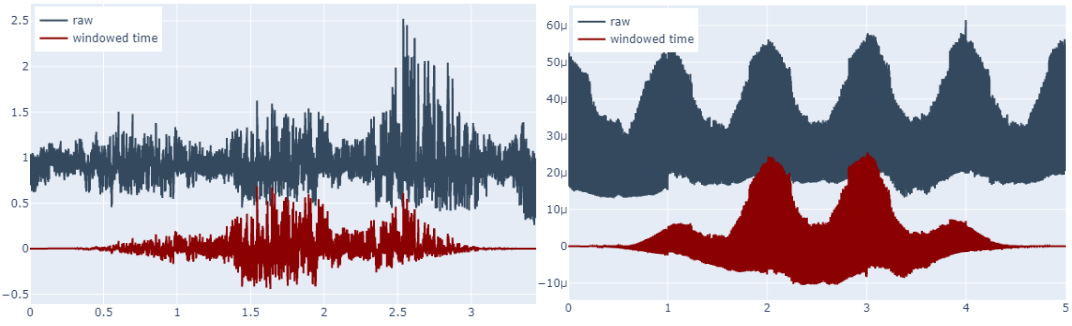
A time series is often expected to display some extent of seasonality. A Fourier transformation deconstructs a time domain representation of a signal into the frequency domain representation. This allows data to be deconstructed into separate seasonal components such that the different frequencies can be analysed. The Fast Fourier Transformation is given by (5.1)

$$F[k] = \sum_{n=0}^{N-1} f[n] \times e^{-i2\pi k \frac{n}{N}} \tag{5.1}$$

Fast Fourier Transformation is subject to some serious limitations. It assumes that the time domain will continue forever, although the analysis is based on a finite set of data. FFT implicitly assumes that the signal repeats itself after the measured interval. Thus, the two endpoints of the data are interpreted as if they were connected, this causes glitches in the assumed signal. These artificial discontinuities appear in the frequency domain as high-frequency components, which do not occur in the original signal, this leads to spectral leakage.

$$w[n] = \alpha_0 + \alpha_1 \cos(\frac{2\pi n}{N}) + \alpha_2 \cos(\frac{4\pi n}{N}) \tag{5.2}$$
$$\alpha_0 = \frac{1-\alpha}{2}, \quad \alpha_1 = \frac{1}{2}, \quad \alpha_2 = \frac{\alpha}{2}$$

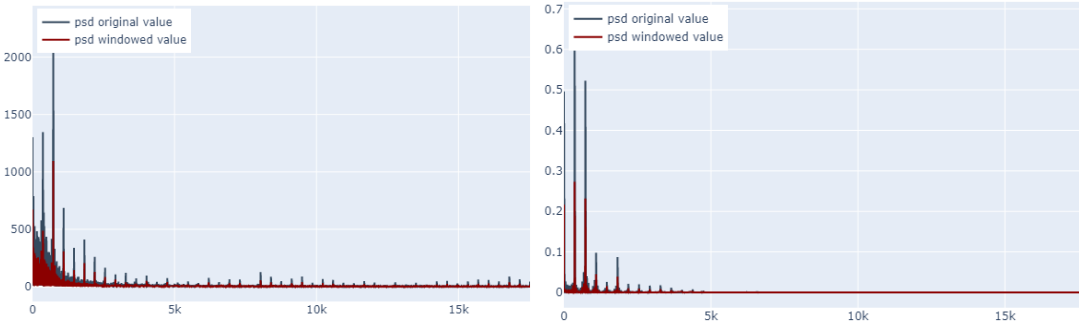
Spectral leakage can be minimized by applying a technique called windowing. "Windowing reduces the amplitude of the observations at the boundaries of the finite dataset. Windowing consists of multiplying the time record by a finite-length window with an amplitude that varies smoothly and gradually moves towards zero at the edges. This makes the endpoints of the waveform converge and, therefore, results in a continuous waveform without sharp transitions" (n.d.(b)). This thesis applies a blackman window, which is formulated by (5.2) (Blackman and Tukey, 1988) with a value of $\alpha = 0.16$ and N equal to the length of the dataset. Figure 5.4 displays the original time-series and one that is shifted by its median and has a blackman window applied to it. It is clear that the ends of the time series converge to zero and will therefore fit very smoothly to each other.



(A) (B)

FIGURE 5.4: plots of historic behaviour of MCF's (A) and profiled fractions (B) represented as raw data (gray) and windowed data (red)

Intuition and literature both indicate the presence of seasonality in the data. Three underlying seasonal components can generally be observed from electricity demand data; intraday, intraweek and intrayear cycles (Taylor, 2003). However, the time components are extracted from the data by means of the profile fractions. The MCF may therefore either lack seasonal components or exhibit different seasonal patterns that have not been extricated by the profile fractions. To verify the academically recognized frequencies in the data, a Fourier analysis is also applied on the profiled fractions.



(A) (B)

FIGURE 5.5: FFT plot of MCF's (A) and profiled fractions (B) represented as raw data (gray) and windowed data (red)

Results of the FFT are visualized in Figure 5.5. The peaks in the frequency domain representation of the profiled fractions (Figure 5.5b) correspond almost exactly with known theory. It has its largest peak at $f365$, which corresponds to a daily fluctuation, daily patterns understandably show a considerable level of similarity. The second largest peak follows at $f730$ which constitutes a twice daily fluctuation. This can be explained by the daily pattern of energy usage which tends to have two peaks and two troughs in between. The last distinct peak in the frequency domain is located at $f1$ which corresponds to a yearly pattern. Other intuitive patterns such as a monthly or weekly fluctuation that would occur at $f12$ and $f52$ are not prevalent in the highest frequency peaks. These effects are presumably already covered by the yearly and daily fluctuations.

The MCF displays other frequency patterns (Figure 5.5a), the daily and yearly peak in frequency is significantly smaller and seems to be at least partially captured by the profiled fractions. The largest peak presents itself at $f730$, which suggests that the twice daily fluctuation is still prevalent even when time-related features are already accounted for by the profiled fractions. Another interesting peak occurs at $f12$, which introduces a monthly variation.

It can be concluded that most obvious seasonal patterns are already accounted for by the profiled fractions. This is also evident when comparing the historic time series of both the MCF and the profiled fractions in Figure 5.4. The profiled fractions display a clear seasonal pattern, whereas the time series of the MCF looks more random.

5.2.4 Correlation analysis

The pearson correlation parameter summarizes the strength of a bivariate relationship. A correlation analysis is applied to the variables in this study to investigate the interdependence of the variables. The connection between the independent variables and the MCF is specifically of interest to get a rough idea of the explanatory power of some of the variables.

Literature suggests the presence of a high correlation between weather factors such as temperature and system load. A large part of this correlation is presumably already captured by the profile fractions. The profile fractions are determined by the time indications throughout a year. Time of the year is subsequently a major determinant in the state of the weather. The part of the variation in weather that is not captured by the time of the year can still have a significant impact on the eventual system load, which will translate in a correlation between that variable and the MCF.

Pearson Correlation Coefficient

Size of effect	ρ	% variance
small	.1	1
medium	.3	9
large	.5	25

FIGURE 5.6: Cohen's effect

Table 5.3 displays the correlation parameters of five different non-time related variables by year. The value of the pearson correlation parameter varies between -1 (perfect negative correlation) and 1 (perfect positive correlation). Cohen's effect as described in Figure 5.6 can be utilized to interpret the correlation coefficients.

Table 5.3 reports the variables *Temperature* and *Radiation* as most strongly correlated with the MCF. Both have a negative sign, which is intuitive for radiation since high values indicate a large generation of solar power. The direct relationship between Temperature and MCF is less straightforward, high temperatures would induce electricity usage for devices such as airconditioning. However, it can be noted that the correlation between *Temperature* and *Radiation* is considered high according to Cohen's effect interpretation. Therefore, it seems that the data might be subject to some extent of multicollinearity, which complicates the evaluation of the individual effects of *Radiation* and *Temperature* on MCF.

Two other features that show a significant amount of correlation with those variables are *Price* and *Clouds*. *Clouds* has a positive correlation with MCF, cloudiness will impede radiation from generating solar power. *Price* seems to have no meaningful correlation with the MCF.

Throughout the years no distinct time trend of increasing or decreasing correlation can be distinguished. Whereas it is suggested by research and intuition that the importance of weather variables will increase over time.

Parameters	1	2	3	4	5
Entire dataset					
1. MCF	1				
2. Temperature	-0.217	1			
3. Radiation	-0.255	0.525	1		
4. Clouds	0.190	-0.045	-0.215	1	
5. Price	-0.016	-0.275	-0.183	0.063	1
Year=2019					
MCF	1				
Temperature	-0.133	1			
Radiation	-0.248	0.547	1		
Clouds	0.176	-0.062	-0.169	1	
Price	0.072	-0.278	-0.080	0.023	1
Year=2020					
MCF	1				
Temperature	-0.109	1			
Radiation	-0.294	0.541	1		
Clouds	0.207	-0.052	-0.229	1	
Price	0.115	-0.146	-0.262	0.153	1
Year=2021					
MCF	1				
Temperature	-0.183	1			
Radiation	-0.235	0.487	1		
Clouds	0.102	0.288	-0.185	1	
Price	-0.004	-0.174	-0.221	-0.075	1

TABLE 5.3: Pearson correlation statistics by year

The Pearson correlation coefficient is a measure of linear association, it is unable to capture non-linear relationships between the variables. The figures in Appendix D visualise the correlation in a scatter plot with a 3rd degree polynomial regression line. The regression line reveals a

5.3 Data preparation

The raw datasets contains 2.360.552 datapoints, of which 432.110 are duplicates that have the same values for *Datetime* and *Grid Area*. Duplicated entries can occur when the MCF's are corrected by ENTRNCE after they have already been reported and imported in the dataset. ENTRNCE reevaluates the MCF's periodically 5 and 10 days after reporting. Both the initial MCF and the corrected MCF's will be imported from the database. The most recently corrected MCF is kept in the dataset, the others are dropped which leaves a database of 1.928.444 points.

Grid areas with only a limited amount of underlying connections add a lot of undiversifiable risk to the data, this phenomenon was discussed in section 5.2.1. From the 17 grid areas that are in the dataset, 9 of them were selected for the predictive comparison analysis in this thesis. The selection was based on criteria related to the amount of active connections in the grid areas during the period of interest and the stability of the number of active connections. Due to the dynamic market and some firm specific events, the amount of connections during the selected period was subject to some radical changes. Some grid areas did not contain more than a few connections until the start of 2022 whereafter they suddenly served an above average number of connections. The threshold for the amount of connections was chosen to be 250.000, and the number of customers in a grid area had to be above this threshold at all times during the period of interest. The number of connections and the total amount of electricity exploited are heavily correlated. The choice to set a threshold on the amount of connections and not the amount of electricity stems from the desire for a diversifiable risk in the grid areas. The other areas are deleted from the dataset to prevent them from negatively affecting the predictive accuracy of the model. The new dataset now contains 1.067.124 observations. Table 5.4 provides an overview of the remaining grid areas.

TABLE 5.4: Selection of grid areas

Grid code	Name	Data range
871694830000000309	Enexis	01-10-2018
871689200000010161	Stedin	01-10-2018
871688520000076884	Enexis	13-11-2018
871687120000052782	Liander	01-10-2018
871688600000002202	Stedin	01-10-2018
871687400000002254	Stedin	01-10-2018
871685900000056162	Liander	01-10-2018
871690910000025589	Liander	01-10-2018
871687910000219120	Enexis	01-10-2018

90 % of the data is located between 0.5684 and 1.1763. The most extreme outlier is located 17.65 standard deviations away from the mean. The extreme values in the MCF recordings can often be explained by underlying variables. Moreover, the selection of grid areas is chosen carefully such that extreme values of MCF are based on a large underlying customer base. Additionally, there is a financial incentive for the company to be able to predict specifically high and low values of the MCF. Removing them from the data will have the opposite effect. Therefore, no single outlier values are eliminated from the dataset.

Appendix B summarises the variety of weather variables and the percentage of data that is available for the period of interest in this research. The weather stations that are matched to the grid areas contain sufficient data on the variables radiation and temperature. However, the variable *clouds* reports more than 25% of missing data, all concentrated at the end of the dataset. This variable is therefore dropped from the selection. The missing values on other variables are imputated by a K-nearest neighbours imputation algorithm, with k=5.

Weather variables are interpolated to obtain subhourly observations. The forecasted weather variables are collected in different units than the historic weather variables, they have to be converted. The weather variables are assigned to the grid areas by matching the coordinates of their weather stations to the coordinates of the zip codes in a grid area. The datasets are then merged on *Datetime* and location (*grid area*).

Each input variable has a different scale associated with it. Neural networks require the inputs of the model to be in the same range. Inputs of different scales could cause the model to assign greater influence to some of the inputs due to their original measurement scales. Another motivation for applying input normalization is related to the gradient problem. "The rescaling of the input within small ranges gives rise to even smaller weight values in general. This makes the output of the units of the network near the saturation regions of the activation functions less likely. Furthermore, it allows to set the initial range of variability of the weights in very narrow intervals." (Baeldung, 2020) This thesis standardizes the observations for predictive analysis by neural networks as follows:

$$z = \frac{x - \mu_x}{\sigma_x}$$

5.4 Training, Validation and Test datasets

5.4.1 Training data

Each grid area is equipped with a vast amount of data. A training dataset is the sample of available data that is used to fit the predictive model on. The quantity of data points that is necessary to train a machine learning based model depends on a wide range of factors. The algorithm needs sufficient data to be able to obtain an accurate idea of the complex relationships that exist between the data points. The complexity of the model, the training method and the diversity of the input factors are only a few examples of factors that determine the amount of data necessary to build a valuable predictive model. For time series applications a general rule of thumb is that the training data should contain at least more observations than the period of the maximum expected seasonality.

The maximum seasonality that presents itself in the FFT is a yearly seasonality. The data is subhourly, therefore the model for one grid area would need at least 365*96 datapoints. Each grid area has at least two years of recorded MCF's available.

A supervised learning problem requires the data to be labeled – that is, annotated with the observed outcome value. A model is then trained to detect and learn underlying patterns between the data inputs and the accompanying labels. The trained

model can be considered as a mapping function that will map input data into output values. Ideally, this model would generalize well beyond the training data. The ability of a model to perform well on previously unobserved data points is called generalization and it is one of the central challenges of machine learning. A common pitfall of machine learning models is that they have too much capacity, which often implicates that they will learn the data too well and overfit the dataset. Overfitting happens when a model learns the noise and random fluctuations that are present in the training data as if it represents structure in the data.

5.4.2 Validation Set

An overfitted model can be easily discerned, it performs exceptionally well on training data and poorly on new data. To evaluate the generalizability of a model, a trained model is often applied on a validation set. The validation set is used to evaluate the model's performance whilst training, performance on the validation set can be applied to optimally tune the hyperparameters of the model. Generally if the model overfits the data, the complexity of the model should be scaled down. Vice versa, if the model underfits the data, this can be a sign to add some more complexity to the model. An additional application of the validation set is to implement an early stopping mechanism, the performance on the validation set can be used to determine when a model should stop training.

The ultimate model minimizes the total error of the prediction. Where the total error is defined as the sum of the squared bias and the variance of the prediction. The bias describes the difference between the predicted and the desired outcome. The variance takes into account the fluctuations of the model, a model with high variance does not generalize well on unseen data. Variance increases with complexity whereas bias decreases with complexity. The optimal model attempts to balance bias and variance such that it minimizes the total error. A visual representation of this bias-variance tradeoff is provided by Figure 5.7

A rule of thumb for splitting the data into a train and a validation set is the 80:20 ratio, 20% of the data is put aside as the validation set and the remaining 80% serves as the training set. Since the dataset consists of time series data, it is important to maintain the order of the dataset when splitting it into training and validation sets.

The last 20% of the data will be selected as the validation set.

5.4.3 Test set

The model with tuned hyperparameters will be applied on the test set for the final evaluation of its performance, the reported performance in this research is derived from the test set. For the sake of clarity: the samples in the test set are not included in training or validation data.

In this research the models will be evaluated on a rolling forecasting origin where the test set covers 3 months of data through the whole year to evaluate performance throughout different time periods. In practice the model is a multi-step forecast that builds a new model each day, with a delay between the prediction date and the date of available training data ranging from one to two weeks. However, for model comparison considerations it is adequate to examine a longer test dataset, this will reduce computational tim tremendously. The influence of the delay in available training data will however be evaluated by considering a variation of delays ranging from one hour to one week.

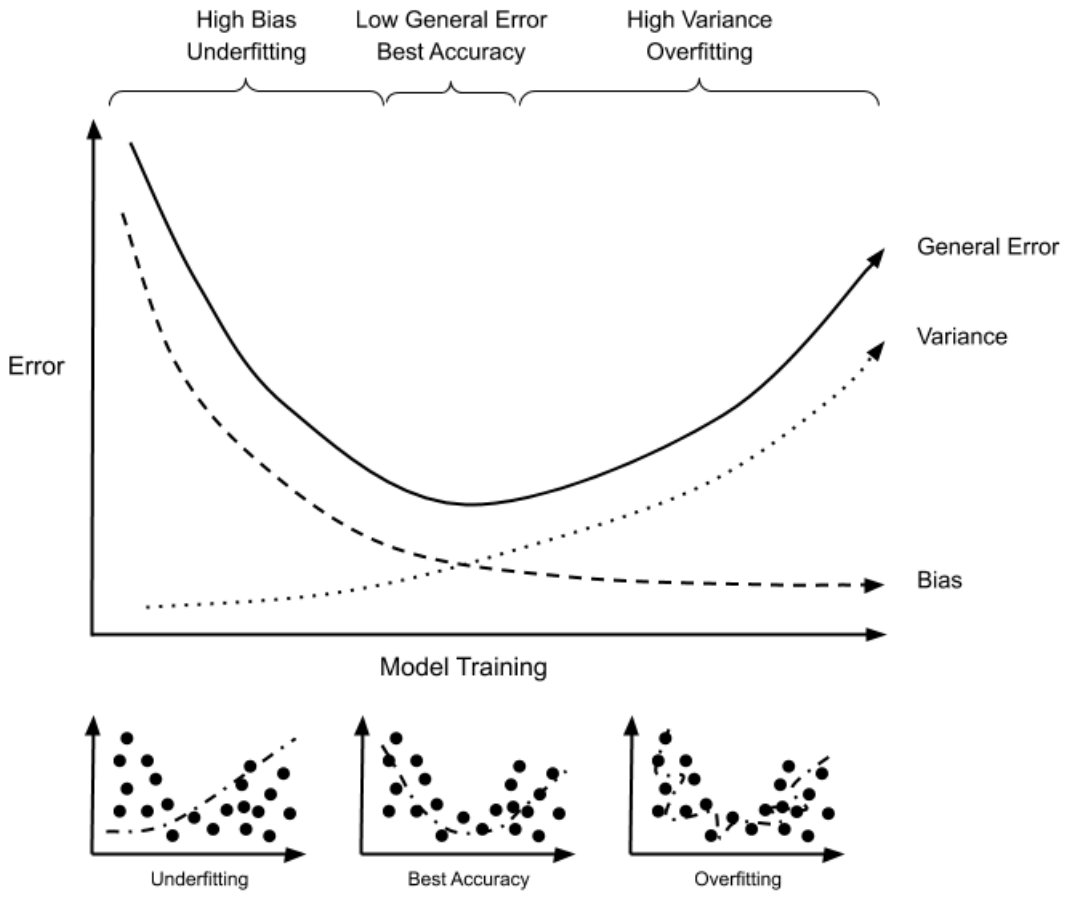


FIGURE 5.7: Visualisation of the bias-variance tradeoff

Chapter 6

Methodology

This chapter will describe the setup of the experiments conducted in this research. As mentioned in chapter 5, the reported performance of the models applied in this thesis is generated on a test set of samples via a rolling window method. The aggregated test set covers the period 01-04-2021 until 01-04-2022, this period enfolds an entire year to prevent an accuracy bias towards a specific season. The accuracy measures of the four test sets are averaged to obtain the final metrics. The 30 months preceding the test set will serve as training data. The dataset in this thesis will be split by a 8:2 ratio into a validation and training set. The validation set will be applied to tune the (hyper)parameters of the models and to monitor the number of epochs the neural networks uses to train. Figure 6.1 exhibits the four test sets and their respective training- and validation data.

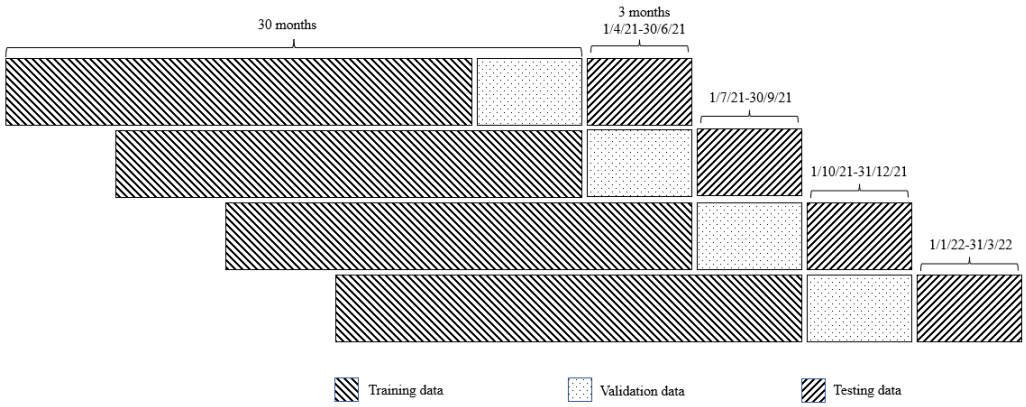


FIGURE 6.1: The rolling window method

The data in this thesis will be deployed to model time series using three different methods: ARIMA, ANN an LSTM. All methods will produce a univariate and a multivariate analysis. The univariate time series analysis only considers previous values of the variable of interest. A multivariate time series considers an additional set of dependent variables to incorporate in the forecasting model. ARIMAX is an extension of the ARIMA model that allows for the inclusion of exogenous variables. As acknowledged and discussed in Chapter 5, the exogenous variables of interest are temperature and radiation. The MCFs are recorded on a subhourly basis, this information is however made available only once per week. For theoretical and comparison purposes a direct forecast will be included in the results, this forecast will predict MCF's 15 minutes ahead. Since there is a significant delay in receiving the historical data up to approximately one week, this model can not be applied in practice. For this purpose a lag is introduced to the existing model. The models will

be evaluated for different values of $\tau \in [0, 4, 48, 96, 672]$, where τ symbolizes the delay in available data represented by number of periods (15-minute intervals). Data will be clustered geographically, which entails that for each grid a separate model will be trained. The error metrics will be averaged over the nine grid areas.

6.1 The benchmark model

The quality of the models in this thesis is benchmarked against a naive persistence forecast (NPF). The NPF uses the previous value of the time series as a prediction without any adjustment. This model is used as benchmark because it makes no assumptions on the data, is easy to understand and quick to implement. The performance of the NPF model will be recorded for all values of τ. For $\tau = 0$, the model will take the previous value of the MCF as a prediction, for $\tau = 96$ the model will shift the MCF 96 timesteps and take that value as a prediction.

6.2 The ARIMA model

The first model investigated is the ARIMA model, introduced in chapter 3. Models are implemented and evaluated using the *statsmodels* (Seabold and Perktold, 2010) and *sklearn* (Pedregosa et al., 2011) modules in python 3.9. Because of computational limitations, the models with $\tau > 0$ are clustered on time and date elements such that the dataset is divided into τ separate training sets which will produce τ ARIMA models

6.2.1 Stationarity

Chapter 3 discusses in detail the theoretical aspects and requirements of an ARIMA model. The Augmented Dicky-Fuller test (definition 3.1.2) is proposed to test the stationarity of the time series. The ADF test-statistic for every grid area in the selected dataset lies below -20, compared to the critical value at the 1% level of -3.43, this provides strong evidence against the null hypothesis. The null-hypothesis of non-stationarity can be rejected, it can be inferred that the series is mean-stationary. It can therefore be concluded that the differencing parameter of ARIMA can be set equal to zero (d=0). This results in an ARIMA(p,0,q) model, which is essentially the same as an ARMA(p,q) model.

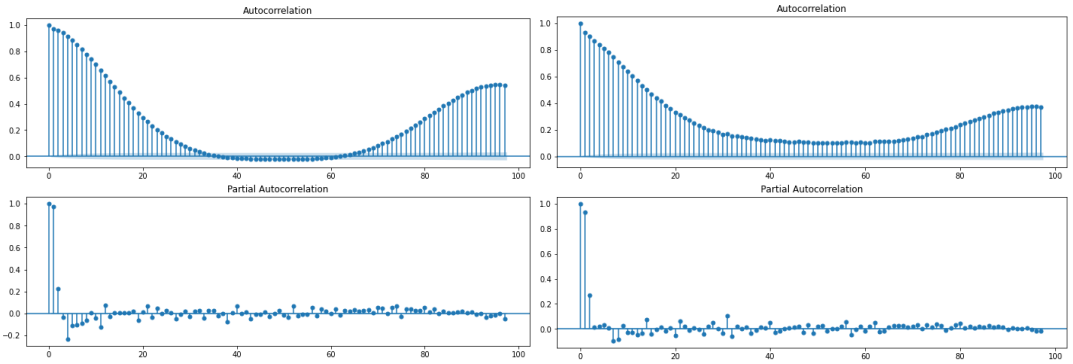
However, an increase in variance was evident from the time series visualisation in chapter 5. This violation of the homoskedasticity assumption will not be picked up by the ADF test-statistic. A logarithmic transformation (6.1) can help to stabilize the variance in the data. A constant is added to the dependent variable before transformation as a technique to deal with the negative values in the data.

$$\tilde{Y}_t = \log(Y_t + 1 - min(Y)) \tag{6.1}$$

6.2.2 Identification of parameters

Figure 6.2 plots the PACF and ACF for two out of the nine grid areas. The two grid areas behave relatively similar, but the plots show notable differences. A range of

suitable values for the parameters q and p can be selected. The partial autocorrelation function has two or three lags that seem significantly different from zero, a reasonable first guess would be to set p=2. For the ACF a sharp decline is observed between the second and fifth lag, so any value between those lags would be a candidate for q. The final order of the ARIMA model is determined by this intuition to be (2,0,3). Because all grid areas have similar plots, the same model is used for every grid area.



(A) (B)

FIGURE 6.2: ACF and PACF plots of two grid areas in the dataset

However the additional models that take into account a delay of τ require thorough consideration of the model architecture that is based on a complete dataset with no lags between the prediction and the inputs of the model. Therefore two other combinations of parameters are included and evaluated: the ARIMA(2,0,1) model and the ARIMA(1,0,1) model.

The ARIMA model is the representative of classical time series models in this thesis. The extension to an ARIMAX models allows for the extension to a multivariate forecasting technique. The technical details on the ARIMA model are extensively discussed in chapter 3

6.3 Neural Networks

The following models of interest are neural networks. First artificial networks are considered, inputs are varied to assess their influence on the accuracy of the predictions. For comparison purposes, it is preferred to maintain a framework that closely resembles the setup of the models discussed previously. The geographical clustering results in the construction of nine separate neural networks, whose architecture will be designed identically. The computational limitations of the ARIMA method necessitated a partition of the dataset in τ separate subsets, that all had their own ARIMA model trained. The neural does not have these limitations and can therefore implement a lag of τ on the entire dataset. To compensate for the additional information gained by training separate ARIMA models on subsets of the data, time variables will be added. The allocation of the variables will depend on τ. For a lag of 1 hour ($\tau = 4$), the variable *Minutes* will be added to the model, the lags of 12 hours and one day ($\tau = 48$ and $\tau = 96$) will additionally be assigned the variable *Hour*. Lastly the model predicting values one week ahead ($\tau = 672$) will also receive the variable *Weekday*.

6.3.1 Weight initialization

The weight initialization of neural networks is a procedure that determines the initial weights of a network. Assignment of constant weights will lead the network to learn the same features during the iterations of a training process. Therefore, neural networks with constant weight schemes will perform very poorly. Best practice 'to break symmetry', is to initialize the weights randomly. Model performance can vary with the weight allocations. Performance of a neural network is therefore often reported in terms of the mean and standard deviation of a number of simulations performed. Running a large number of simulations would produce reliable results that are generalizable, however it is time consuming and costly. Most important is that the different models in this research are suitable for comparison. Therefore, all models that incorporate random weight initialization are seeded such that results are reproducible and appropriate for comparison.

6.3.2 Architecture of the model

The model architecture was chosen by a grid search on the number of neurons and layers. Several layouts were tested, with the number of nodes ranging from 10 to 150 and the number of layers varying from 1 to 3. The grid search is evaluated on the validation data. The network architecture that performs on average the best on the validation data is saved and assigned to all grid areas to produce the accuracy metrics on the test set. One model is chosen for all grid areas, even though some grid areas would have benefited more from a different model design. The ARIMA models are also the same for all grid areas, therefore it is best practice to maintain this continuity.

The final ANN model has three layers and 120 nodes in each layer, the model architecture is represented by Table 6.1. The number of inputs n_x varies from 1 to 10, and the model always has one output. The number of parameters in each layer is determined by the number of nodes (or inputs) in the previous layer n_{l-1} and the current layer n_l in the following way: $n_{l-1} \cdot n_l + n_l$

LAYER TYPE	OUTPUT SHAPE	NUMBER OF PARAMETERS
Dense	(120)	1,320
Dense	(120)	14,520
Dense	(120)	14,520
Dense	(1)	121
Total		30,481

TABLE 6.1: Model architecture of ANN with 10 inputs

The LSTM model in this thesis is a hybrid model that stacks the dense layers from the ANN upon a LSTM layer. A dropout layer is implemented before the final dense layer to prevent overfitting. A dropout layer adds noise to the data by randomly setting inputs to 0 at a dropout rate, the dropout rate in this model is 0.2. Inputs not set to 0 are scaled up by 1/(1 - 0.2) such that the sum over all inputs is unchanged. The lookback period of the LSTM model and the number of neurons in the LSTM layer were chosen via a grid search to be 12 and 10 respectively. A lookback period of 12 was chosen out of a grid search that investigated lookback periods ranging from

12 to 96 in increments of 12. The results were very similar and a smaller lookback period is less computationally expensive.

Layer type	Output shape	Number of parameters
LSTM	(10)	840
Dense	(120)	1,320
Dense	(120)	14,520
Dense	(120)	14,520
Dropout	(120)	0
Dense	(1)	121
Total		31,321

TABLE 6.2: Model architecture of hybrid LSTM model with 10 inputs

6.3.3 Hyperparameters

The other hyperparameters of the model can also be chosen by means of a grid search. However since all the combinations of parameters have to be considered, adding an extra parameter to a grid search demands a lot of time and computational effort. Therefore hyperparameters in this thesis are chosen based on common practice and intuition. The ADAM algorithm is used as the gradient-descent based optimizer and all models optimize with the MSE as a loss function (Kingma and Ba, 2014). Furthermore, based on previous studies and common practice, all models use ReLU as activation layer. The batch size for all models has been set to 256. The number of epochs is determined by an early stopping mechanism that monitors validation loss with a patience of 10. Finally the learning rate is chosen to be 0.001. The parameters that are not mentioned in this section, such as the momentum and regularization parameters are kept at their default values.

6.3.4 Software

The programming language applied to obtain the results presented in the next section, is Python 3.9. Using Anaconda (version 2.1.4). The ARIMA models have been generated using statsmodels (Seabold and Perktold, 2010) and the neural networks were implemented using Keras (Chollet et al., 2015). Keras runs on Tensorflow as backend (Abadi et al., 2015). Multiple libraries and packages have been used during this research. An overview:

- pandas (McKinney, 2010)
- NumPy (Harris et al., 2020)
- plotly (Inc., 2015)
- matplotlib (Hunter, 2007)
- Scikit learn (Pedregosa et al., 2011)
- Keras (Chollet et al., 2015)
- Statsmodels (Seabold and Perktold, 2010)

Chapter 7

Empirical results

7.1 Performance Criteria

This thesis reports model performance evaluation based on a selection of forecast accuracy measures. There is no unique axiomatic approach for selecting the best error measure, because each error measure has its own drawbacks and advantages. One measure will be more suitable than another depending on the problem at hand and the assessment criteria. Error-based metrics are undoubtedly the most commonly applied in academic literature and among practitioners. They estimate the performance of a method by measuring the prediction error between the predicted and the observed values. In a practical setting these performance measures will often be considered in conjunction with some investment-based metrics such as the realized net return or the accumulated savings from a particular method.

According to a range of surveys the most commonly used error metrics are the: mean square error (MSE), root mean square error (RMSE), mean absolute error (MAE) and the mean absolute percentage error (MAPE). An early study by Carbone and Armstrong, 1982 reports that 30 out of 70 academicians prefer the MSE and RMSE, which has high correspondence with preferences of the practitioners reported in this survey. In a more recent study Fildes and Goodwin, 2007 reports that the MAPE is the most commonly used accuracy measure with 44.3% percent of 149 participants using this measure.

The following sections will discuss each metric in depth and treat its advantages and disadvantages. Assume that there are n samples of model errors $e_i, i = 1, 2, ..., n$. The model errors are defined as follows: $e_i = y_i - \hat{y}_i$ where y_i is the true observed value and $\hat{y}_i$ is the predicted value.

7.1.1 Mean absolute error

The Mean Absolute Error (MAE) (7.1) is the mean of all absolute errors, it averages the deviation from the predicted value to the actual value in real terms. The MAE applies the same scale as the underlying data, it is scale dependent and can therefore not convey information by itself. The MAE is thus meaningless in situations where one would want to compare the performance of two models that are applied on data of different dimensions. However when the underlying data is normalized or measured in the same unit it provides an easily interpretable indication of performance.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|e_i| \tag{7.1}$$

7.1.2 Mean Squared Error

The Mean Squared Error (MSE) (7.2) averages the square or all errors. The MSE decreases as the error approaches zero and just like the MAE it is a scale dependent quality measure. Whereas the MAE gives the same weight to all errors, the MSE penalizes large deviations from the true value by assigning them more weight than smaller deviations. The MSE incorporates both information about the bias and variance, and therefore contains more information than the MAE.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n} e_i^2 \tag{7.2}$$

7.1.3 Root Mean Squared Error

The Root Mean Squared Error (RMSE) (7.3) simply takes the root of the MSE. The MSE has the same units of measurement as the square of the underlying data, the RMSE therefore has the same units as the data. The RMSE is more suitable to be compared against the MAE because they have the same units of measurements. The RMSE varies with the variability of the the error magnitudes, the MAE and the sample size n. By definition the RMSE will never be smaller than the MAE, it tends to become increasingly larger than the MAE as the variability of the errors increases. The RMSE tends to grow larger than the MAE with a factor $\sqrt{n}$ since its lower limit is fixed at the MAE and its upper limit ($\sqrt{n}\times$ MAE) increases with $\sqrt{n}$ (Chai and Draxler, 2014). The greater difference between the MAE and the RMSE, the greater the variance in the individual errors in the dataset (Fenza et al., 2011).

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} e_i^2} \tag{7.3}$$

7.1.4 Mean Absolute Percentage Error

The Mean Absolute Percentage Error (MAPE) (7.4) measures the accuracy of a forecast system as a percentage of the actual value. The main distinction between the MAPE and the other error metrics previously mentioned is that the MAPE is a relative performance measure that is not dependent on the unit of measurement of its underlying data. The MAPE is therefore ideally suited to compare the obtained results with other studies. The fact that the MAPE is represented as a percentage or ratio makes it also easy to understand, it is a popular instrument among industry practitioners (Byrne, 2012). However the MAPE yields extremely large percentage errors (outliers), when the underlying data is very small (Kim and Kim, 2016). The variable of interest in this research is in fact on average smaller than 1 and often approaches zero, it can therefore be expected that this metric will be less suitable to use for evaluation purposes.

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n} \left|\frac{e_i}{y_i}\right| \tag{7.4}$$

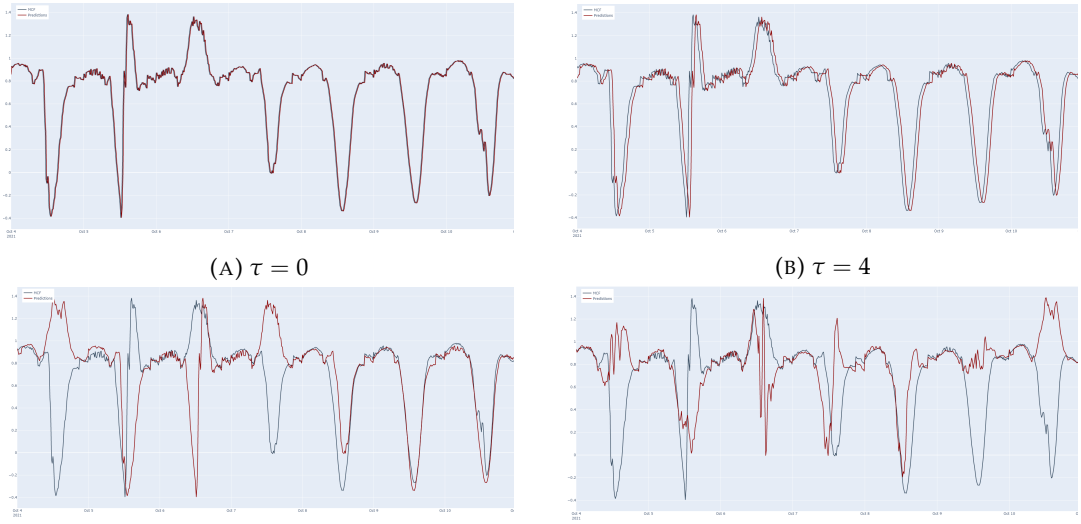
7.2 Empirical results Benchmark model

Table 7.1 presents the results of the naive persistence forecast. As expected, the performance drastically deteriorates as τ increases. The benchmark model delivers a strong performance for small values of τ, which suggests that prior values of the MCF hold a lot of information for the future values of the MCF. This seems inherent to the idea that the MCF is dependent on external factors such as temperature which are unlikely to change substantially during a small window of time. However as τ increases the window of time expands and the prior values further back in the past seem to hold less explanatory power. The increase of performance error in τ levels of as τ increases further, the effect is particularly strong for small values of τ, whereas a shift from $\tau = 96$ to $\tau = 672$ has less effect. The performance error displays a conspicuous spike at $\tau = 48$, which contradicts the periodicity that was implied by the FFT in Chapter 5.

τ	MAE	MSE	MAPE	RMSE
0	0.0339	0.0111	0.1240	0.1036
4	0.0827	0.0267	0.3342	0.1612
48	0.2657	0.1742	1.1573	0.4069
96	0.1637	0.1111	0.6628	0.3277
672	0.2092	0.1667	1.2726	0.4010

TABLE 7.1: Results of naive persistence forecast

Figure 7.1 visualises the NPF predictions and the true values of the MCF for a week in October 2021. The choice was made to not include visualisations for $\tau = 48$. From the graphs it becomes immediately clear that the MCFs do in fact exhibit periodicity, that is the reason why the NPF still delivers a reasonable performance for the larger time gaps.



(A) $\tau = 0$ (B) $\tau = 4$

(C) $\tau = 96$ (D) $\tau = 672$

FIGURE 7.1: Naive persistence predictions

7.3 Empirical results ARIMA model

This section reviews the results from the classical time series models that have been discussed in chapter 3

7.3.1 ARIMA results

The numerical results of the three ARIMA models are presented in Tables 7.2, 7.3 and 7.4. It is clear that all models produce results that are relatively close to each other in terms of accuracy, the ARIMA(2,0,3) model that was chosen based on the ACF and PACF plots performs competently for the complete data with $\tau = 0$. For the models that consider a delay in data, the ARIMA(1,0,1) seems to perform the best.

τ	MAE	MSE	MAPE	RMSE
0	0.0333	0.0128	0.1041	0.1004
4	0.0755	0.0260	0.3081	0.1532
48	0.1690	0.1769	0.4136	0.3246
96	0.1435	0.0755	0.3780	0.2565
672	0.1705	0.1142	0.3659	0.3136

TABLE 7.2: Results of ARIMA(2,0,3) forecast

τ	MAE	MSE	MAPE	RMSE
0	0.0322	0.0191	0.0732	0.1178
4	0.0751	0.0286	0.1666	0.1627
48	0.1551	0.0846	0.2503	0.2773
96	0.1619	0.0650	0.2204	0.3104
672	0.1690	0.1002	0.1965	0.3027

TABLE 7.3: Results of ARIMA(2,0,1) forecast

τ	MAE	MSE	MAPE	RMSE
0	0.0366	0.0162	0.1125	0.1133
4	0.0699	0.0259	0.2190	0.1492
48	0.1518	0.0800	0.2570	0.2603
96	0.1422	0.0734	0.4165	0.2534
672	0.1626	0.0900	0.3032	0.2813

TABLE 7.4: Results of ARIMA(1,0,1) forecast)

In comparison to the naive persistence model, performance gain from ARIMA models for the $\tau = 0$ dataset is negligible. No additional accuracy seems to be gained from learning regression coefficients on the time lags and including a moving average element. That predicates that a lot of the explanatory information is

contained in the first lag of the value to predict. The importance of learning dependencies in the data becomes apparent as τ increases, the RMSE of the ARIMA(1,0,1) model for $\tau = 96$ and $\tau = 672$ are substantially lower than those of the naive persistence forecasts. It was already clear from the NPF that, the further away the lag was located from the value to predict, the less information it contained about that value. This deficiency can thus be partially omitted by designing a model that can learn relationships between the lags and the moving average process in the data.

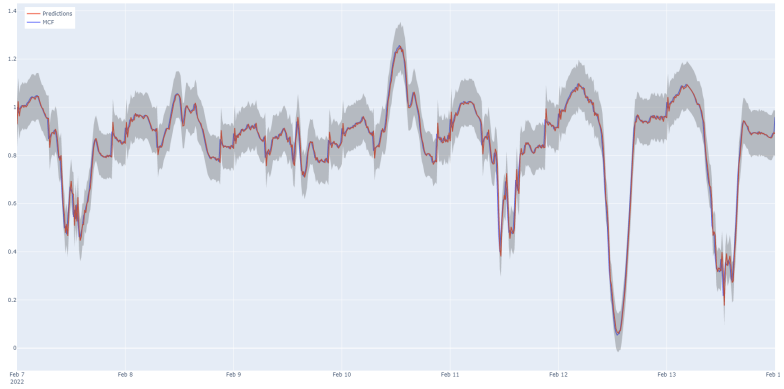


FIGURE 7.2: ARIMA model $\tau = 0$

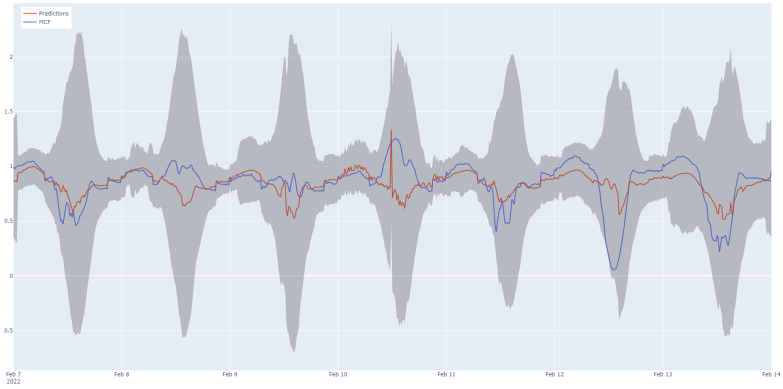


FIGURE 7.3: ARIMA model $\tau = 672$

Figure 7.2 displays both the predicted value (red) from the ARIMA(2,0,3) model for $\tau = 0$ and the actual values (blue) for a period in February 2022. The predictions almost exactly match the actual values. The grey bound is the 95% confidence region of the prediction, it covers a range that reaches a maximum at 0.2. Figure 7.3 studies the same time period, but now the predictions are produced by the ARIMA model that deals with a lag of $\tau = 672$. It is immediately clear that the observations deviate from the true values quite a lot. The confidence bounds are also extremely large, during the peaks and troughs of the MCF they cover a range of approximately 2. Which is 2 times as large as the confidence bounds in Figure 7.2

7.3.2 ARIMAX results

Perhaps even more interesting than the strong influence of the delay factor on the predictability of the model is the performance of ARIMAX models. The addition of exogenous variables to the forecast has been found to be very successful in research.

However in this study the addition of variables to the ARIMA model significantly deteriorates the performance of the model. This could be due to the the fact that the ARIMA model can only capture linear relationships. Chapter 5 already provided some insights into the mostly nonlinear relationships of the variables with the MCFs. Another explanation could be that there are some interactive effects in place that can not be picked up by the ARIMAX model.

7.4 Empirical results neural networks

This section reviews the results from the neural networks that were cited in chapter 4. Details about the specifics and architectures of the models in this section are provided in chapter 6.

7.4.1 Artificial neural network

The first set of neural networks are produced to compare against the ARIMA models. They take as inputs only previous values of the dependent variable and as mentioned in chapter 6, additional values that compensate for the clustering on time and day for different values of τ.

τ	MAE	MSE	MAPE	RMSE
0	0.0358	0.0107	0.1022	0.0997
4	0.0812	0.0241	0.2752	0.1519
48	0.1653	0.0849	0.2009	0.2800
96	0.1472	0.0780	0.2907	0.2791
672	0.1615	0.0897	0.1973	0.2995

TABLE 7.5: Results of ANN

Table 7.5 summarizes the average results of the networks described above. The results of the ANN are not significantly different from the error metrics obtained by the ARIMA models. In fact, when the results are compared against the optimal ARIMA model for that τ, the ANN slightly underperforms compared to the ARIMA models in terms of their RMSE. This difference is however negligible and most likely insignificant. Therefore, it can be concluded that given the previous values of the variable of interest, ARIMA and ANN performance is equal.

The second set of neural networks adds the set of dependent variables that was discussed in chapter 5. The variable price was found to significantly worsen the prediction results of the model. The disruptive effects of the price variable in the evaluation metrics might be caused by the extreme behaviour of prices during the test period. It was already mentioned in chapter 5 that in the years 2021 and 2022 prices skyrocketed and also experienced additional volatility. Since the last 9 months of 2021 and the first three months of 2022 are part of the test dataset, it is highly likely that the neural network receives values for the price variable that are extremely far from any value in the training set. For the sake of completeness, the results of this neural network are presented in Table 7.6. It is clear that the accuracy metrics are significantly worse than the previous ANN model and the ARIMA models. Going forward, the variable price will no longer be considered.

τ	MAE	MSE	MAPE	RMSE
0	0.0358	0.0109	0.1022	0.1046
4	0.0824	0.0241	0.2682	0.1551
48	0.1753	0.0845	0.2396	0.2906
96	0.1664	0.0769	0.2452	0.2773
672	0.1871	0.0932	0.2644	0.3054

TABLE 7.6: Results of ANN with exogenous variables including price

The model that adds only the variables *Radiation* and *Temperature* to the first ANN is evaluated in Table 7.7. These results are a substantial improvement from the ANN that included price as a variable. The error metrics are also significantly smaller than the error metrics of the ARIMA and ARIMAX models in the previous section. However improvement of the error metrics only happens for values of $\tau \geq 48$. Additional complexity of the prediction model does not capture any extra information for small time lags, neither does adding additional variables. Figure 7.4 visualizes the result of this network for $\tau = 672$, when compared to the plot of Figure 7.3 it is clear that the ANN model is better capable of tracking the movements of the actual MCF.

τ	MAE	MSE	MAPE	RMSE
0	0.0357	0.0109	0.1759	0.1026
4	0.0775	0.0229	0.2431	0.1493
48	0.1462	0.0672	0.3459	0.2565
96	0.1315	0.0629	0.2943	0.2480
672	0.1427	0.0703	0.2301	0.2627

TABLE 7.7: Results of ANN with exogenous variables excluding price

FIGURE 7.4: ANN model $\tau = 672$

Table 7.8 presents an overview of the results from an ANN model that incorporates additional information on the following calendar variables: *Month*, *Day Of the Year*, *Minutes*, *Hour*, *Weekday* and a binary variable on *Daylight savings time*. As mentioned before calendar information is already incorporated in profiled fractions, it would thus be unexpected for these additional variables to generate an improved performance. However from the results it appears that the models with large τ benefit in performance from these additional variables. This might be due to some interactive effects between weather variables and calendar variables. For instance a temperature of 20 degrees might have a different effect on MCFs in March than it does in June. A neural network is able to detect the nature of relationships between variables without being given a specific format of that relationship.

τ	MAE	MSE	MAPE	RMSE
0	0.0351	0.0109	0.1062	0.1023
4	0.0718	0.0212	0.1887	0.1433
48	0.1406	0.0594	0.2396	0.2408
96	0.1281	0.0568	0.2792	0.2347
672	0.1385	0.0631	0.3608	0.2485

TABLE 7.8: Results of ANN with exogenous variables and additional time elements

Figure 7.7 and Figure 7.6 display the predictions (red) and the actual values of the MCF for $\tau = 96$ and $\tau = 672$ respectively. The improvement of performance from this graph is not that obvious, however the error metrics measured over the entire year show a significant advancement.

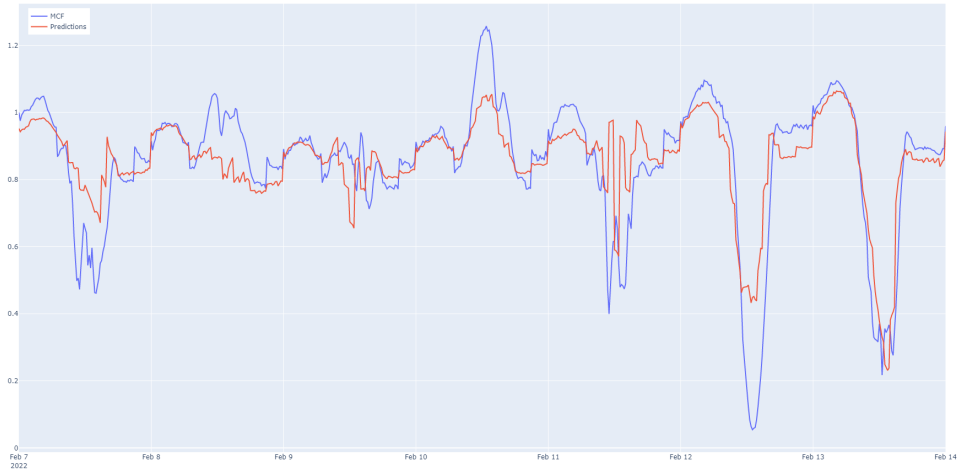


FIGURE 7.5: ANN model $\tau = 96$

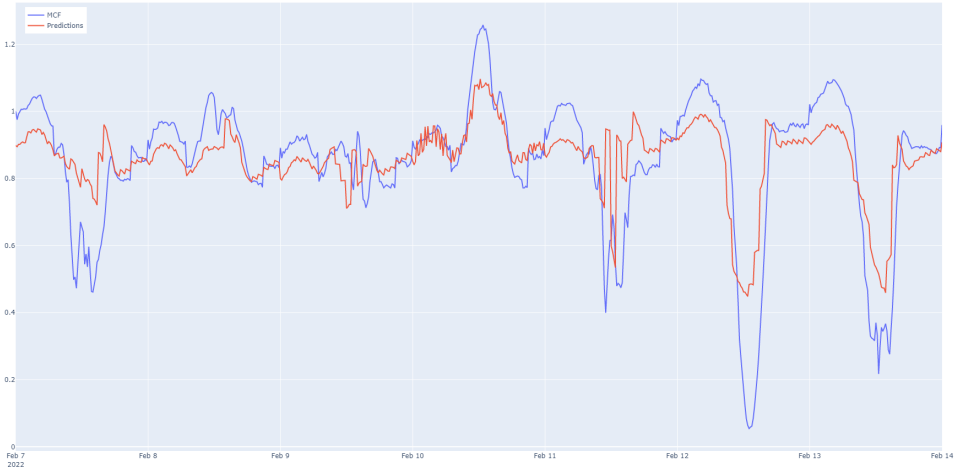


FIGURE 7.6: ANN model $\tau = 672$

7.4.2 Long Short-term Memory Network

Lastly this thesis presents the results of the LSTM model in Table 7.9. The LSTM extends on the best performing ANN model that includes exogenous variables and additional time elements. It is noteworthy to mention that this thesis applies a LSTM layer stacked on a general neural network and found that to increase performance. However, simple LSTM models tested in the grid search did not outperform the ANN model from the previous section. This finding contradicts previous research (Chandramitasari, Kurniawan, and Fujimura, 2018).

τ	MAE	MSE	MAPE	RMSE
0	0.0377	0.0132	0.1022	0.1122
4	0.0800	0.0207	0.1833	0.1417
48	0.1256	0.0474	0.3296	0.2157
96	0.1163	0.0466	0.3296	0.2791
672	0.1237	0.0511	0.2425	0.2242

TABLE 7.9: Results of LSTM

FIGURE 7.7: LSTM model $\tau = 672$

Chapter 8

Conclusion

This thesis explores an extensive comparison of various prediction models for short-term load forecasting. This thesis considers a measure correction factor as the variable to predict, which differs from other researches in the fact that direct effects due to calendar information is already isolated by the profiled fractions. Profiled fractions are constructed based on historical values. It is thus hard to compare the numerical results from this thesis to other studies in the field. Additionally, the MAPE is provided in the empirical section of this study but due to the small values of MCFs it might be hard to interpret.

First a theoretical framework was set up that covers the technical characteristics of each model. Several variations of each model are tested on a dataset containing historical demand information, weather information and calendar information. The baseline model against which all models are compared is a naive persistence forecast.

One of the sub-questions touches upon the effect of multistep-ahead forecasting, defining τ as the number of steps. The performance of the models is evaluated for a variety of timesteps. According to intuition, multistep-ahead forecasting proved more difficult as τ increases. Furthermore, simpler models such as the naive persistence forecast or the ARIMA models proved to be suitable for prediction of τ-step ahead forecasting, for small values of τ. Adding more complexity to the model setup did nothing for prediction accuracy, if anything, performance would deteriorate for more complex models such as ANN or LSTM models. Neither would the inclusion of additional variables. This suggests that most information about the MCF is contained in its direct previous values and can be transformed to its next value in a very straightforward way.

As τ increases, performance of all forecasting models deteriorates drastically. From $\tau = 4$ onwards there is a payoff from using more advanced models. The ARIMA model outperforms the naive persistence, and is in turn outperformed by the ANNs. This is in line with findings in recent literature as discussed in chapter 2. The neural networks in this thesis are optimized by means of a grid search.

The introduction of exogenous variables to the ARIMAX model causes accuracy metrics to explode. This is most likely caused by the inability of such models to pick up on interactions between variables and nonlinear relationships. Furthermore, the variable price is removed from the selection due to its explosive behaviour during the test period. The introduction of two exogenous weather variables into the ANN structure does improve the RMSE by approximately 10% (depending on τ). It can be concluded that weather variables possesses predictive power for the MCFs. This predictive power is also partially encompassed by lagged values of the MCF and thus the addition of those variables only becomes of service when direct lags are not

available.

Finally the LSTM models are evaluated, surprisingly the simple 'vanilla LSTM' and the stacked LSTM do not outperform the ANN models. Another interesting observation is the optimal lookback window which is set to be 12. This seems rather short considering that the LSTM models are specifically known for their capability to learn long-term dependencies. However the addition of a LSTM layer to the feed forward neural network does significantly improve the performance.

In conclusion, this paper showed that LSTM models are suitable models in short-term electricity consumption forecasting when using a measure correction factor. The main implication of this research is that additional complexity is specifically beneficial for τ-step ahead forecasting when τ is large. ANNs and LSTMs are suitable alternatives to traditional time series models such as ARIMA and can outperform the former based on RMSE metrics.

Appendix A

List of consumer profiles

TABLE A.1: Consumer profiles

Category	Min	Max	
E1A	-	$\leq 3 \times 25$	not remotely readable
E1B	-	$\leq 3 \times 25$	remotely readable and night tariff
E1C	-	$\leq 3 \times 25$	remotely readable and evening tariff
E2A	> 3 $\times$25	$\leq 3x80$	not remotely readable
E2B	> 3 $\times$25	$\leq 3x80$	remotely readable
E3A	> 3x 80	< 100kw	operating time $\leq$ 2000
E3B	> 3x 80	< 100kw	operating time > 2000, operating time $\leq$ 3000
E3C	> 3x 80	< 100kw	operating time > 3000, operating time <5000
E3D	> 3x 80	< 100kw	operating time > 5000
E4		<100kw	public lightning

Appendix B

List of weather stations

TABLE B.1: List of weather stations

WEATHER STATION	COORDINATES	COMPLETENESS		
		Temp	Rad	Clouds
209 IJmond	52.4N, 4.5E	0%	0%	0%
210 Valkenburg	52.1N, 4.4E	0%	0%	0%
215 Voorschoten	52.1N, 4.4E	99.9%	100%	72.5%
225 IJmuiden	52.4N, 4.5E	0%	0%	0%
235 De Kooy	52.9N, 4.8E	99.8%	100%	73%
240 Schiphol	52.3N, 4.8E	100%	100%	72.8%
242 Vlieland	53.2N, 4.9E	99.8%	0%	72.7%
248 Wijdenes	52.6N, 5.1E	0%	0%	0%
249 Berkhout	52.7N, 5.0E	99.9%	100%	0%
251 Hoorn Terschelling	53.4N, 5.4E	99.8%	99.8%	0%
257 Wijk aan zee	52.5N, 4.6E	99.9%	100%	0%
258 Houtribdijk	52.6N, 5.4E	0%	0%	0%
260 De Bilt	52.1N, 5.2E	99.9%	100%	72.9%
265 Soesterberg	52.1N, 5.3E	0%	0%	0%
267 Stavoren	52.9N, 5.4E	99.9%	100%	72.9%
269 Lelystad	52.5N, 5.5E	99.9%	100%	72.9%
270 Leeuwarden	53.2N, 5.8E	99.9%	100%	72.9%
273 Marknesse	52.7N, 5.9E	99.9%	100%	0%
275 Deelen	52.1N, 5.9E	99.9%	100%	72.9%
277 Lauwersoog	53.4N, 6.1E	99.9%	100%	72.9%
278 Heino	52.4N, 6.3E	99.9%	100%	0%
279 Hoogeveen	52.8N, 6.6E	99.9%	100%	73%
280 Eelde	53.1N, 6.6E	99.9%	100%	72.9%
283 Hupsel	52.1N, 6.7E	99.9%	100%	0%
285 Huibertgat	53.5N, 6.3E	0%	0%	0%
286 Nieuw Beerta	53.2N, 7.2E	99.9%	100%	0%
290 Twenthe	52.3N, 6.9E	99.7%	99.7%	72.6%
308 Cadzand	51.3N, 3.3E	0%	0%	0%
310 Vlissingen	51.5N, 3.6E	99.9%	100%	72.8%
312 Oosterschelde	51.7N, 3.6E	0%	0%	0%
313 Vlakte van De Raan	51.4N, 3.2E	0%	0%	0%
315 Hansweert	51.4N, 3.9E	0%	0%	0%
316 Schaar	51.6N, 3.6E	0%	0%	0%
319 Westdorpe	51.2N, 3.9E	99.9%	100%	0%

TABLE B.1: (Continued)

WEATHER STATION	COORDINATES	COMPLETENESS		
		Temp	Rad	Clouds
323 Wilhelminadorp	51.5N, 3.9E	99.9%	100%	0%
330 Hoek van Holland	52.0N, 4.1E	99.9%	100%	0%
331 Tholen	51.4N, 4.1E	0%	0%	0%
340 Woensdrecht	51.5N, 4.4E	99.9%	0%	72.8%
344 Rotterdam	52.0N, 4.5E	99.9%	100%	73%
348 Cabauw	52.0N, 4.9E	99.9%	100%	0%
350 Gilze-Rijen	51.5N, 4.9E	99.8%	99.9%	72.7%
356 Herwijnen	51.9N, 5.2E	99.7%	99.8%	0%
370 Eindhoven	51.5N, 5.4E	99.9%	100%	73%
375 Volkel	51.7N, 5.7E	99.9%	100%	73%
377 Ell	51.1N, 5.7E	99.9%	100%	72.9%
380 Maastricht	50.9N, 5.8E	99.9%	99.9%	73%
391 Arcen	51.5N, 6.2E	99.8%	99.8%	0%

Appendix C

List of grid areas

TABLE C.1: Information on grid areas

Grid code	Name	Data range
871692100000010038	Stedin	01-10-2018
871694830000000309	Enexis	01-10-2018
871689200000010161	Stedin	01-10-2018
871688520000076884	Enexis	13-11-2018
871690200000000007	Enduris	01-12-2018
871687800090000015	Westland	01-12-2018
871687120000052782	Liander	01-10-2018
871691280000000008	Rendo	01-07-2019
871688600000002202	Stedin	01-10-2018
871687400000002254	Stedin	01-10-2018
871694600000002173	Stedin	01-10-2018
871692510000000005	Stedin	01-10-2018
871690499910000003	Enexis	01-01-2019
871685900000056162	Liander	01-10-2018
871691600019188908	Coteq	01-01-2020
871690910000025589	Liander	01-10-2018
871687910000219120	Enexis	01-10-2018

Appendix D

Correlation plots

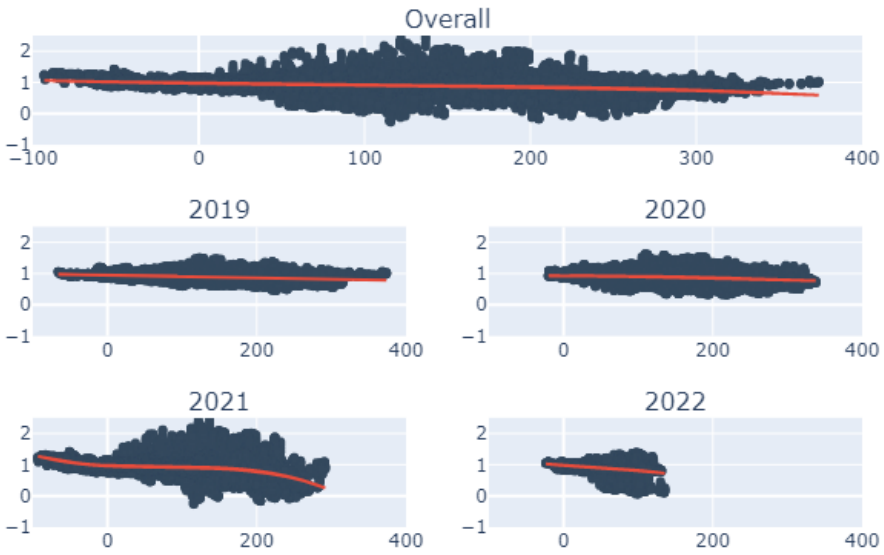


FIGURE D.1: Correlation analysis of Temperature on MCF by year

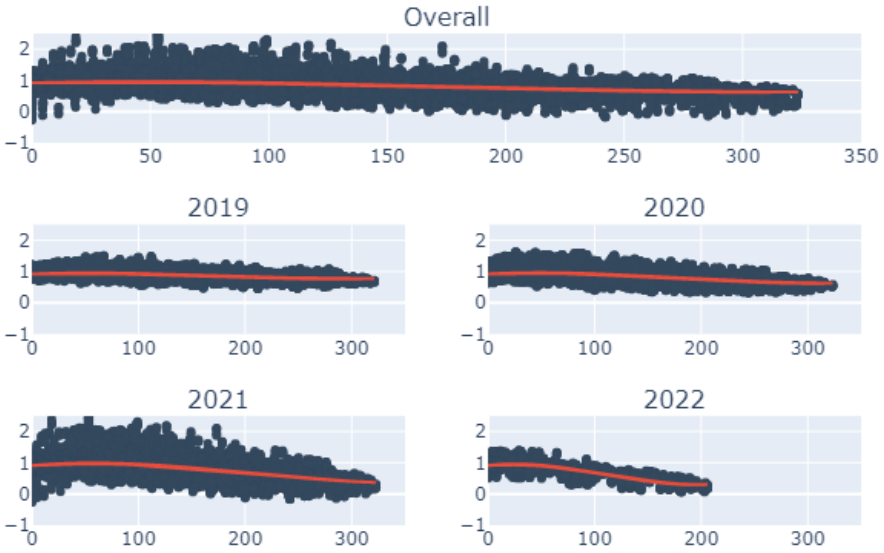


FIGURE D.2: Correlation analysis of Radiation on MCF by year

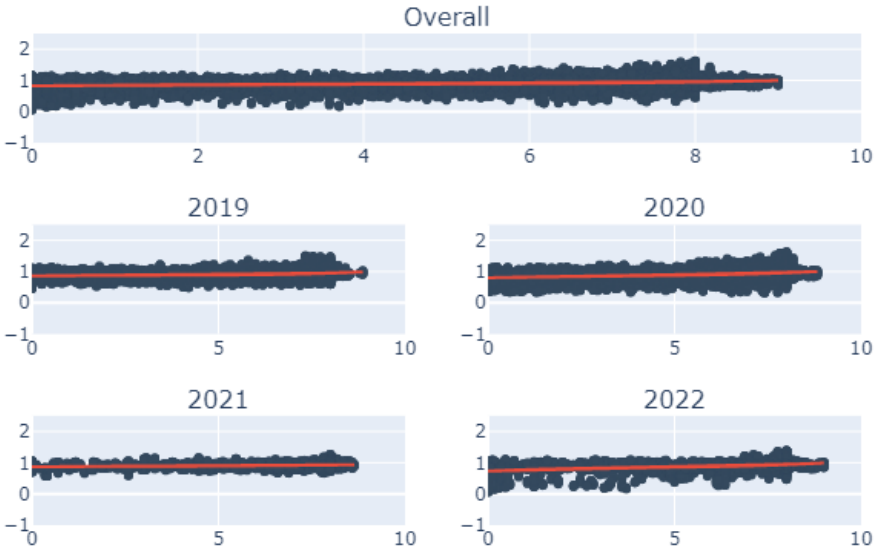


FIGURE D.3: Correlation analysis of Clouds on MCF by year

Bibliography

(N.d.[b]).

Abadi, Martín et al. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. URL: https://www.tensorflow.org/.

Al-Faris, Abdul Razak F (2002). "The demand for electricity in the GCC countries". In: *Energy Policy* 30.2, pp. 117–124.

Amjady, Nima (2001). "Short-term hourly load forecasting using time-series modeling with peak load estimation capability". In: *IEEE Transactions on power systems* 16.3, pp. 498–505.

Azoff, E Michael (1994). *Neural network time series forecasting of financial markets*. John Wiley & Sons, Inc.

Baeldung (2020). *Normalizing inputs of Neural Networks*. URL: https://www.baeldung.com/cs/normalizing-inputs-artificial-neural-network.

Benchmark of markets and regulations for electricity, gas and heat and overview of flexibility services to the electricity grid (2019).

Blackman, R. B. and John Wilder Tukey (1988). *The measurement of power spectra from the point of view of Communications Engineering*. Dover.

Byrne, Robert F (2012). "Beyond Traditional Time-Series: Using Demand Sensing to Improve Forecasts in Volatile Times." In: *Journal of Business Forecasting* 31.2.

Caire, P, G Hatabian, and C Muller (1992). "Progress in forecasting by neural networks". In: *[Proceedings 1992] IJCNN International Joint Conference on Neural Networks*. Vol. 2. IEEE, pp. 540–545.

Carbone, Robert and J Scott Armstrong (1982). "Note. Evaluation of extrapolative forecasting methods: results of a survey of academicians and practitioners". In: *Journal of Forecasting* 1.2, pp. 215–217.

Chai, Tianfeng and Roland R Draxler (2014). "Root mean square error (RMSE) or mean absolute error (MAE)?–Arguments against avoiding RMSE in the literature". In: *Geoscientific model development* 7.3, pp. 1247–1250.

Chakraborty, Kanad et al. (1992). "Forecasting the behavior of multivariate time series using neural networks". In: *Neural networks* 5.6, pp. 961–970.

Chandramitasari, Widyaning, Bobby Kurniawan, and Shigeru Fujimura (2018). "Building deep neural network model for short term electricity consumption forecasting". In: *2018 International Symposium on Advanced Intelligent Informatics (SAIN)*. IEEE, pp. 43–48.

Chollet, François et al. (2015). *Keras*. https://keras.io.

Contreras, J. and J.R. Santos (2006). "Short-term demand and energy price forecasting". In: *MELECON 2006 - 2006 IEEE Mediterranean Electrotechnical Conference*, pp. 924–927. DOI: 10.1109/MELCON.2006.1653249.

Council of European Union (2018). *Directive (EU) 2018/2001 of the European Parliament and of the Council of 11 December 2018 on the promotion of the use of energy from renewable sources*. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=uriserv:OJ.L_.2018.328.01.0082.01.ENG&toc=OJ:L:2018:328:TOC.

Dubey, Swapnil, Jatin Narotam Sarvaiya, and Bharath Seshadri (2013). "Temperature dependent photovoltaic (PV) efficiency and its effect on PV production in the world–a review". In: *Energy Procedia* 33, pp. 311–321.

Fahad, Muhammad Usman and Naeem Arbab (2014). "Factor affecting short term load forecasting". In: *Journal of Clean Energy Technologies* 2.4, pp. 305–309.

Fan, JY and JD McDonald (1994a). "A real-time implementation of short-term load forecasting for distribution power systems". In: *IEEE Transactions on Power Systems* 9.2, pp. 988–994.

Fan, J.Y. and J.D. McDonald (1994b). "A real-time implementation of short-term load forecasting for distribution power systems". In: *IEEE Transactions on Power Systems* 9.2, pp. 988–994. DOI: 10.1109/59.317646.

Faraway, Julian and Chris Chatfield (1998). "Time series forecasting with neural networks: a comparative study using the air line data". In: *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 47.2, pp. 231–250.

Fenza, G. et al. (2011). "A hybrid context aware system for tourist guidance based on collaborative filtering". In: *2011 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011)*, pp. 131–138. DOI: 10.1109/FUZZY.2011.6007604.

Fildes, Robert and Paul Goodwin (2007). "Against your better judgment? How organizations can improve their use of management judgment in forecasting". In: *Interfaces* 37.6, pp. 570–576.

Fitch, Frederic B (1944). "McCulloch Warren S. and Pitts Walter. A logical calculus of the ideas immanent in nervous activity. Bulletin of mathematical biophysics, vol. 5, pp. 115–133". In: *Journal of Symbolic Logic* 9.2.

Foster, Judith (2020). "Electric load forecasting with increased embedded renewable generation". PhD thesis. Queen's University Belfast.

Fuller, WA (1976). *Introduction to Statistical Time Series*.

G. Box, G. Jenkins (1970). *Time Series Analysis Forecasting and Control.*

Granger, Clive WJ and Paul Newbold (1975). *Economic Forecasting-the atheist's viewpoint*. University of Nottingham.

Gross, George and Francisco D Galiana (1987). "Short-term load forecasting". In: *Proceedings of the IEEE* 75.12, pp. 1558–1573.

Hagan, Martin T and Suzanne M Behr (1987). "The time series approach to short term load forecasting". In: *IEEE transactions on power systems* 2.3, pp. 785–791.

Harris, Charles R et al. (2020). "Array programming with NumPy". In: *Nature* 585.7825, pp. 357–362. ISSN: 1476-4687. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.

Hippert, Henrique Steinherz, Carlos Eduardo Pedreira, and Reinaldo Castro Souza (2001). "Neural networks for short-term load forecasting: A review and evaluation". In: *IEEE Transactions on power systems* 16.1, pp. 44–55.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long short-term memory". In: *Neural computation* 9.8, pp. 1735–1780.

Hornik, Kurt, Maxwell Stinchcombe, and Halbert White (1989). "Multilayer feedforward networks are universal approximators". In: *Neural networks* 2.5, pp. 359–366.

Hu, Caihong et al. (Oct. 2018). "Deep Learning with a Long Short-Term Memory Networks Approach for Rainfall-Runoff Simulation". In: *Water* 10, p. 1543. DOI: 10.3390/w10111543.

Hunter, John D (2007). "Matplotlib: A 2D graphics environment". In: *Computing in science & engineering* 9.3, p. 90.

Hwang, Sun Y and IV Basawa (2001). "Nonlinear time series contiguous to AR (1) processes and a related efficient test for linearity". In: *Statistics & probability letters* 52.4, pp. 381–390.

Inc., Plotly Technologies (2015). *Collaborative data science*. URL: https://plot.ly.

Kantz, Holger and Thomas Schreiber (2004). *Nonlinear time series analysis*. Vol. 7. Cambridge university press.

Karakurt, Izzet (2021). "Modelling and forecasting the oil consumptions of the BRICS-T countries". In: *Energy* 220, p. 119720. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2020.119720. URL: https://www.sciencedirect.com/science/article/pii/S0360544220328279.

Kim, Sungil and Heeyoung Kim (2016). "A new metric of absolute percentage error for intermittent demand forecasts". In: *International Journal of Forecasting* 32.3, pp. 669–679. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2015.12.003. URL: https://www.sciencedirect.com/science/article/pii/S0169207016000121.

Kingma, Diederik P and Jimmy Ba (2014). "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980*.

Lijesen, Mark G (2007). "The real-time price elasticity of electricity". In: *Energy economics* 29.2, pp. 249–258.

Liu, Chang et al. (2017). "Short-term load forecasting using a long short-term memory network". In: *2017 IEEE PES innovative smart grid technologies conference Europe (ISGT-Europe)*. IEEE, pp. 1–6.

Makridakis, Spyros et al. (1982). "The accuracy of extrapolation (time series) methods: Results of a forecasting competition". In: *Journal of forecasting* 1.2, pp. 111–153.

McKinney, Wes (2010). "Data Structures for Statistical Computing in Python". In: *Proceedings of the 9th Python in Science Conference*. Ed. by Stéfan van der Walt and Jarrod Millman, pp. 51 –56.

Minsky, Marvin and Seymour Papert (1969). "Perceptrons." In.

Mocanu, Elena et al. (2016). "Deep learning for estimating building energy consumption". In: *Sustainable Energy, Grids and Networks* 6, pp. 91–99.

Palomares-Salas, JC et al. (2009). "ARIMA vs. Neural networks for wind speed forecasting". In: *2009 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications*. IEEE, pp. 129–133.

Park, Dong C et al. (1991). "Electric load forecasting using an artificial neural network". In: *IEEE transactions on Power Systems* 6.2, pp. 442–449.

Pedregosa, F. et al. (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Peng, TM, NF Hubele, and GG Karady (1990). "Conceptual approach to the application of neural network for short-term load forecasting". In: *IEEE International Symposium on circuits and systems*. IEEE, pp. 2942–2945.

— (1992). "Advancement in the application of neural networks for short-term load forecasting". In: *IEEE Transactions on Power Systems* 7.1, pp. 250–257.

Rosenblatt, Frank (1957). *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory.

— (1961). *Principles of neurodynamics. perceptrons and the theory of brain mechanisms*. Tech. rep. Cornell Aeronautical Lab Inc Buffalo NY.

Rosenblatt, Murray (1956). "Remarks on some nonparametric estimates of a density function". In: *The annals of mathematical statistics*, pp. 832–837.

Rumelhart, David E, Geoffrey E Hinton, and Ronald J Williams (1986). "Learning representations by back-propagating errors". In: *nature* 323.6088, pp. 533–536.

Seabold, Skipper and Josef Perktold (2010). "statsmodels: Econometric and statistical modeling with python". In: *9th Python in Science Conference*.

Şen, Zekâi (2013). "Wind power variations under humid and arid meteorological conditions". In: *Energy conversion and management* 75, pp. 517–522.

Senjyu, Tomonobu et al. (2002). "One-hour-ahead load forecasting using neural network". In: *IEEE Transactions on power systems* 17.1, pp. 113–118.

Sharda, Ramesh and R Patil (1990). "Neural networks as forecasting experts: an empirical test". In: *Proceedings of the International Joint Conference on Neural Networks*. Vol. 2. IEEE, pp. 491–494.

Sharda, Ramesh and Rajendra B Patil (1992). "Connectionist approach to time series prediction: an empirical test". In: *Journal of Intelligent Manufacturing* 3.5, pp. 317–323.

Siami-Namini, Sima and Akbar Siami Namin (2018). "Forecasting economics and financial time series: ARIMA vs. LSTM". In: *arXiv preprint arXiv:1803.06386*.

Staffell, Iain and Stefan Pfenninger (2018). "The increasing impact of weather on electricity supply and demand". In: *Energy* 145, pp. 65–78. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2017.12.051. URL: https://www.sciencedirect.com/science/article/pii/S0360544217320844.

Statistics, Central Bureau of (2021). *Verbruik hernieuwbare energie*.

Taylor, James W (2003). "Short-term electricity demand forecasting using double seasonal exponential smoothing". In: *Journal of the Operational Research Society* 54.8, pp. 799–805.

Tennet (2022). *Imbalance Pricing System*.

Understanding LSTM networks (n.d.). URL: https://colah.github.io/posts/2015-08-Understanding-LSTMs/.

Valor, Enric, Vicente Meneu, and Vicente Caselles (2001). "Daily air temperature and electricity load in Spain". In: *Journal of applied Meteorology* 40.8, pp. 1413–1421.

Werbos, Paul (Jan. 1974). "Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Science. Thesis (Ph. D.). Appl. Math. Harvard University". PhD thesis.

Wold, Herman (1938). "A study in the analysis of stationary time series". PhD thesis. Almqvist & Wiksell.

WorldStandards (2021). *Complete lijst van gebruikte stekkers, netspanning frequentie per land*. URL: https://www.worldstandards.eu/nl/elektriciteit/landenoverzicht-stekkers-voltage/ (visited on 09/30/2010).

Yule, George Udny (1927). "VII. On a method of investigating periodicities disturbed series, with special reference to Wolfer's sunspot numbers". In: *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character* 226.636-646, pp. 267–298.

Zemouri, Ryad and Noureddine Zerhouni (2012). "Autonomous and adaptive procedure for cumulative failure prediction". In: *Neural Computing and Applications* 21.2, pp. 319–331.

Zhang, G Peter, B Eddy Patuwo, and Michael Y Hu (2001). "A simulation study of artificial neural networks for nonlinear time-series forecasting". In: *Computers & Operations Research* 28.4, pp. 381–396.

Zhang, Guoqiang, B Eddy Patuwo, and Michael Y Hu (1998). "Forecasting with artificial neural networks:: The state of the art". In: *International journal of forecasting* 14.1, pp. 35–62.

Zhang, Haotian (2014). "Smart Grid Technologies and Implementations". PhD thesis. City University London.

Zheng, Jian et al. (2017). "Electric load forecasting in smart grids using long-short-term-memory based recurrent neural network". In: *2017 51st Annual Conference on Information Sciences and Systems (CISS)*. IEEE, pp. 1–6.