

ΟΙΚΟΝΟΜΙΚΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΑΘΗΝΩΝ



ATHENS UNIVERSITY
OF ECONOMICS
AND BUSINESS

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

DEPARTMENT OF MANAGEMENT SCIENCE AND TECHNOLOGY

Short-Term Electric Load Forecasting Using Machine Learning & Neural Networks

Author:
Panagiotis Ntarzanos

Supervisor:
Prof. Panagiotis Louridas

A thesis submitted for the degree of

MSc in Business Analytics

January 20, 2020

This thesis is dedicated to Elder Athinagoras, for never giving up, on anything. Especially me.

Abstract

Load forecasting is an important component for power system's energy management procedure. Since in power systems the next days' power generation must be scheduled everyday, day-ahead Short-Term Load Forecasting (STLF) is a necessary daily task for power dispatch. Short-Term Load Forecasting has always been one of the most critical, sensitive and accuracy demanding factors of the power systems. Besides playing a key role in reducing the generation cost, an accurate STLF improves not only the whole system's economic viability but also its safety, stability and reliability in operation, in order to realize futuristic Smart Power System.

Electric load forecasting is an important tool, used to ensure that the energy supplied by utilities meets the consumers needs. Load forecasting has always been defined as the science of predicting the future load on a given system, for a specified period of time ahead. Underprediction of STLF leads to insufficient reserve capacity preparation and thus increases the operating cost by using expensive peaking units. On the other hand, overprediction of STLF leads to the unnecessarily large reserve capacity, which is also related to high operating cost.

Since precise load forecasting remains a great challenge, the research work in this master thesis is to develop efficient and accurate predictive models for electric load demand forecasting in short-term horizon (for 1-2 days ahead), concerning the Greek Electric Network Grid, by utilizing novel algorithms and up-to-date techniques. More specifically, we developed several models by using Machine Learning techniques and Recurrent Neural Networks, which are considered to be State of the Art algorithms for predictive analytics, not only for normal days but also taking account special days (e.g Holidays, Weekends etc). Validation of our research points out that these techniques generate better results in comparison with the currently used systems predictions. The input data that this thesis uses originate, from IPTO which stands for Independent Power Transmission Operator, and depict the historically hourly actual load of the Greek Electric Network Grid. Moreover, we also used historical load prediction data that Protergia's analysts declared in IPTO's load declaration platform, in order to be able to compare them with the thesis predictions and quantify the improvement.

Keywords: Machine Learning, Deep Learning, Time-series, Short-Term Electricity Load Forecasting, ARIMA, Prophet, LSTM, Random Forest, LightGBM

Acknowledgements

Foremost, I would like to express my sincere gratitude to my supervisor Prof. Panagiotis Louridas for the continuous support of my thesis, for his patience, motivation, enthusiasm and immense knowledge. His guidance helped me in all the time of research and writing of this thesis.

I would also like to thank the expert who was involved in the process of this research project Mr. Theodoros Paschos, Protergia's Energy Analyst, who provided me with the appropriate data used in this research. Without his passionate participation and input, this thesis could not have been successfully conducted.

Finally, I must express my very profound gratitude to my parents for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

Panagiotis Ntarzanos

Contents

1	Introduction	5
2	Electricity Market	7
2.1	The Energy Market Traditional Model	7
2.2	Liberalization of European electricity market	7
2.3	Balancing Market	8
2.4	Energy Market in Greece	9
2.4.1	RAE (Regulatory Authority of Energy)	9
2.4.2	HEDNO S.A. (Hellenic Electricity Distribution Network Operator S.A.) . .	12
3	Basic Concepts of Load Forecasting	14
3.1	Characteristics of the Electric Power System	14
3.2	Classification of Developed STLF Methods	15
3.3	Influencing factors of short-term load forecasting	16
3.3.1	Load Forecasting Techniques	17
3.3.2	Requirements of the STLF Process	19
3.3.3	Difficulties in the STLF	20
4	Time Series	22
4.1	Components of Time Series	22
4.2	Related Work and Models	23
4.3	Fundamental steps in forecasting procedure	24
4.4	Time Series Modeling	25
5	Time Series Forecasting using ARIMA & SARIMAX models	27
5.1	ARMA & ARIMA Models	27
5.2	SARIMA model	28
5.3	SARIMAX Model	28
6	Time Series Forecasting using the PROPHET algorithm	30
7	Time Series Forecasting using Neural Networks	33
7.1	Basic Concepts of Neural Networks	33
7.2	Long Short-Term Memory Units (LSTMs)	35
8	Time Series Forecasting using Random Forest	40
8.1	Basic Components of Random Forests	40
8.2	Random Forest Regression	42
9	Gradient Boosting Machines: LightGBM	46
9.1	Basic Concepts of Gradient Boosting	46
9.2	Gradient Boosting Decision Trees	47
9.3	LightGBM	48

10 Forecast Performance Measures	49
10.1 Basic Concepts of Prediction Accuracy	49
10.2 Forecast Performance Measures	50
10.2.1 MFE-Mean Forecast Error	50
10.2.2 MAE-Mean Absolute Error	50
10.2.3 MAPE-Mean Absolute Percentage Error	51
10.2.4 MSE-Mean Squared Error	51
10.2.5 SSE-Sum of Squared Error	52
10.2.6 SMSE-The Signed Mean Squared Error	52
10.2.7 RMSE-Root Mean Squared Error	52
11 Experiment-ML Models for STLF in Greek Electricity Network	53
11.1 Goals & Environment	53
11.2 Development Platform	54
11.3 Datasets	55
11.3.1 IPTO's Actual Load demand dataset	55
11.3.2 OoEM's Load Prediction dataset	56
11.3.3 Weekdays & Holidays	56
11.4 Preprocessing & Feature Engineering	56
11.4.1 Outliers-Missing Values	59
11.4.2 Summary Statistics & Data Visualizations	59
11.5 SARIMAX Model	63
11.6 Prophet Model	64
11.7 Multivariate Multi-Step LSTM Model	67
11.8 Time Series as a Regression Problem	69
12 Conclusion & Future Work	75
A Appendix	77
Bibliography	78

List of Figures

4.1	Weekly BP/USD exchange rate series (1980-1993)	26
4.2	Monthly internation airline passenger series (Jan 1949-Dec 1960)	26
6.1	Automated forecasting process with analyst-in-the-loop	31
7.1	The McCulloch-Pitts Neuron model	33
7.2	Artificial Neural Network's Architecture	34
7.3	Cost function and Gradient Descent	35
7.4	Unfolded basic Recurrent Neural Network	36
7.5	Recurrent Neural Network loop	36
7.6	Inner structure of an LSTM unit	37
7.7	LSTM Network Architecture	38
7.8	Data flow inside LSTM memory cell	39
8.1	Random Forest Structure	41
8.2	Random Forests pseudo-code	43
8.3	Random Forest Regression Algorithm	44
9.1	Splitting data: Leaf-wise VS Level-wise	48
11.1	Actual LV VS OoEM's LV Adjusted Forecast	59
11.2	Low Voltage Load Demand vs Holiday	60
11.3	Low Voltage Load Demand vs Weekday	60
11.4	Low Voltage Load Demand Boxplot	60
11.5	Daily Low Voltage Load Demand per Hour	61
11.6	2018 Low Voltage Load Demand (Historical Data)	62
11.7	SARIMAX optimal model summary	63
11.8	SARIMAX LV load forecasts vs Actual load vs OoEM's load forecast	64
11.9	PROPHET's components	66
11.10	PROPHET LV load forecasts vs Actual load vs OoEM's load forecast	66
11.11	Multi-Variate & Multi-Step LSTM Model Summary	67
11.12	LSTM Model's Train & Validation Loss	68
11.13	LSTM model's load forecasts vs Actual Load	69
11.14	Auto-correlation plot over 24 hour period	70
11.15	LightGBM LV load forecasts vs Actual load vs OoEM's load forecast	71
11.16	Actual LV load demand vs OoEM's LV load forecast	72
11.17	Actual LV load demand vs LightGBM model's LV load forecast	73
11.18	LightGBM LV load forecasts vs Actual load vs OoEM's load forecast	73
11.19	LightGBM LV load forecasts vs Actual load vs OoEM's load forecast	73
11.20	LightGBM LV load forecasts vs Actual load vs OoEM's load forecast	74

Chapter 1

Introduction

Electrical energy is a type of energy that needs to be transmitted in a fast, high quality and efficient manner worldwide since storage of the electricity energy generated from coal, natural gas, nuclear and renewable energy sources (wind, solar etc.) is not possible. With increasing population, urbanization, industrialization, and especially the rapidly developing technology the need for electrical energy is increasing gradually. Furthermore, the impact of increasing energy demand on the economy has been recognized thoroughly and, the studies on producing electricity steadily, sustainable electricity sources and supply-demand balance continue without slowing down. The constant increase in the requirement for electricity, the limited availability of resources and the lack of storage of electricity force the sector participants to prepare various action plans and safety measures. The necessity of consuming electricity at the moment it is produced imposes the provision of supply-demand balance to market actors.

Electrical power system basically comprises of a generation, transmission, distribution and utilization system. The major operation and maintenance cost is in the production sector, followed by the distribution sector. In practice the economic importance of the distribution system is huge and the amount of investment involved, demands extensive planning, design and operation. The planning problem is an attempt to minimize the cost of sub-transmission systems, substations, feeders, laterals, and so on, as well as the cost incurred in the losses.

In recent years, emphasis has been placed on balancing supply and demand, especially in economic terms, due to rapid increase in electricity demand. On the other hand, environmental factors or national regulations such as dependence to external resources for electricity generation leads to obligations concerning savings in electricity consumption. Considering all these aspects, planning activities are carried out in production, transmission and distribution systems to prevent shortages or surpluses in the electricity market. Therefore, accurate forecast of electricity load demand is important for the operation of electricity power systems, energy suppliers, consumers and policymakers for control, planning and regulation purposes. Thus, load forecasting is an indispensable section of designing, planning and operation of electric utilities, and moreover it is necessary to allocate considerable amount of electric energy in order to increase the economy significantly by electric power manufacturers. The electricity load forecasting, is a way of estimating what future electric load will be for a given perspective based on the available system information. Concerning the bigger picture, there are numerous areas in power system energy management that require efficient and accurate load forecasting models, such as:

1. **Distribution System Planning:** Distribution system planning is a very essential task, concerning the growing demand for electrical energy by additions of technically capable and reasonably economical distribution system units. The main objective of the distribution system planning is to ensure that the increasing load demands and high load densities are supplied the optimal way by the additional distribution systems. Distribution system planning should also consider the load magnitude and its geographical location. Concerning STLF, efficient and accurate load forecasting is a very critical factor to a cost effective distribution system planning. The system planning decisions greatly depend on the spatial load forecasting that gives the growth pattern of the load pertaining to a time, place and quantity.

2. **Distribution System Expansion:** The distribution system expansion is greatly influenced by the load growth in a particular geographic area served by a particular utility. Hence, an accurate forecast of the load growth and its effect on the system performance is very important for the expansion process.
3. **Operation and Maintenance:** A Smart Grid's power system regular operation and maintenance, is highly simplified by an accurate load forecast. It helps in guiding the personnel in making the appropriate switching and loading decisions, pertaining to the load profiles obtained during the load forecasting process.
4. **Financial Planning:** Accurate electric load forecasting, helps the system planning executives in making appropriate financial decisions related to the expansion of network, approvals of budgets, human resource management, technological upgrades etc.
5. **Load Management:** The electrical power supply utilities, try to meet all the customer demands for electrical energy whenever that demand occurs. Due to the stringent financial constraints because of the high cost of labor, materials and interest rates, very significant environmental concerns and the ever increasing shortage of fuels, the utilities are seriously diverting towards the option of load management as an alternative to capacity expansion to the extent possible. Effective load forecasting forms a genuine basis for the efficient load management.

Chapter 2

Electricity Market

The European Union energy market in general and the electricity market more specifically, has undergone huge changes in the last decade. Since mid-2007 when the last provisions of the second liberalization directives came into force ("Parliament Council Directive concerning Internal market in electricity's common rules", 2003), the EU energy market is, at least on paper, completely liberalized. Production and supply have essentially become market activities in which various participants compete for market shares, as well as trade in organized markets, such as energy exchanges that offer energy products, as well as derivatives. In truth, liberalization is an ongoing process, not without obstacles, which has so far reached different stages of implementation across EU member states.

2.1 The Energy Market Traditional Model

Traditionally, electrical power was supplied by state-owned companies, which covered all activities (production, transmission, distribution, and retail) along the supply chain and bore responsibility for supplying all customers of a certain geographic area. They were vertically integrated organizations, acting as autonomous Electric Power System (hereinafter EPS) in their own right. These "traditional versions" of a Transmission System Operator (hereinafter TSOs) centrally dispatched production, forecast the cumulative load (company-wise) and aim to balance the deviations between production and consumption. As these companies had a market monopoly, they were not motivated to reduce production costs or to introduce innovative technologies. All costs were reflected in the end consumer's KWh price.

2.2 Liberalization of European electricity market

Starting in 1999, the EU initiated a process of market liberalization. In 2009, the latest round of market legislation was adopted with the codification of the "Parliament and Council Directive concerning internal market in electricity's common rules" (hereinafter "Directive 2009/72/EC"). The directive, also known as the Third Energy Package, introduced legislation that aims gradually to liberalize the energy market. This will increase competition, allowing new players to join the sector, and, ultimately, liberalize energy prices. Its ultimate goal is to establish a competitive, reliable and sustainable supply of energy that will be driven by innovation and new technologies. Serena (2014) studied the historical motivation for introducing this legislation, the challenges of its introduction and issues/benefits concerning consumers.

The EU is aiming to reduce the concentration of power in the energy market, which it intends to achieve by separating the ownership of the channels of production, sales, distribution, and transmission (unbinding); the ownership of these companies will be private. There is also a separation within the same activities (vertical separation) where competition in the wholesale market allows for new producers to enter the market and new business models to emerge in the retail market. Production and sales have become commercial activities as the TSO and Distribution System Operators (hereinafter DSOs) have become independent and regulated. Across EU member states the situation regarding DSOs varies extensively; some member states have hundreds of DSO's, whereas others have only one or two. The transmission system is accessible to all players under

the same conditions. For regulation purposes, every EU member state establishes its own independent National Regulatory Authority. These cooperate within the framework of the Agency for the Cooperation of Energy Regulators (hereinafter ACER) (Market legislation, 2009).

The new situation in the market is already enabling new business models, which are competing with traditional ones. Investments into energy infrastructure are being made at an increased pace, with the objective of increasing cross-border trading and access to diverse sources of energy. A European Internal Electricity Market is envisioned, but not yet implemented. Currently, electrical energy produced, is traded at various regional electricity markets, such as:

1. **Long-term market:** Standardized products of electrical energy are traded for up to three years in advance. The products offered are weekly, monthly, quarterly and yearly products of base, peak, and off-peak energy. Retailers buy most electrical energy at these markets, in order to avoid price risk.
2. **day-ahead market:** Insufficiently bought electrical energy, according to a DSO's prediction is bought a day in advance, at elevated prices but in order to minimize loss.
3. **Intra-day market:** Similar to the day-ahead market, but with the difference that energy is bought at very short notice and delivered one or two hours later. The prices are obviously even higher.
4. **Balancing market:** A TSO settles imbalances between demand and supply after markets have closed (gate closure). This happens in and near real time. The day-ahead and intra-day markets are examples of spot markets. All the above-listed markets are physical markets where the commodity is physically delivered. Apart from these markets, there are derivatives markets where prices of the derivative instruments depend on prices of the underlying electricity products. The reasons for trading can be to hedge electricity price risk, or even for speculative reasons.

2.3 Balancing Market

Electricity as a commodity is totally different from other goods that are being traded. Since there are still no reliable technologies available for storing large quantities of energy ("bulk storage"), energy has to be consumed as soon as it is produced. Power demand and supply have to remain essentially balanced in real time for the grid frequency and the whole power grid to remain stable. Even the smallest deviations from the prescribed 50 Hz may result in system instability and occasionally even black-outs. Therefore, market players are subject to very strict balancing rules, and inability to fulfill the requirements is heavily penalized. Furthermore, the inclusion of Renewable Energy Sources (hereinafter RES), such as solar and wind generation, into the grid represents new challenges for network operators' daily balancing activities. RES display intermittent behavior, which affects flows within the grid, making it difficult to ensure the security of supply, to guarantee the availability of an adequate reserve margin to cope with unforeseeable events and to ensure that electricity markets function in a proper way.

TSOs have a legal obligation to keep the grid frequency constant at all times. Frequency is regulated by means of automatic and manual regulation mechanisms. The TSO holds in store active power capacity made available by producers, which can be activated at any time in order to bring balance into the grid. There are several levels of reserves, and they differ by function and response time. In the EU, the primary reserve is intended to stop the frequency drift in the case of unpredictable major events (for example a power plant going down). The secondary reserve brings the frequency back to its nominal value. The tertiary reserve is able to solve longer-term imbalances, which last up to a few hours.

The secondary and tertiary reserves are part of a larger network safeguard called ancillary services. Before market liberalization, ancillary services were the only way to balance the demand and supply of electrical energy. The overall balancing services' procurement costs were naturally transferred to the price the end users were paying, partly in the form of imbalance charges and partly in the form of network charges. The new legislation introduced a balancing market where demand and supply for balancing electrical energy meet. Trading is done by placing bids to buy and offers to sell energy. The origin of the balancing energy can be a power generating facility or demand facility (these will be discussed in the next chapter).

Currently, there are only national balancing/reserve markets in all EU states. A pan-European Exchange of Balancing Energy is envisioned, but, due to the scale and complexity of this task, a phased approach has been adopted. The first phase will be a coordination on a regional basis, followed by a merging of these regional balancing markets. Balancing cost varies by country and is dependent on multiple factors, such as market size, the share of RES in the grid, and so on. It is expected that the pan-European balancing market will benefit all participating countries as they will be able to balance supply and demand more accurately and also reduce their balancing costs.

2.4 Energy Market in Greece

In the previous sections, we had a thorough analysis of the EU electricity market and how it got -and keeps getting- liberalized. Although, concerning the fact that we are going to study load forecasting in the Greek Electricity Network Grid, we think that we should provide the current situation in Greek Energy sector and through the analysis of the current business architecture, we will try to identify the important role of accurate short-term load forecasting in Greek's electrical network grid energy management. The Greek wholesale electricity market has been organized as a pure mandatory pool since its inception in 2005. After gradual refinements, a transitional market design, implemented over a five-year period, was substituted on 30th September 2010 by its final provisional form. The revised market design, which is termed as the 5th Reference Day, reflects the full implementation of the 2005 Grid and Market Operation Code.

The Regulatory Authority for Energy (RAE) is an independent regulatory authority, established under Law 2773/1999, in the framework of harmonization with Directives 2003/54 / EC and 2003/55 /EC on electricity and natural gas, its responsibility to oversee the domestic energy market in all its fields, by recommending to the relevant state bodies and by taking measures itself to achieve the goal of liberalizing the electricity and natural gas markets. Under Law 2773/1999, and in particular with its subsequent amendments, the role assigned to RAE is mainly advisory while monitoring and controlling the energy market in all sectors, namely the production of electricity from conventional fuels, from renewable energy sources and natural gas.

Furthermore, RAE has assumed specific responsibilities in relation to the petroleum market. With the adoption of Law 3851/2010, there were substantial changes in relation to the existing legislative regime governing Renewable Energy Sources, as well as the responsibilities of RAE in this context. These changes concern both the licensing process for RES stations and the procedure for the assessment of applications for a production license.

2.4.1 RAE (Regulatory Authority of Energy)

More specifically, with regard to the licensing process, RAE has now taken a decisive role in the granting of production licenses, with the Ministry of Environment and Waters to exercise control over the legality of RAE's decisions, which was abolished under the provisions of Law 4001/2011. The role of RAE as a national energy regulator has been upgraded since 2011, with the enhancement and reinforcement of its decisive responsibilities for the regulation of the electricity and natural gas markets, responsibilities delegated to it at the request of the Third European Energy Bureau, which directs national energy regulators to "guarantors" of the proper functioning of energy markets. Specifically, on 3 September 2009, the European Union adopted the so-called Third Energy Package, consisting of Directive 2009/72 / EC, "on common rules for the internal market in electricity and repealing Directive 2003/54 / EC", Directive 2009/73 / EC, "on common rules for the internal market in natural gas and repealing Directive 2003/55 / EC", Regulation (EC) 713/2009, "Establishing an Agency for the Cooperation of Energy Regulators", Regulation (EC) 714/2009, "on conditions for access to the network for cross-border exchanges in electricity and repealing Regulation (EC) 1228/2003 ", Regulation (EC) 715/2009, "on conditions for access to the natural gas transmission networks and repealing Regulation (EC) 1775/2005 ».

Under EU law, European regulations are directly applicable by the Member States, without the need to incorporate them into national law. These Specific Regulations entered into force on 3 March 2011. European Directives 2009/72 / EC and 2009/73 / EC were transposed into Greek law by Energy Law 4001/2011, which entered into force on 22 August 2011 (Government Gazette A '179). This Law, among others, has redefined the nature and role of RAE so that it "is the national regulatory authority for electricity and natural gas within the meaning of Directives 2009/72 / EC and 2009/73 / EC "(Article 4 of the Act). Pursuant to the provisions of Law 4001/2011, RAE

has independent legal personality, as well as administrative and financial autonomy, and is charged with new, significantly increased, executive powers. According to Chapter C "RAE Competences", Part I of Energy Law 4001/2011, RAE's main, decisive responsibilities in electricity and natural gas are:

- **Market monitoring and supervision of the energy market:** RAE monitors and supervises the operation of the domestic energy market, prepares studies, compiles, publishes and reports, makes recommendations, decides or recommends to the competent bodies the necessary measures, including the adoption of regulatory and individual acts, in particular for the observance of the competition rules and regulatory obligations set out in Law 4001/2011, consumer protection, fulfillment of public service obligations, environmental protection, a range of energy supplies and the development of the European Union's internal energy market. To this end, RAE monitors and supervises the degree and effectiveness of competition in the domestic energy market, at wholesale and retail levels, the prices for home consumers, including prepayment systems, supplier switch rate, interruption rate, service provision and related fees, as well as customer complaints, the occurrence of distortions or restrictions of competition and restrictive contractual practices, such as exclusivity clauses that may prevent customers from entering into contracts simultaneously with more than one supplier, or limit the choice of supplier, the compatibility of the terms of electricity and natural gas contracts with the possibility of interruption, as well as long-term supply contracts, with national and European law and the observance of the specific regulatory obligations imposed on energy companies in accordance with the provisions in force and the conditions of the licenses granted to them. In the context of the above, RAE may issue non-binding directives and guidelines on matters falling within its sphere of competence and the manner in which it is exercised in order to ensure the correct and uniform application of the regulatory framework of the Energy Law and fuller information for stakeholders. RAE is also required to monitor the level of transparency, including wholesale prices, and to ensure that energy companies comply with their transparency obligations.
- **Consumer protection:** RAE supervises the implementation of consumer protection measures as defined in Part B of Energy Law 4001/2011. As regards complaints submitted by consumers against energy companies, they are the responsibility of the Authority only if they arise from or relate to regulatory oversight issues foreseen by the Energy Act and are specified in the regulatory decisions adopted under its authority, but not issues of dispute purely civil or commercial.
- **Monitoring the security of the country's energy supply:** RAE monitors the security of energy supply, in particular in relation to the supply and demand balance on the Greek energy market, the level of foreseeable future demand, the projected additional generation, transmission and distribution capacity of electricity and natural gas under planning or under construction, quality and level of maintenance and reliability of Transmission Systems and Distribution Networks, the implementation of measures to meet peak demand, and the conditions of gas supply in relation to the possibility of developing new production capacity. It also monitors the implementation of safeguard measures taken in the event of a sudden crisis in the energy market. Specifically, for natural gas, RAE has been designated as the Competent Authority to ensure the implementation of the measures set out in the European Gas Security Regulation 994/2010 of the European Parliament and of the Council of 20 October 2010 (L 295).¹⁰
- **Licensing:** RAE decides on the granting, modification and revocation of permits for the pursuit of energy activities in accordance with the specific provisions of the Energy Law, provided that the principles of transparency and equal treatment are respected, and taking into account the specific characteristics of applicants, consumer protection, protection of the environment and ensuring conditions of healthy competition. In particular, RAE shall, when granting permits, take all necessary action to comply with the projections of the Long-Term Energy Planning of the country, as well as any restrictions contained therein, or a binding text submitted by the Hellenic Republic to European Commission or International Organizations. RAE monitors and controls how to exercise the rights granted by these licenses, as well as compliance with the obligations of licensees.

- **Supervision of Independent Transmission Operators Certification:** RAE decides on the certification of natural gas and electricity companies in accordance with the criteria of the European Directives and the Energy Law in order for these companies to be designated as Transmission System Operators and monitors the continuous compliance of these Operators with the Transmission System Operators these criteria. While respecting the confidentiality of commercially sensitive information, the Authority may request any information and information from Transmission System Operators and undertakings that engage in any of the electricity or natural gas generation or supply activities.
- **Supervisory Board and Compliance Officer:** The placement or removal of the Independent Transmission Controller's Compliance Officer, as well as the terms of the relevant mandate they receive from the Supervisory Board of the Independent Transmission Manager, shall be subject to prior approval of RAE. In addition, the Authority shall verify whether the professional independence criterion is met by half of one member of the Supervisory Board and the Management Board of the Operator.
- **Program Development Monitor:** RAE decides on modifications to the Development Programs prepared by the relevant Transmission Managers, examining whether the Development Program: (a) covers all identified needs; and (b) complies with the corresponding, non-binding, 10-year development of the Transmission System for Electricity and Natural Gas, which is compiled in accordance with the European Regulations 714/2009 and 715/2009. RAE monitors and evaluates the implementation of the above Development Programs. It also monitors the time it takes for Transmission System Operators and Distribution Networks to make user connections, perform repairs, and provide service to system users and their Networks. It may also set deadlines on the above and penalties forfeited in favor of users in the event of non-observance of the deadlines.
- **Adoption of non-competitive invoices:** RAE decides on the methodology for calculating non-competitive business tariffs and the amount of such charges in such a way that these tariffs are non-discriminatory and reflect the cost of the services provided, taking into account the need for short-term and long-term incentives for Transmission System Operators and Distribution Networks to improve the efficiency of their networks, to promote the development of the energy market and the security of the supply.
- **Granting exemption from third party access obligations:** RAE decides to grant discharge of part or all of the natural gas System capacity and interconnections with other Electricity Transmission Systems in third countries, from the obligation to provide access to third parties or from the obligation to own ownership. To this end, it shall cooperate with the Regulatory Authorities of other Member States, any third State involved, the ACER, the Energy Community bodies and the European Commission. Monitoring access to energy interconnections RAE establishes, monitors and supervises the application of the access rules to interconnections, including the relevant tariffs and the methodology for calculating them, the capacity allocation and decommissioning and congestion management mechanism and the provision of balancing services, the out-of-court resolution procedure differences arising in the application of the above, as well as any other necessary detail. To this end, RAE shall request an opinion from the relevant Transmission System Operators. For this purpose, RAE cooperates with the Regulatory Authorities of other countries with which our country has an energy interconnection.
- **Taking regulatory measures for the proper functioning of energy markets:** RAE may impose on undertakings engaged in energy activities measures and conditions appropriate to the result sought which are deemed necessary to ensure the implementation of the provisions of the Energy Law and the existence of conditions of fair competition and orderly functioning of the market. From all the above, it is clear that the authority has a very wide range of responsibilities as well as the enormous amount of work that this entails. In order to effectively exercise its responsibilities and to successfully perform its regulatory work, the provisions of Law 4001/2011 provide RAE with specific, very important additional tools, such as the ability to collect all kinds of data, conduct investigations, examine complaints and the consequent imposition of sanctions, interim measures, etc.

2.4.2 HEDNO S.A. (Hellenic Electricity Distribution Network Operator S.A.)

In essence, HEDNO S.A. (Hellenic Electricity Distribution Network Operator S.A.) was formed by the separation of the Distribution Department from PPC S.A., according to L.4001/2011 and in compliance with 2009/72/EC EU Directive relative to the electricity market organization with the goal to undertake the tasks of the Hellenic Electricity Distribution Network Operator. It is a 100% subsidiary of PPC S.A., however, it is independent in operation and management retaining all the independence requirements that are incorporated within the above mentioned legislative framework. Company tasks include the operation, maintenance and development of the power distribution network in Greece, as well as the assurance of a transparent and impartial access of consumers and of all network users in general. We aim at providing reliable power supply to our Customers, quality of electricity voltage and constant improvement of quality in services.

More specifically, HEDNO has the absolute jurisdiction of implementing any new connection to the network, meter measurement counting, electricity interruptions, the repair of grid failures and the maintenance of the national electricity grid. In essence, the market design introduced a distinction between the day-ahead market and the balancing mechanism that follows, as in other countries with compulsory pools. This structure reflects with more clarity the factors influencing prices, the uncertainties involved and the implied risks at these distinct time scales. More specifically, during the transitory market regime, the Day Ahead market provided an indicative unit commitment schedule and a reference spot price (SMP forecast), which served purely as a signal. Cash-flows were based on ex-post SMP prices. These were derived by re-solving the same cost-minimization algorithm as in the day-ahead schedule by inserting metered values of the various inputs (mainly demand, plant availability and renewables output) instead of day-ahead forecasts. These ex-post prices were applied to the actual quantities consumed or produced (the latter reflecting to a large extent the real-time dispatch orders of the TSO).

As opposed to an overall market settlement (through ex-post SMP prices), the current market design involves two distinct settlement processes. The settlement of the day-ahead market, in which generators' payments (suppliers' charges) are computed, based on the SMP prices and the plant schedules derived from the day-ahead dispatch (load declarations submitted). The settlement of imbalances, in which deviations from day-ahead schedules are charged or compensated, depending on whether they are exogenous or reflect the TSO's dispatch orders. There is also a provision for imbalance penalties, if certain limits are violated regarding the magnitude and the frequency of the deviations. In the day-ahead market, uniform pricing still applies, reflecting the offer of the most expensive unit dispatched so that predicted demand is satisfied.

Zonal pricing, intended to reveal congestion problems and signal the location of new capacity, has not been activated yet, although two zonal prices, applicable to generators, are explicitly derived, currently only as an indication. Participants may enter into bilateral financial contracts (CfDs), but physical delivery transactions are constrained within the pool and related contracts do not exist. A cap of 150 €/MWh has been imposed on generators' offers. There are also supplementary mechanisms and rules that still apply. A lower limit is imposed on generators' offers, equal to the minimum variable cost of each unit in each trading period, as -in the current structure- the incumbent has a strong incentive to suppress wholesale prices. A cost-recovery mechanism ensures that generators dispatched by the TSO, beyond the day-ahead schedule, are remunerated based on their declared minimum variable costs plus a 10% margin. This mechanism creates a safety net, which often makes participants rather indifferent to the price levels. A Capacity Adequacy Mechanism is applied for the partial recovery of capital costs, with suppliers being obliged to buy capacity certificates from generators. The value of these certificates was revised in November 2010 from 35.000 to 45.000 €/MW, in order to alleviate the impact of low demand on generators' revenues. Adjusting the value of the certificates based on the technical flexibility and environmental impact of each plant was also explored as a possible future refinement.

Regarding the market structure, PPC retained its dominant position over 2011 but its market share declined substantially, both in the generation and the supply side. In the generation sector, a significant change towards a less concentrated structure occurred in 2010, as two new IPP units began commercial operation, and this change was reinforced in 2011, with the addition of two other IPP plants. In terms of thermal capacity, this direction of market evolution is not expected to persist in the future, as all private plants have now been completed, and the new plants expected are owned by PPC. This would change however, if plant divestments, intensively discussed under Troika's pressure or alternative measures of PPC's capacity allocation are implemented in the

coming years. Apart from conventional generation, changes in market structure were enhanced by a steady, currently explosive, renewables' penetration, in which PPC's share is minor. Despite the substantial amount of capacity that had applied for connection in the past, the TSO estimated that due to the economic recession, various investment plans were cancelled, which seems a reasonable assessment. Electricity demand remained fairly stable over 2011 at 51872 GWh, exhibiting a minor decline of 0.94% relatively to 2010 at the interconnected system.

At the national level, demand was also unaffected, amounting to 61834 GWh (relatively to 61817 GWh in 2010). It is notable that demand recovered over the last quarter, exhibiting an increase of 2% relatively to Q4 2010, as air-conditioning was used quite intensively for heating, perceived by consumers as a less expensive option than oil. Given the addition of substantial capacity over two consecutive years, the market share of PPC declined significantly in 2011. In the interconnected system, PPC's share, in terms of volume, dropped to 75% of local production, relatively to 85.2010, while IPP's share more than doubled, PPC's natural gas production declined by 816 GWh (13.5%) It is interesting that despite this decline, the natural gas cost for PPC increased by €10 million due to an increase in natural gas prices of 18.4% and a €21 million tax contribution due to the levy imposed on natural gas. At the national level (including non-connected islands), PPC's production covered 70.1% of total demand in 2010, with the corresponding share being 77.3% and 85.6% in the previous two years. Subsequently, this market share was suppressed further, reaching 67.5% in Q1 2012. In absolute terms, PPC's production plus import activity was reduced by 4451 GWh, while in 2010 this quantity had already been reduced by 5123 GWh. The import activity of PPC was also reduced, by 18%.

Regarding renewables generation, PPC's production remained low (246 vs. 374 GWh), due to water scarcity affecting its small hydro units, while independent renewable production reached 3958 GWh. Wind parks connected to the high-voltage Grid yielded 2535 GWh. the renewables sector remains an attractive investment field, with the level of feed-in-tariffs being a major incentive, even if their levels are adjusted downward for new capacity. In addition, bureaucratic (licensing) obstacles are progressively simplified. Given the targeted level of wind penetration (7500 MW by 2020) and its intermittent nature, stand-by reserve as well as secondary reserve is expected to emerge as a significant component of financial returns for thermal plants. This reimbursement did not occur in 2011, as natural gas plants competed severely for reserve provision, to get dispatch and receive the cost-recovery payment, yielding reserve prices close to zero. Hence, ancillary prices are not independent of the supplementary mechanisms introduced in the market and cannot be corrected if viewed in isolation.

Chapter 3

Basic Concepts of Load Forecasting

The system load is the sum of all the consumers' load at the same time. The objective of system STLF is to forecast the future system load. Good understanding of the system characteristics helps to design reasonable forecasting models and select appropriate models in different situations.

3.1 Characteristics of the Electric Power System

Electrical power is produced by transforming energy, derived from various energy sources, such as nuclear energy, fossil fuels (coal, gas, and oil), into electricity. Alongside traditional power generating facilities, more recently the use of RES, such as bio-energy, hydro-power, wind, solar and geothermal energy, is constantly rising. Electricity is transmitted to the end consumer through the power grid. The first part of the network, called the transmission grid, operates at high and extra-high voltage and is dedicated to long-distance transmission. It is managed by Transmission System Operators.

The second part of the network, called the distribution grid, operates partly at high voltage, but mostly at medium and low voltages, and is dedicated to the regional distribution of electricity, supplying it to lower-level distribution systems and to directly connected customers. It is run by Distribution System Operators. The thresholds, determining the delimitation between a transmission system and a distribution system, are established at the national level. Both TSOs and DSOs (commonly called system operators) are legally bound to ensure the security and reliability of supply of their part of the system. The network's electrical components used to supply, transfer, and use electric power is called an Electric Power System also known as a power grid.

The objective of short-term load forecasting is to find out the load variation trend and predict future development by using correct prediction methods. The characteristics of short-term load forecasting are as follows:

1. **Uncertainty**

It is uncertain to know the future development of the load because the changes are related with a lot of factors that are also constantly developing and changing. Although several of these factors can be predicted, the remaining are difficult to predict, which makes our prediction results uncertain.

2. **Conditionality**

The future load change occurs under the necessary conditions and the assumed conditions when predicting it. The necessary condition is the ability to predict the essential rules of change in load, and prediction results obtained in this case are usually effective. In many cases, the load change in the future is difficult to determine. That is why the assumed conditions exist. Assumed conditions are based on some certain research and obtained through repeated analysis. Therefore, some prerequisite are given prior to our predicting outcomes.

3. **Temporality**

The short-term load forecasting is conducted by applying scientific prediction method during a certain period of time scale, such as minutes, hours and days. In this way, the temporality is one important feature of the short-term load forecasting.

4. Multi-scheme

In various environments, sometimes it is necessary to predict the future load trends according to the uncertainty and conditionality of the short-term load forecasting. Thus, a variety of short-term load forecasting method is developed. The load forecasting is conducted based on the real-time data. The short-term load forecast model might fail to fulfill its function while the load characteristics change over time. Therefore, to ensure an accurate prediction during the change in load characteristics, it is necessary to choose an appropriate short-term load forecasting methods and make corresponding adjustments based on the previous short-term load forecasting model.

3.2 Classification of Developed STLF Methods

In terms of lead time, load forecasting is divided into four categories:

- Long-term forecasting with the lead time of more than one year.
- Mid-term forecasting with the lead time of one week to one year
- Short-term load forecasting with the lead time of 24 to 168 hours
- Very short-term load forecasting with the lead time shorter than one day

Different categories of forecasting serve for different purposes. In this thesis short-term load forecasting which serves the next day unit commitment and reliability analysis is focused on. The research approaches of short-term load forecasting can be mainly divided into two categories: *statistical methods* and *artificial intelligence* methods. In statistical methods, equations can be obtained showing the relationship between load and its relative factors after training the historical data, while artificial intelligence methods try to imitate human beings way of thinking and reasoning to get knowledge from the past experience and forecast the future load. The statistical category includes multiple linear regression, stochastic time series, general exponential smoothing, state space, etc. Recently support vector regression (SVR), which is a very promising statistical learning method, has also been applied to short-term load forecasting and has shown good results. Usually statistical methods can predict the load curve of ordinary days very well, but they lack the ability to analyze the load property of holidays and other anomalous days, due to the inflexibility of their structure. Expert system, artificial neural network (ANN) and fuzzy inference belong to the artificial intelligence category. Expert systems try to get the knowledge of experienced operators and express it in an “if...then” rule, but the difficulty is sometimes the experts’ knowledge is intuitive and could not easily be expressed. Artificial neural network does not need the expression of the human experience and aims to establish a network between the input data set and the observed outputs. It is good at dealing with the nonlinear relationship between the load and its relative factors, but the shortcoming lies in overfitting and long training time. Fuzzy inference is an extension of expert systems. It constructs an optimal structure of the simplified fuzzy inference that minimizes model errors and the number of the membership functions to grasp nonlinear behaviour of short term loads, yet it still needs the experts’ experience to generate the fuzzy rules.

Generally artificial intelligence methods are flexible in finding the relationship between load and its relative factors, especially for the anomalous load forecasting. In this section, types of load forecasting are illustrated in the following table in details. The input load flow study or emergency analysis in the daily operation like energy transfer scheduling, and demand-side management is needed to consider for short term load forecasting. In order to correlate predicted advancement in demand, the medium and long-term forecasting are applied for enlargement of capacity of generation, transmission and distribution. It has big challenge with the upward trend in the electricity market prices since forecasting accuracy is the basic requirement in energy field. Moreover, forecasting performance is important thing to balance between energy supply and consumption. Therefore, electrical producers and authority expect to reduce the generating energy cost. As mentioned above, STLF has been emerged more and more important role in Energy Management System (EMS) with power markets development over the past decades.

Under normal circumstances, according to the forecasting period and purposes, load forecasting can be classified as **long-term load forecasting**, **medium-term load forecasting**, **short-term**

load forecasting, and **ultra short-term load forecasting**. Usually, the load and power consumption can be predicted by adopting these different time-scale forecasts. The **long-term load forecast** period is generally 10 to 15 years or even longer. The prediction target is usually the regional load capacity or the annual electricity consumption. The prediction purpose is to provide the base data for the power grid planning that help determine the grid operation mode and annual maintenance plans. The main factors affecting long-term load forecasting are national economic development, population, industrial restructuring and national tariff policy and so on. Currently, mature prediction methods comprise the trend extrapolation and various types of relevant prediction methods taking into account main factors, such as regression analysis, running average, exponential smoothing and grey prediction.

The **medium-term load forecast** period is generally 5 or 6 years. The prediction target usually is the load capacity of a region or the monthly electricity consumption. The forecast data generally indicates cyclical growth and each month of one year consists with the similar growth pattern. The prediction purpose is to arrange monthly maintenance plan, operation mode, reservoir operation plans and coal transportation plans. The main factors affecting medium-term load forecasting come from production planning from large users, weather conditions, industrial restructuring situations and national tariff policy and so on. Currently, the mature prediction methods are trend extrapolation based on the historical data of the same month and the time series prediction methods based on the yearly data.

The **short-term load forecast** period is generally 24 or 48 hours or even one week. The prediction target usually is the load capacity of a region or the daily and weekly electricity consumption data. The forecasting data generally indicates daily or monthly periodicity and the same date type of one year that follows the similar periodic pattern. The prediction purpose is to arrange day forecast for power generation projects as well as suspending or restarting power plans. The main factors affecting short-term load forecasting are week type, weather conditions and national tariff policy and so on. Currently, the mature prediction methods are the trend extrapolation based on the historical data of the same day and the artificial neural network based on the daily data .

The **ultra short-term load forecast** period is generally within 24 hours. The prediction target is usually the load capacity of several hours after the current moment. The forecast data generally has the similar change of instantaneous rate with the data a few days ago. The prediction purpose is mainly used for real-time security analysis, real-time economic dispatch, automatic generation control (AGC). There are few factors that can influence the ultra short-term load forecasting. However, only in summer the temperature can be regarded as a main impact factor contributing to the change on the predicted results. Currently, the mature prediction methods are trend extrapolation allowing for the instantaneous variation rate during the same time interval of previous several days, such as linear extrapolation and exponential smoothing methods.

3.3 Influencing factors of short-term load forecasting

Similarly, there are multiple influencing factors concerning the demand of load which can be analyzed as time, day, temperature, weather, random and economic factors. Generally, these factors can be mainly classified into the following categories:

- weather
- time
- economy
- random disturbance.

Among them, one of the most popular factors is temperature associated with meteorological situations for short term load forecasting. Selection of suitable variables for load forecasting is related to generalized mathematical models. The accuracy of such models depends on the quality of input information. Deterministic and stochastic are two different categories of variable. On the other hand, in general, the electric consumption load is based on human activities. Similarly, human activities are also depended on population and their economic status. Therefore, the variables are more or less interconnected to each other. For instance, as a result of changing the consumer's

comfort feeling for heaters (water, room) and air conditioner and so on, it is caused by the changes of weather conditions.

Furthermore, time is one of the most popular factors that can be separated into midnight, morning, evening, night, lunch time and so on when we consider one day. So, we can forecast next day electricity demand because we have different past data for one day. Similarly, we can consider seasonally, yearly, monthly, weekly and daily etc. As we note earlier, weather is also one important thing affecting load demand, for example, we can diverge temperature, cloud cover or sunshine, humidity and so on. Moreover, calendar can also be considered like seasonal variation, daily variation, weekly cyclic and holidays, hence we can get different data and predict different results. Likewise, the remaining things such as population, human facilities, economic for business and electricity prices are influenced on forecasting electricity load demand. Thus, it is sage to say that the system load is complicated and diverse since it is affected by different social and natural conditions. These social factors include energy utilization, agricultural structure, national policy, the rate of economic growth, population growth, social conditions, national holiday system, development level of science and technology. Natural factors consist of the complex and diverse changes in the weather, natural disasters, season changing and so on.

Since the short-term load forecast has a shorter time interval during the prediction, the main factors, including the level of social progress, national policies, the use of energy and the changing of the seasons and other natural factors, has less impact on the short-term load forecasting. Usually there are four main factors that can impact the shorter load forecast.

1. **Meteorological condition.** It includes many aspects like season, wind, pressure, temperature, humidity and sunshine and many other conditions. With the people's living standards improvement and the social economics development, an increasing number of household electrical appliances are put into daily use, such as refrigerators and air conditioners. All those electrical devices make residential load become an increasing component of the total power load. Since the impacts of changes from the season, temperature and other meteorological factors have been increasingly imposed on the load, it is necessary to consider their impact on human comfort and psychological indices due to meteorological factors within the acceptable levels. For the detailed short-term load forecasting, the associated meteorological factors are carefully selected in combination with the historical data of the actual power system load sequence so as to make more accurate prediction.
2. **Holidays.** Typically, holidays such as Christmas, Easter and Weekend usually poses a certain impact on the load changes of power system. This is mainly because during the holidays, the power consumption is considerably reduced by most enterprises and other high-power industrial load. On the contrary, the main component of the power system load includes the service industries, such as residential electricity consumption, commercial electricity, catering industry. So, the overall power consumption level is dramatically reduced.
3. **Emergencies.** Several urgent factors cause the interference of the power load, such as: unexpected incidents, unplanned overhaul, large electricity load fluctuations and limitation of electricity consumption.

Therefore, it is useful to conduct the corresponding processing about the historical load data. Bottom line is the short-term load forecasting accuracy level is the outcome of the combined effect of a variety of influencing factors. For short-term load forecasting, the right technology must be applied to address those associated influencing factors in order to achieve the precise and scientific short-term load forecasting. However, the impact that influencing factors have on the future load change are usually difficult to be defined by using a specific function expression. Meanwhile, the impact of short-term load forecasting factors may be correlated with each other. All of these conditions undoubtedly increase the difficulty of short-term load forecasting.

3.3.1 Load Forecasting Techniques

STLF has become one of the major areas of research in electrical power engineering because it is essential concerning the efficient operation of a power system. The electricity supply industry uses forecasting in a mode termed *off-line forecasting*, which refers to the scheduling of generating units based on expected demand. The accuracy of these forecasts is essential for the reliability of supply in large part due to the relatively large time constants of thermal generating units, which

account for a large fraction of the generating capacity are the most prolific in the power supply industry. The electric power industry uses a technique termed *online forecasting* to ensure that load dispatching is within the operational parameters of the system. Load dispatching is performed a few hours in advance, and is therefore dependent on the accuracy of the forecasts in order to find the optimal generating mix to minimize the cost function. Many techniques have been used in both online and offline forecasting, with most properly classified as statistical techniques, although the emergence of artificial intelligence technologies has expanded the options over the last decades.

The restructuring and modernization of the electric power industry has also influenced the importance of forecasting, putting more pressure on the financial risks associated with poor predictions and potentially changing the way electricity is consumed. The driving force to improve forecasting techniques, as well as the differences of economic and climatic variables between utilities triggered an active research area in STLF. The system load is influenced by a number of factors, including: economics, time, weather, and random effects. One of the challenges associated with STLF, is that these factors influence different areas in different ways and therefore require some specificity built into the prediction algorithms. The following is an abridged list of STLF techniques used by the power industry, not including the multitude of variations and approaches to each method:

- Multiple Linear Regression
- Stochastic Time Series
- General Exponential Smoothing
- State Space Methods
- Knowledge Based Approach
- Kalman Filter
- Neural Networks

In the following sections, we will look at how some of the other techniques in the previous list have been applied to LSTF. The variability of electric power service areas and generation portfolios motivates the varying forecasting techniques, of which a few are examined in this section.

TimeSeries Forecasting Methods

Time series forecasting methods are based on the assumption that the data have an internal structure, such as autocorrelation, trend or seasonal variation. The methods detect and explore such a structure. Time series have been used for decades in such fields as economics, digital signal processing, as well as electric load forecasting. In particular, ARMA (Autoregressive Moving Average), ARIMA (Autoregressive Integrated Moving Average) and ARIMAX (Autoregressive Integrated Moving Average with eXogenous variables) are the most often used classical time series methods. ARMA models are usually used for stationary processes while ARIMA is an extension of ARMA to nonstationary processes. ARMA and ARIMA use the time and load as the only input parameters. Since load generally depends on the weather and time of the day, ARIMAX is the most natural tool for load forecasting among the classical time series models. In our thesis and based on the seasonality that is present in our data, we will use another variation called SARIMAX (Seasonal Autoregressive Integrated Moving Average with eXogenous variables).

Neural Networks

The use of artificial neural networks (ANN or simply NN) has been a widely studied load forecasting technique since 1990. Neural networks are essentially non-linear circuits that have the demonstrated capability to do non-linear curve fitting. The outputs of an artificial neural network are some linear or non-linear mathematical function of its inputs. The inputs may be the outputs of other network elements as well as actual network inputs. In practice network elements are arranged in a relatively small number of connected layers of elements between network inputs and outputs. Feedback paths are sometimes used. In applying a neural network to load forecasting, one must select one of a number of architectures (e.g. Hopfield, Back Propagation, Boltzmann machine), the number and connectivity of layers and elements, use of bi-directional or uni-directional links and

the number format (e.g. binary or continuous) to be used by inputs and outputs. The most popular artificial neural network architecture for load forecasting is back propagation. This network uses continuously valued functions and supervised learning. That is, under supervised learning, the actual numerical weights assigned to element inputs are determined by matching historical data (such as time and weather) to desired outputs (such as historical loads) in a pre-operational “training session”. Artificial neural networks with unsupervised learning do not require pre-operational training.

Similar Day Approach

This approach is based on searching historical data for days within one, two or three years with similar characteristics to the forecast day. Similar characteristics include weather, day of the week and the date. The load of a similar day is considered as a forecast. Instead of a single similar day load, the forecast can be a linear combination or regression procedure that can include several similar days. The trend coefficients can be used for similar days in the previous years.

Integration of Different Algorithms

As there are many presented methods for STLF, it is natural to combine the results of several methods. One simple way is to get the average value of them, which can lower the risk of individual unsatisfactory prediction. A more complicated and reasonable way is to get the weight coefficient of every forecasting method by reviewing the historical prediction results. The comprehensive result is deduced by weighted average method.

3.3.2 Requirements of the STLF Process

In nearly all the energy management systems of the modern control centres, there is a short-term load forecasting module. A good STLF system should fulfill the requirement of accuracy, fast speed, automatic bad data detection, friendly interface, automatic data access and automatic forecasting result generation.

Accuracy

The most important requirement of STLF process is its prediction accuracy. As mentioned before, good accuracy is the basis of economic dispatch, system reliability and electricity markets. The main goal of most STLF literature and also of this thesis is to make the forecasting result as accurate as possible.

Fast Speed

Employment of the latest historical data and weather forecast data helps to increase the accuracy. When the deadline of the forecasted result is fixed, the longer the runtime of the STLF program is, the earlier historical data and weather forecast data can be employed by the program. Therefore the speed of the forecasting is a basic requirement of the forecasting program. Programs with too long training time should be abandoned and new techniques shortening the training time should be employed. Normally the basic requirement of 24 hour (96 points) forecasting should be less than 20 minutes.

Automatic Bad Data Detection

In the modern power systems, the measurement devices are located over the system and the measured data are transferred to the control centre by communication lines. Due to the sporadic failure of measurement or communication, sometimes the load data that arrive in the dispatch centre are wrong, but they are still recorded in the historical database. In the early days, the STLF systems relied on the power system operators to identify and get rid of the bad data. The new trend is to let the system itself do this instead of the operators, to decrease their work burden and to increase the detection rate.

Friendly Interface

The interface of the load forecasting should be easy, convenient and practical. The users can easily define what they want to forecast, whether through graphics or tables. The output should also be with the graphical and numerical format, in order that the users can access it easily.

Automatic Data Access

The historical load, weather and other load-relevant data are stored in the database. The STLF system should be able to access it automatically and get the needed data. It should also be able to get the forecasted weather automatically on line, through Internet or through specific communication lines. This helps to decrease the burden of the dispatchers

Automatic Forecasting Result Generation

To reduce the risk of individual inaccurate forecasting, several models are often included in one STLF system. In the past such a system always needs the operators' interference. In other words, the operators have to decide a weight for every model to get the combined outcome. To be more convenient, the system should generate the final forecasting result according to the forecasting behavior of the historical days.

Portability

Different power systems have different properties of load profiles. Therefore a normal STLF software application is only suitable for the area for which it has been developed. If a general STLF software application, which is portable from one grid to another, can be developed, the effort of developing different software for different areas can be greatly saved. This is a very high-level requirement for the load forecasting, which has not been well realized up until today.

3.3.3 Difficulties in the STLF

Precise Hypothesis of the Input-output Relationship

Most of the STLF methods hypothesize a regression function (or a network structure, e.g. in ANN) to represent the relationship between the input and output variables. The optimal way to define the regression form or the network structure is a major difficulty because it needs detailed a prior knowledge of the problem. If the regression form or the network structure were improperly selected, the prediction result would be unsatisfactory. For example, when a problem itself is a quadratic, the prediction result will be very poor if a linear input-output relationship is supposed. Another similar problem is parameter selection: not only the form of the model, but also the parameters of it should be well selected to get a good prediction. Moreover, it is always difficult to select the input variables. Too many or too few input variables would decrease the accuracy of prediction. It should be decided which variables are influential and which are trivial for a certain situation. Trivial ones that do not affect the load behavior should be abandoned.

Generalization of Experts' Experience

Many experienced working staff in power grids are good at manual load forecasting. They are even always better than the computer forecasting. So it is very natural to use expert systems and fuzzy inference for load forecasting. But transforming the experts' experience to a rule database is a difficult task, since the experts' forecasting is often intuitive.

The Forecasting of Anomalous Days

Electric loads of anomalous days are not easy to be predicted precisely, due to the dissimilar load behaviour compared with those of ordinary days during the year, as well as the lack of sufficient samples. These days include public holidays, consecutive holidays, days preceding and following the holidays, days with extreme weather or sudden weather change and special event days. Although the sample number can be greatly enhanced by including the days that are far away from the target day, e.g. the past 5 years historical data can be employed rather than only one or two years, the load growth through the years might lead to dissimilarity of two sample days. From the experimental

results it is found that days with sudden weather change are extremely hard to forecast. This sort of day has two kinds of properties: **the property of the previous neighbouring** and **the property of the previous similar days**. The combination of these two properties is a challenging task.

Less Generalization Ability Caused By Overfitting

Overfitting is a technical problem that needs to be solved during the load forecasting procedure. Historical training data are trained in the proposed model and a basic representation can be obtained and in turn used to predict the testing data. For the out-coming training module, if the training error for the training data is low but the error for the testing data is high, “overfitting” is said to have occurred. A significant disadvantage of neural networks is overfitting; it shows perfect performance for training data prediction but much poorer performance for the future data prediction. Since the goal of STFL is to predict the future unknown data, technical solutions should be applied to avoid overfitting.

The Destroy of Load Curve Nature By Compulsory Demand-side Management

Because of the economical development and relative lag in power investment, energy shortage has appeared in many countries. To avoid reliability problem and assure the power supply of very important users, compulsory demand-side management is often executed. This compulsory command destroys the natural property of load curve. When this kind of load curve is included in training, it serves as noise and deteriorates the final results.

Chapter 4

Time Series

4.1 Components of Time Series

Whether we wish to predict the trend in financial markets or electricity consumption, time is an important factor that must now be considered in our models. For example, it would be crucial to not only know when the electricity demand will move up, but also when it will move down. When forecasting Time Series data, the aim is to estimate how the sequence of observations will continue into the future. The simplest Time Series forecasting methods use only information on the variable to be forecast, and make no attempt to discover the factors that affect its behaviour. Therefore they will extrapolate trend and seasonal patterns, but they ignore all other information such as marketing initiatives, competitor activity, changes in economic conditions, and so on.

Time-Series is a sequential set of observations(data points), usually recorded in a proper chronological order and is mathematically formulated as a set of vectors

$$x(t), t = 0, 1, 2, \dots$$

where t denotes the elapsed time. A Time Series containing records of a single variable is termed as univariate, otherwise when records of more than one variable are considered, it is termed as multivariate. Furthermore, Time-Series can be continuous or discrete. In a continuous Time Series observations are measured at every instance of time, whereas a discrete Time Series contains observations measured at discrete points of time. For instance temperature, load demand etc. can be presented as continuous Time-Series, whereas exchange rates between two currencies is considered to be under the class of discrete Time Series. Usually in a discrete Time-Series the consecutive observations are recorded at equally spaced time intervals such as hourly, daily, weekly, monthly or yearly time separations. The variable under study in a discrete Time-Series is supposed to be measured as a continuous variable using the real number scale. Furthermore, concerning continuous Time-Series, they can be transformed to discrete by resampling (merging) observations over a specified time period.

A Time-Series is non-deterministic in nature, i.e. we cannot predict with certainty what will occur in future. Generally a Time Series $\mathbf{x}(t), t = 0, 1, 2, \dots$ is assumed to follow certain probability model which describes the joint distribution of the random variable \mathbf{x} . The formula that characterizes the probability structure of the Time-Series is termed as *a stochastic process* [23]. Thus the sequence of observations of the series is actually a sample realization of the stochastic process that produced it. A usual assumption is that the Time Series variables are independent and identically distributed e.g they follow the normal distribution. However, an interesting point is that Time Series are in fact following more or less some regular pattern in long term. For example if the temperature today of a particular city is extremely high, then it can be reasonably presumed that tomorrow's temperature will also likely to be high. This is the reason why Time Series forecasting using a proper technique, yields result close to the actual value.

In general, a Time Series is simply a series of data points ordered in time. In a Time Series, time is often the independent variable and the goal is usually to make a forecast for the future, however, there are other aspects that come into play when dealing with Time Series.

- **Is it stationary?** Stationarity is an important characteristic of Time Series. A Time Series is said to be stationary if its statistical properties do not change over time. In other words,

it has constant mean and variance, and covariance is independent of time. Ideally, we want to have a stationary Time Series for modelling.

- **Is there a seasonality pattern?** Seasonality refers to periodic fluctuations. For instance, electricity load demand peaks during the day and presents lower values during the night.
- **Is the target variable autocorrelated?**

There are two main goals of Time Series analysis: **(a) identifying the nature of the phenomenon represented by the sequence of observations**, and **(b) forecasting (predicting future values of the Time Series variable)**. Both of these goals require that the pattern of observed Time Series data is identified and more or less formally described. Once the pattern is established, we can interpret and integrate it with other data. Regardless of the depth of our understanding and the validity of our interpretation of the phenomenon, we can extrapolate the identified pattern to predict future events. Most Time Series patterns can be described in terms of two basic classes of components: *trend and seasonality*. The term trend represents a general systematic linear or (most often) nonlinear component that changes over time and does not repeat or at least does not repeat within the time range captured by our data. The general tendency of a Time Series to increase, decrease or stagnate over a long period of time is termed as Secular Trend or simply Trend. Thus, it can be said that trend is a long term movement in a Time Series.

The term seasonality may have a similar nature (e.g. a plateau followed by a period of exponential growth), however, it repeats itself in systematic intervals over time. More formally, it's defined as the correlational dependency of order k between the i_{th} series' observation and the $(i - k)_{th}$ observation, known as autocorrelation; k is usually called the lag. If the measurement error is not too large, seasonality can be visually identified in the series as a pattern that repeats every k elements. Those two general classes of Time Series components may coexist in real-life data. Seasonal patterns of Time-Series can be represented by correlograms. A correlogram visualizes and quantifies the autocorrelation function (ACF), or in other words the sequential correlation coefficients (and their standard errors) for consecutive lags in a predefined range of lags. To determine a proper model for a given time series data, it is necessary to carry out the ACF and PACF analysis. These statistical measures reflect how the observations in a time series are related to each other. For modeling and forecasting purpose it is often useful to plot the ACF and PACF against consecutive time lags. These plots help in determining the order of AR and MA terms. Below we give their mathematical definitions: For a time series $x(t), t = 0, 1, 2, \dots$ the Autocovariance at lag k is defined as:

$$\gamma_k = Cov(x_t, x_{t+k}) = E[(x_t - \mu)(x_{t+k} - \mu)]$$

The Autocorrelation Coefficient at lag k is defined as:

$$\rho_k = \frac{\gamma_k}{\gamma_0}$$

Here μ is the mean of the time series, i.e. $\mu = E[x_t]$. The Autocovariance at lag zero i.e. γ_0 is the variance of the time series. From the definition it is clear that the autocorrelation coefficient ρ_k is dimensionless and so is independent of the scale of measurement. It's obvious that $-1 \leq \rho_k \leq 1$. Box & Jenkins termed γ_k as the theoretical Autocovariance Function (ACVF) and ρ_k as the theoretical Autocorrelation Function (ACF). Furthermore, Partial Autocorrelation function (PACF) represents the correlation between an observation k in the past and the current observation, after checking for observations at intermediate lags (i.e. at lags $< k$). In general, the stochastic process of a Time-Series is not known and thus it's not possible to determine neither the actual nor the theoretical ACF and PACF values. The ACF and PACF values calculated from the training data are respectively termed as sample ACF and PACF.

4.2 Related Work and Models

Time-series forecasting is a complex research field therefore has attracted researcher's spotlight over the recent years. In general, the goal of Time-Series forecasting is the collection and studying of a variable's past values/observations, in order to build the optimal model that describe the data's internal structure. In other words, Time-Series modeling can be described as the prediction of the future based on the past observations. The constant need for Time-series modeling can be observed

in many real-world scenarios, thus reasonable case is necessary in order to fit the optimal model to describe the relevant time-series data. Over the past decades, researchers developed numerous efficient algorithms concerning Time-Series modeling and improvement of forecasting accuracy.

The most popular and common stochastic Time-Series model, is the Autoregressive Integrated Moving Average (ARIMA) model. This model supposes that the Time-Series under study, is linear and follows a known distribution, like the normal distribution. ARIMA model is a superset of other stochastic Time-Series models, like the Autoregressive (AR), Moving Average (MA) and Autoregressive Moving Average (ARMA). On the other hand, concerning seasonal Time-Series modeling, Box and Jenkins proposed a rather effective variation of ARIMA, which is called the Seasonal ARIMA (SARIMA). The main advantage of ARIMA model is its ability to describe multiple variations of Time-Series data in a flexible and simplistic way. On the other hand, the main drawback of these model is the assumption of the relevant Time-Series data linearity, which is not really the case for most of the real-world scenarios. In order to overcome the above mentioned limitation, numerous non-linear stochastic models have been proposed in literature, nonetheless they are not simplistic and easy-to-understand like the ARIMA model.

Lately, also Neural Networks have drawn significant attention in the research around Time-Series modeling. Originally ANNs were inspired from biology, although they have been efficiently applied for forecasting and classification tasks. Neural Networks are capable of modeling non-linear data and moreover they do not require the associated data to follow any statistical distribution, thus making them excellent candidates for Time-Series modeling. Neural Networks build the model in an adaptive way based on the given data, so they are concerned as data-driven and self-adaptive forecasting models. Over the past decades, numerous articles have been published in literature concerning the application of ANNs for Time-Series modeling and the research is still going on.

A big shift concerning Time-Series modeling, emerged with the introduction of Support Vector Machines (SVM). Although SVMs were originally used to deal with pattern classification tasks, eventually were applied in many other forecasting problems. SVM's extraordinary feature is that it's intended not only for accurate classification, but also for better generalization on the training data, and thus making SVM a well known technique for Time-Series forecasting problems. SVM's incorporate structural risk minimization (SRM), in order to establish a decision rule with good generalization capacity, based on a subset of the training data observations, the support vectors. Furthermore, in SVMs the training procedure is similar to solving a linearly constrained quadratic optimization problem, therefore provides the globally optimal value in contrary with ARIMA or Neural Networks. Normally in SVM the input data are mapped to a high dimensional feature space, by utilizing some special functions, known as support vector kernels.

4.3 Fundamental steps in forecasting procedure

A forecasting procedure typically involves five basic steps.

- **Step 1: Problem definition** The most crucial task in a forecasting problem is the precise problem definition. This task requires data collection, database maintenance and most of all data understanding. Moreover, the analyst providing the forecasts has to be aware of way that the forecasts are going to be used inside the organisation.
- **Step 2: Information gathering.** In general, there are at least two types of information that must be incorporated in the forecasting procedure:
 - statistical data
 - relevant domain expertise data

Proper model fitting usually requires loads of historical data, which is not always easy to acquire. Moreover, its ordinary that as we go further in the past, historical data will be less helpful because of structural changes in the system under study. So a common procedure is to incorporate only the most recent data in our model.

- **Step 3: Exploratory Data Analysis (EDA).** One vital step in every forecasting problem is the EDA part, in other words data visualizations. By plotting our data we are able to identify patterns, trends and seasonality effects in our data. Furthermore, with exploratory data analysis we can quantify existing correlations among our variables.
- **Step 4: Model fitting.** Generally, the optimal model is a function of the data used to train the model and more specifically of the existing correlations among the response variable and any other explanatory variables. Each candidate model is based on a set of assumptions (explicit and implicit) and involves numerous parameters which have to be evaluated based on the known historical data.
- **Step 5: Model Evaluation.** Only after selecting a model and evaluating it's parameters, then we use the model to produce the forecasts. The selected model's performance can be evaluated, by the time the appropriate data for the forecast period become available. In literature, numerous accuracy metrics have been developed in order to asses model's forecasting accuracy.

4.4 Time Series Modeling

Usually when plotting Time-Series, we may observe arbitrary irregularities- sudden ups and downs- in the data, which are result of unforeseeable extreme events such as strike, earthquake etc. Due to their randomness, the above mentioned fluctuations cannot be predicted by any statistical technique existing in literature. Considering the above mentioned limitations, two different types of models are generally used for Time-Series modelling and these are **Multiplicative and Additive models**.

$$\text{MultiplicativeModel} : Y(t) = T(t) \times S(t) \times C(t) \times I(t)$$

$$\text{AdditiveModel} : Y(t) = T(t) + S(t) + C(t) + I(t)$$

where $T(t)$, $S(t)$, $C(t)$ and $I(t)$ stand for trend, seasonality, cyclical and irregular variation respectively. The additive model presupposes that the four above mentioned components are independent, when on the other hand that's not the case for multiplicative model, where the four components may affect one another.

As we noted earlier, we usually find Time-Series data in numerous areas like economy, industry etc. [7, 8, 10] and based on the business need, different types of Time-Series data may exist. To illustrate the pattern of Time-Series data, we plot the recorded observations in relation to their time of occurrence, based on chronological order. In the following graphs, we provide such visualizations. The first Time-Series plot originates from the paper [8] and presents the weekly exchange rate between the US dollar (\$) and the GBP (£) for the specified time between 1980 and 1933. The second plot presents seasonal Time-Series data and show on monthly basis, the number of worldwide airline passengers between January 1949 and December 1960 as examined in paper [3, 6, 11]. **Time-Series Analysis** is the procedure of fitting an appropriate Time-Series model on known data and estimating the relevant model's parameters. In Time-Series analysis, we gather past observations in order to construct an appropriate mathematical model that captures in the best way data patterns and their nature. Afterwards, the constructed model is used to predict the Time-Series' future values, which is a crucial procedure because business decisions and precautionary measures are obtained based on these forecasts.

Last but not least, we should have in mind that When constructing a Time-Series model we should incorporate the principle of **parsimony**. Under this principle, the model with the smallest number of parameters among the other candidate models should be preferred, nevertheless providing an accurate description of the associated Time-Series inherent features. In simple words, when you are requested to choose among a range of adequate explanations (models) of the Time-Series data, choose the simplest one. Furthermore, the more complex the model, the larger the risk of overfitting. An over-fitted model performs very well on the training data, although it's not going to be accurate for forecasting. Having that in mind, parsimony principle is usually incorporated into the model, as a guide to prevent model overfitting. So to sum up, we should always choose the most parsimonious model between the available candidates when constructing forecasting models.



Figure 4.1: Weekly BP/USD exchange rate series (1980-1993)

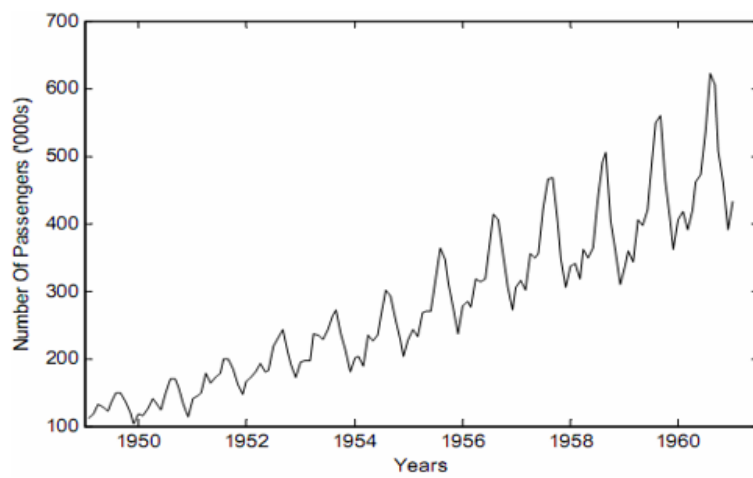


Figure 4.2: Monthly international airline passenger series (Jan 1949-Dec 1960)

Chapter 5

Time Series Forecasting using ARIMA & SARIMAX models

Selecting the appropriate model for fitting the data is a procedure of high importance, because it mirrors the underlying structure of the time-series and the produces the future forecasts. A time-series model may be labeled as linear or non-linear depending on whether each value of the time-series is a linear or non-linear function of the previous observations. Generally, time-series models may have numerous forms and illustrate different stochastic processes. Autoregressive moving average models fall into the category of stochastic models and their main objective is to describe the evolution of a phenomenon based on the time. Since it is not possible to record all the factors that affect load evolution over time, it is very difficult to be described by a deterministic model. Yet the choice of a stochastic model, such as ARIMA, is satisfactory as load demand depends on non-deterministic factors, such as weather or any random events, and a description of load demand over time can be succeeded with this stochastic model. These stochastic models contain randomness (random error or prediction error), the values of the metric under study over time and in some cases other stochastic factors, thus the resulting model is a linear combination of the above mentioned terms.

The most common linear time-series models are the AutoRegressive (AR) and Moving Average (MA) models. Moreover in literature, combinations of these two models the AutoRegressive Moving Average (ARMA) and the AutoRegressive Integrated Moving Average (ARIMA) models have been proposed. Due to their simplicity, intuitiveness and implementation, linear models have attracted a lot of attention nowadays. Nonetheless most of the time-series in real-life scenarios present non-linear patterns, such as stock price forecasting where non-linear models are considered to be more accurate for predicting volatility changes in economic time-series. So numerous non-linear time-series models have been proposed in literature and in this chapter we will introduce remarkable linear and non-linear stochastic time-series models and their salient features.

5.1 ARMA & ARIMA Models

An ARMA model (p, q) is a combination of AR (p) and MA (q) models and is suitable for time-series modeling. An AR model (p) calculates the future value of the response variable as a linear combination of previous observations for that value and a random error factor together with a fixed term. Mathematically the model AR (p) can be formulated as [12, 23]:

$$y_t = c + \sum \phi_i y_{t-i} + \epsilon_t$$

where y_t and t are respectively the actual value and random error in time period t , with $i = 1, 2, \dots$ while p is a model parameter and c is a constant. Integer constant p is referred to as the model's order. Furthermore, for the shake of simplicity the fixed term is omitted. In order to assess the parameters of an AR procedure based on the time-series data, we use the Yule-Walker equations [23].

On the other hand, just like an AR model (p) relates previous values in the series, an MA model (q) uses previous errors as variables. The model MA (q) is given by [12, 21, 23] and can be

expressed using the following equation.

$$y_t = \mu + \sum \theta_j \epsilon_{t-j} + \epsilon_t = \mu + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} \dots + \theta_q \epsilon_{t-q} + \epsilon_t$$

By default, ARMA models should and can be used only for stationary time-series. Nevertheless, in real life scenarios most of the time-series data, such as those related to economics and operations present non-static behavior. Time-series data that contain trend and seasonal patterns, are also considered to be non-stationary [3, 11]. Thus, the ARMA models are not capable to properly describe the non-stationary time-series data, which are frequently found in real life case studies. In order to bridge that gap, the ARIMA model is therefore proposed [6, 23, 27], which is a generalization of an ARMA model aiming to overcome the phenomenon of non-stationarity. More specifically, in ARIMA models, a non-stationary time-series is converted to stationary by applying finite differentiation on the data. The ARIMA model is the most common forecasting technique for one-dimensional time-series data. Although the model can handle sufficiently non-stationary time-series data, it isn't capable of supporting seasonal time-series data. ARIMA works only with data that are either non-seasonal or seasonal data where the seasonality has been removed, thus turning the series into stationary. An extension to ARIMA that supports the modeling of the seasonal element of time-series is called SARIMA.

5.2 SARIMA model

The ARIMA models that we discussed above are intended only for non-seasonal data modeling. On the other hand, a seasonal ARIMA model is an enhancement of ARIMA models, created by incorporating additional seasonal terms to the widely spread non-seasonal ARIMA models. The previously mentioned ARIMA variation is referred to as SARIMA (p, d, q) (P, D, Q, m), where (p, d, q) is the non-seasonal part of the model, (P, D, Q, m) is the seasonal part of the model and finally m is the number of time steps for a single seasonal period. In other words, SARIMA introduces three new parametric elements in order to define AutoRegression (AR), Integration (I) and Moving Average (MA) concerning the seasonal nature of time-series, along with an additional parameter for the seasonality period. In the following list, we present the SARIMA model parameters that require tuning:

- p: Trend AutoRegression order.
- d: Trend difference order.
- q: Trend Moving Average order.
- P: Seasonal AutoRegression order.
- D: Seasonal difference order.
- Q: Seasonal Moving Average order.
- m: The number of time periods in each seasonal period.

5.3 SARIMAX Model

Last but not least, when including in our model external input variables, like temperature, atmospheric pressure, for example when we are dealing with time-series data concerning rain frequency, then we will use another variation of the ARIMA model that is called ARIMAX model and generally can be expressed with the following formula:

$$\phi(B)\nabla^d z(t) = w(B)x(t-b) + \theta(B)a(t)$$

where $x(t)$ is the external variable in time t and When the ARIMA model contains seasonal

$$w(B) = w_0 + w_1B + w_2B^2 + w_3B^3 + \dots$$

variations and we have to include external variables in our model, then this variation is called SARIMAX. SARIMAX is essentially used in case the Time Series under study, appears also seasonality except the trend. So in that case, the seasonal parameters are also included in the model and thus defining the Seasonal AutoRegressive Integrated Moving Average with eXogenous factor (SARIMAX (p, d, q)(P, D, Q, m)).

Stochastic time-series modeling present numerous attractive features. The theory behind these models is well known, thus it's relatively easy to understand the forecasting procedure and its implementation. The model's parameters are relatively easy to tuned and moreover, the estimation of the white noise difference, creates a sense of confidence in the model's predictions. Furthermore, it is possible to establish diagnostic methods and the assessment of the model's parameters is clear. Forecasting time-series models containing exogenous variables, have been used for prediction in a vast variety of applications. Exogenous variables are also referred as covariates, which are considered as parallel input sequences referring to observations which follow the same time sequence as the actual time-series data. The actual time-series data may be referred to as endogenous sequence in contrary to the exogenous data sequence. Exogenous variables observations are incorporated directly in the model at each time step, although there are not configured in the same way as the primary endogenous sequence (e.g. as model AR, MA, etc.).

Last but not least, the application of ARIMA models presupposes the forecasting of short-term intervals. As mentioned earlier, these models are a linear combination of the historical time-series data. This means that if we want to forecast the value Y on time t , then we have to know the values Y_{t-1} , Y_{t-2} . ARIMA models can be an excellent tool for understanding the evolution of values over time by analyzing and extrapolating them into the near future. ARIMA models deal with Time Series from a stochastic point of view and are based exclusively on the distribution of their past values and, above all, on the most recent ones. Our approach is based on a grid search approach, which tests various model order configurations. The final model order is selected by minimizing the Bayesian information criterion (BIC)

$$BIC = AIC + (\log(T) - 2)(p + q + k + 1)$$

Where AIC is the Akaike information criterion, which is defined as follows:

$$-2\log(L) + 2(p + q + k + 1)$$

where L is the value of maximum likelihood function of the model, k is the number of model's parameters, p is the autoregressive order, q is the model's moving average order and T is the number of observed samples.

Chapter 6

Time Series Forecasting using the PROPHET algorithm

Concerning the information used to provide the forecasts, there are two different approaches on time-series analysis. On the one hand, the standard approach includes only endogenous variables, while on the other hand the second approach incorporates exogenous variables also. The number of primary forecasting models is limited, whereas recent surveys have shown that the quality of forecasting could barely improved in the long term by switching between forecasting algorithms. In contrary, Hong and Fan (2016) underline that including appropriate exogenous forecasting variables could significantly improve the model's accuracy. The exogenous predictive variables that will be used to solve a particular forecasting problem could be derived from large-scale comparisons. Such comparisons facilitate benchmarking and model evaluation, i.e. procedures requiring large amounts of data.

The Prophet algorithm -developed by Facebook- is a process for forecasting time-series data based on an additive model where non-linear trends match annual, weekly or daily seasonality and holiday too. Prophet tends to work best with time-series data that present strong seasonal effects and considerable amount of historical data are available. The algorithm is thought to be very robust concerning lack of data, shifts the trend and usually it produces good results. According to the article on Facebook Research, Prophet was initially developed aiming to produce highly accurate forecasts to several business needs. This library tries to address the following difficulties common to many business Time Series:

- Seasonal effects caused by human behavior: weekly, monthly and yearly cycles, dips and peaks on public holidays
- Changes in trend due to new products and market events and
- Outliers

Furthermore, the authors claim that, even with the default settings, in many cases, their library produces forecasts as accurate as those delivered by experienced analysts. Moreover, Prophet has a number of intuitive and easily interpretable customizations that allow gradually improving the quality of the forecasting model. What is especially important, these parameters are quite comprehensible even for non-experts in Time Series analysis, which is a field of data science requiring certain skill and experience. By the way, the original article is called "Forecasting at Scale", but it is not about the scale in the "usual" sense, that is addressing computational and infrastructure problems of a large number of working programs. According to the authors, Prophet should scale well in the following 3 areas:

- Accessibility to a wide audience of analysts, possibly without profound expertise in Time Series.
- Applicability to a wide range of distinct forecasting problems.
- Automated performance estimation of a large number of forecasts including flagging of potential problems for their subsequent inspection by the analyst.

It has been observed that Prophet algorithm with its default settings, tend to generate forecasts quite accurate with considerably less effort in contrary with more advanced forecasting algorithms. In case the Prophet algorithm provides bad forecasts you don't have to do with a black box procedure, because with even no training in time-series analysis we can improve the model's forecasting ability just by tuning its parameters. We observed that a wide variety of business scenarios can be addressed by combining automated forecasting processes in conjunction with an analyst-in-the-loop for extremely special forecasting tasks. The following diagram presents the forecasting method observed to operate efficiently at scale: Prophet has been oftenly used as the

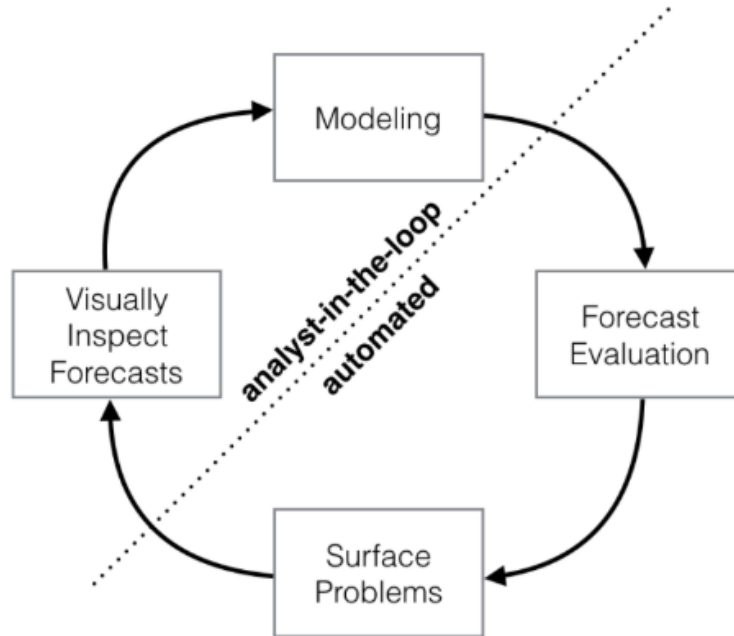


Figure 6.1: Automated forecasting process with analyst-in-the-loop

main forecasting algorithm in numerous case studies based on the following two benefits:

- It's much more genuine to generate reasonable and accurate forecasts with Prophet
- Prophet incorporates numerous different forecasting algorithms, such as ARIMA, exponential smoothing, etc, and also provides the ability for parameter tuning.

In general, selecting the wrong model or not the optimal parameter values, could sometimes generate poor forecasts, and even experienced data analysts may not be able to select the optimal model and its parameters, having in mind the variety of choices. As mentioned earlier, Prophet models may be easily customized by non-experts also, as smoothing parameters for seasonality and trend are provided, thus allowing the user to adjust the degree of how abruptly the algorithm follows trend changes and historical cycles. The principal concept in Prophet, is that by flexibly fitting the trend component and accurately modeling seasonality, we tend to generate more accurate forecasts. For tasks such as curve-fitting, we prefer to use flexible regression models instead of a traditional time-series models because it provides more modeling flexibility, thus making it straightforward to fit the model, handle missing data and outliers.

Accordingly, Prophet generates uncertainty intervals for the trend component by simulating future trend changes to the time-series data. In case of uncertainty modeling concerning future seasonality or holiday impacts, the user can run numerous Hamiltonian Monte Carlo (HMC) iterations and the results will incorporate seasonal uncertainty estimations. When we try to forecast a measures growth, such as total population, a maximum achievable point exists which is referred to as carrying capacity. Prophet is capable of generating forecasts by utilizing a logistic growth trend model bounded by a specified carrying capacity. The Prophet forecasting algorithm is provided in Python and R and predicts future timeline values by utilizing machine learning algorithms. The

Prophet algorithm is considered as an additive regression model with the following four principal components:

1. A piecewise linear or logistic growth curve trend (Prophet identifies trend changes by specifying data change points)
2. an annual seasonal term modeled using Fourier series
3. a weekly seasonal term using dummy variables
4. a user-provided array containing holidays

The parameters used in the algorithm, are customized such as to match every possible time-series provided as input. It includes several parameters that can either be given by the user as fixed or can be given with specific limits that give the algorithm full freedom to select the parameter's optimal values. The mathematical relationship that describes the algorithm output is shown below:

$$y(t) = g(t) + s(t) + h(t) + \epsilon(t)$$

where the above functions denote:

- $g(t)$: Models trend component in time-series, by including two different models. A piecewise linear model and a saturating growth model, based on the forecasting scenario under study.
- $s(t)$: Models seasonality by utilizing Fourier series, which describes how time-series is influenced by seasonal factors.
- $h(t)$: Models the holiday effect and other extreme events that impact time-series data.
- $\epsilon(t)$: The error term.

Each of these four terms is responsible for modeling and simulating a different parameter that contributes to our result and predicting the next day value of the Time Series. For periodic changes $s(t)$, they use Fourier transformation and the search of the optimal coefficients is being left to the algorithm. Moreover, functions like $g(t)$ are being simulated with logistic regression models with small changes and also with the introduction of additional parameters that are either fixed or time-related functions. Concerning $h(t)$ function, it is responsible for being able to identify which periods are holiday and to influence the equation accordingly. For the more correct use of this function, the algorithm -before running- gives the user the choice to enter his country's holiday. Concerning the term $e(t)$, it is a function which affects our result according to the data under examination. In particular, it tries to perceive patterns or anything else that affects our outcome and cannot affect any of the previous three functions. These functions are characterized by a large number of parameters which can either be introduced by use or give the algorithm the freedom to calculate them to minimize the error of forecasts. The error term $\epsilon(t)$ represents information that couldn't be explained by the model and thus it is included in the model as noise.

Time-series data in the real world, often present abrupt changes on their trajectories, then Prophet will automatically identify these changepoints allowing the trend component to adapt appropriately. Prophet identifies these changepoints by first specifying numerous potential changepoints, where rate change is allowed. Following that, applies a sparse prior on the magnitudes of the rate changes (similar to L1 regularization). In other words, Prophet detects numerous possible data points where the rate could change, but will use just a subset of them as potential changepoints. Regarding seasonality and special days, Prophet detects seasonality based on partial Fourier sum, where the number of included terms is specified by a parameter that determines defines how rapidly the seasonality can change. In contrary, when holidays should be modeled, its compulsory to define a new dataframe, which must include holiday occurrences, both in the past and in the future, as the forecasting horizon dictates. Last but not least, it is worth to mention that Prophet algorithm was designed by Facebook company and the algorithms' parameters have been tuned based on the distributions that company's data follow, which may not be the case in every Time Series that we try to apply to this algorithm.

Chapter 7

Time Series Forecasting using Neural Networks

7.1 Basic Concepts of Neural Networks

By default, the main characteristic of neural networks is their ability to learn. In layman's terms, when a neural network is learning it improves its ability to solve a problem through a repetitive process of gradually adjusting the neuron weights to the appropriate values that achieve the solution to the problem. Consecutively, by the end of the training step these parameters will get a fixed value. The challenging part is to achieve generalization. Generalization is the phenomenon where the neural network is general enough to be able to give appropriate outputs for inputs other than those it was trained for. Furthermore, neural networks provide another way of finding non-linear hypotheses since they decide to include ordered terms that best fit the data. Neural networks emulate the way the brain works and its learning procedure. In correspondence to a human's brain, the neuron is the basic building block and it consists of the axon, the dendrites and the synapses. The axon is the neuron exit gate and sends signals to other neurons, the dendrites are the neuron entry gates, and the synapses are the points of union between the neuron and the other neurons' dendrites. Neuron studying and the mathematical modeling of their structure led to artificial Neural Networks. The McCulloch-Pitts model as seen in the following figure, is a simple model for describing the neuron.

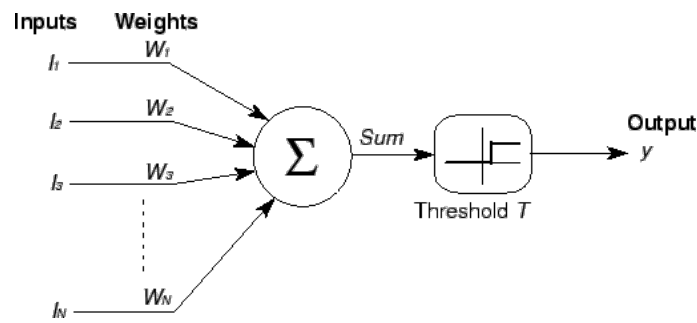


Figure 7.1: The McCulloch-Pitts Neuron model

As seen above, $I_1, I_2, I_3 \dots I_N$ are the neuron inputs and $w_1, w_2, w_3 \dots w_N$ are called synaptic weights. If the sum of the products $I_i * W_i$ is greater than the T threshold, then the neuron passes the sum into the output y , otherwise it remains inactive. In an effort to accurately simulate the brain's functionality, "Artificial Neurons" are a simplified version of the biological ones. The inputs are represented by x_i . Each input is multiplied by some weight w_i , and all $w_i * x_i$ products are added. Following this step, the processed input is fed into an "activation function" which creates the output value, called "activation" value. Finally, there is an output path. There is a number of Neural Networks architectures that are characterized by the differences in their activation functions. The most common one is the Sigmoid activation function $f(u) = 1/(1 + e^{-u})$. More specifically, the activation function can be step transfer function, a linear transfer function, a non-linear transfer

function or a stochastic transfer function. These artificial neurons can be connected in many ways to give “artificial neural networks”. The way that the network is structured is referred to as the network’s architecture. A network can be divided into different layers where the first and last one of them are called input and output layers, respectively. The layers in between are referred to as hidden layers and their number can arbitrarily large or small. It is important to point out though that layers only pass values forward and not backwards. Adding layers to the network increases its depth, as well as its accuracy. In such a network, there is one input layer, one or more hidden layers, and one output layer, as shown in the below figure. The layers in between are called hidden layers because they are hidden from the user.

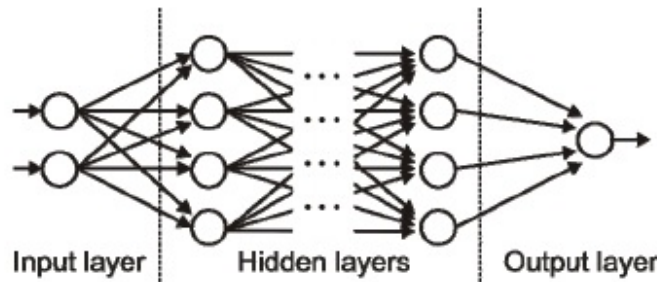


Figure 7.2: Artificial Neural Network’s Architecture

Usually in a neural network, each node in the previous layer is connected to every node in the following layer. Those created paths carry weights that the signal must be multiplied by when moving from one node to the other. This product is what reaches the end node. Each node receives several of these products from the nodes in the previous layer that connect to it. It sums them all up, applies the logistic function and passes the output forward to the next layer where the process repeats. Neural networks learning ability results from the tuning of the weights for each path so that the output layer produces a hypothesis that can model your data. In this way, complex non-linear hypotheses can be formed and the more hidden layers you have the potentially more complex models can be generated.

Neural networks can (accurately) predict an output upon receiving some input. Broadly, a neural network consists of four components:

- Artificial Neurons.
- Topology-how the neurons are connected
- Weights.
- Learning algorithm.

In the same way as every machine or deep learning algorithm, neural networks have to be trained on a dataset before they can be used to predict an output. Usually, the dataset consists of pairs of input-output. As a first step during training, all weights are randomly initialized and then the inputs are forward fed through the network to finally produce the output (“forward pass”). Of course, the output of every node in each hidden layer takes into consideration the activation function for the calculations. The arbitrary weight initialization is bound to result in inaccurate outputs and thus the goal is to calculate the weight values that can successfully predict them with reasonable accuracy. The algorithm used to do this is known as “Backpropagation”. After the forward pass, we have some activations at the output layer. An ideal neural network will result in outputs that are equal to the intended targets (known outputs). In practice this is not always the case and especially when the weights are randomly initialized. To account for that fact, the error is calculated using the “cost function” and then it is “backpropagated” through the network. As mentioned earlier, the goal is to adapt the weights so that the solution to the optimization problem of minimizing the cost function through “gradient descent” is acceptable. The plot of cost function vs weight is more or less convex and looks something like the following figure. Gradient

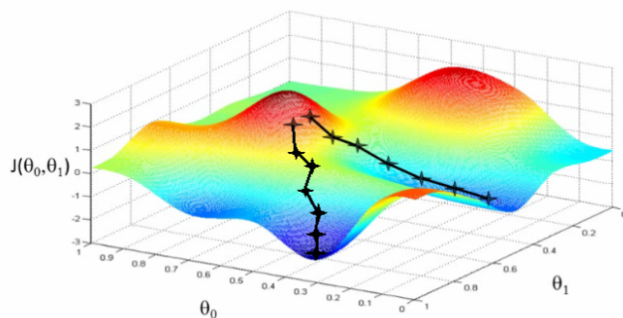


Figure 7.3: Cost function and Gradient Descent

descent operates under the assumption that we make small steps towards the global minimum of the gradient when the plot is convex. In case the plot is not completely convex, we consider that the local minima are a good approximation of the global one. So we make small updates on the weights by multiplying them with the “learning rate” parameter, each time moving them in the direction of the gradient. Neural networks are highly sophisticated non-linear modeling techniques, capable of modeling extremely complex functions. Keeping that in mind, they still fail to model complex phenomena where the popular linear models and their well-known optimization strategies are also straggling to show valid results. Of course, the problem on these types of networks is located on the dimensionality of the inputs and the outputs which could be an issue for large number of variables.

As a matter of fact, the manner in which the neurons of a neural network are structured is intimately linked with the learning algorithm used to train the network. For instance, in **Feed-forward Neural Networks** the information flow is strictly moving from the input layer to the output. In addition, in the **Single Layer Feedforward Networks** example the term “single” refers to the output nodes where the inputs are fed directly via a number of weights. More specifically, the feedforward network architecture utilizes a supervised learning approach to solve the issue of teaching it to perform well for a problem. Therefore, the inputs are fed into the network to produce the outputs through a pattern recognition process that indicates an input image of a “cat” or “elephant”, for example. As mentioned earlier, the goal is to train the network until it has the minimum category guessing error. With the trained set of parameters (or weights, collectively known as a model), the network sallies forth to categorize data it has never seen. An interesting advantage of this network architecture is the lack of time order perception. To be more precise, a trained feedforward network evaluates only the current input example and it will not assume that the next output is somehow dependent on the one under evaluation. A simple example is that seeing a photograph of a cat will not lead the network to perceive an elephant next.

A **Multilayer Feedforward Neural Network** is architected with a number of hidden units-neurons that formulate its hidden layers. The term “hidden” is used just because they do not have an immediate connection with the input or output layers and their main functionality is to help transform the input into the output in a more elaborate-meaningful kind of way. By adding one or more hidden layers, the network is enabled to extract higher-order statistics from its input. Finally, another special Neural Network architecture is the **Recurrent Neural Networks (RNN’s)**. A recurrent neural network differs by having at least one feedback. Feedforward networks are amnesiacs regarding their recent past. There are no self-feedback loops in the network. The feedback loops involve the use of particular branches composed of unit-time delay elements, which result in a nonlinear dynamic behavior.

7.2 Long Short-Term Memory Units (LSTMs)

Recurrent networks, in a sense, use a combination of inputs since they not only take into account the current ones but also the ones fed immediately before into the network. As a matter of fact, the decision a recurrent neural network reached at time step $t-1$ affects the decision it will reach one moment later at time step t . This is a distinguishable characteristic that separates recurrent

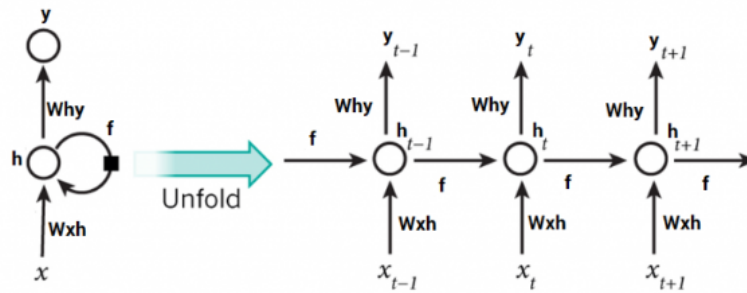
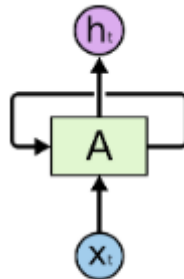


Figure 7.4: Unfolded basic Recurrent Neural Network

from feedforward networks, a perception of “memory”. **Adding memory to neural networks has a purpose: There is information in the sequence itself, and recurrent networks use it to perform tasks that feedforward networks can’t.** That sequential information is



Recurrent Neural Networks have loops.

Figure 7.5: Recurrent Neural Network loop

preserved in the recurrent network’s hidden state, which manages to span many time steps as it cascades forward to affect the processing of each new example. It recognizes correlations between events separated by many moments, and these correlations are called “long-term dependencies”, because an event downstream in time depends upon, and is a function of, one or more events that came before.

RNNs can be considered as a mechanism to share weights over time. The information travel in the hidden states of the recurrent networks and the phenomenon can be described mathematically as follows. The hidden state at time step t is h_t . It is a function of the input at the same time step x_t , modified by a weight matrix W (like the one we used for feedforward neural networks) added to the hidden state of the previous time step h_{t-1} multiplied by its own hidden-state-to-hidden-state matrix U , otherwise known as a transition matrix and similar to a Markov chain. The weight matrices are filters that determine how much importance to accord to both the present input and the past hidden state. The error they generate will return via backpropagation and be used to adjust their weights until error can’t go any lower. The sum of the weight input and hidden state is squashed by the function ϕ – either a logistic sigmoid function or tanh, depending – which is a standard tool for condensing very large or very small values into a logistic space, as well as making gradients workable for backpropagation. Because this feedback loop occurs at every time step in the series, each hidden state contains traces not only of the previous hidden state, but also of all those that preceded h_{t-1} for as long as memory can persist. Given a series of letters, a recurrent network will use the first character to help determine its perception of the second character, such that an initial q might lead it to infer that the next letter will be u , while an initial t might lead it to infer that the next letter will be h .

The main perk of RNNs is the fact that they have the ability to combine previous information with a task at hand. This is a very useful characteristic that needs to be used carefully since not

always do we have a problem that requires this combination of past and present knowledge. For example, consider a language model trying to predict the next word based on the previous ones. If we are trying to predict the last word in “the clouds are in the sky,” we don’t need any further context – it’s pretty obvious the next word is going to be sky. As a result, RNNs are able to learn and use the previous information needed to find the word. Their ability though is not limitless since as the learning time window increases RNNs are rendered unable to make connections with past information. Consider trying to predict the last word in the text “I grew up in France... I speak fluent French.” Recent information suggests that the next word is probably the name of a language, but if we want to narrow down which language, we need the context of France, from further back.

The aforementioned phenomenon of short memory for RNNs is the result of the problem of Vanishing Gradient that also affects feedforward neural networks. This term is used to describe the fact that the gradient almost vanishes due to the small values of the activation functions derivatives that get multiplied multiple times as we move towards the starting layers. RNN remembers things for just small durations of time, i.e. if we need the information after a small time it may be reproducible, but once a lot of words are fed in, this information gets lost somewhere. In theory, RNNs are absolutely capable of handling such “long-term dependencies.” In practice, on the other hand, extensive research and literature suggests that this is not the case.

To solve this problem, a special family of RNNs was created and is called Long Short Term Memory networks or LSTMs. LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn. RNNs network architecture resembles a chain of repeating modules that are comprised of a simple structure. LSTMs build on that simple structure and instead of having just one neural network layer, they have four mingling in a particular way. The key to LSTMs is the cell state, the horizontal line running through the top of the diagram. The cell state is kind of like a conveyor belt. LSTMs use this mechanism to move information around. We may have some addition, modification or removal of information as it flows through the different layers. It runs straight down the entire chain, with only some minor linear interactions. LSTMs

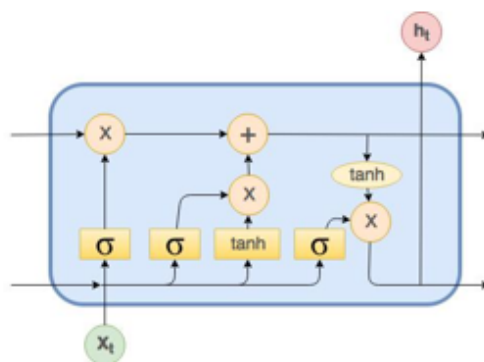


Figure 7.6: Inner structure of an LSTM unit

use gate structures to manipulate the information on the cell state. Even though information can run through unchanged, it can also be enriched or simplified this way. In general, gates are a way to optionally let information through. They are composed out of a sigmoid neural net layer and a pointwise multiplication operation. The sigmoid layer outputs numbers between zero and one, describing how much of each component should be let through. This way, LSTMs can selectively remember or forget things. The information at a particular cell state has three different dependencies. These dependencies can be generalized to any problem as:

- The previous cell state (i.e. the information that was present in the memory after the previous time step)
- The previous hidden state (i.e. this is the same as the output of the previous cell)

- The input at the current time step (i.e. the new information that is being fed in at that moment)

Now let's get into the details of the architecture of LSTM network: A typical LSTM network is

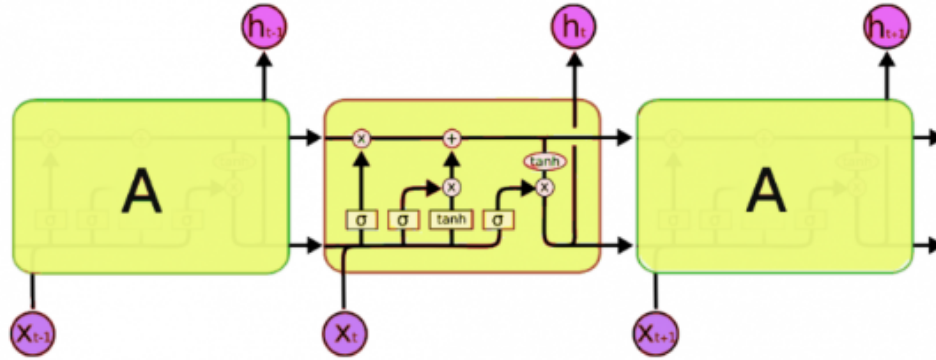


Figure 7.7: LSTM Network Architecture

comprised of different memory blocks called cells (the rectangles that we see in the image). There are two states that are being transferred to the next cell; the cell state and the hidden state. The memory blocks are responsible for remembering things and manipulations to this memory is done through three major mechanisms, called gates.

Forget gate As the name indicates, a forget gate is used to remove information from the cell state. This is a much-needed optimization technique for LSTMs since information that is no longer needed or of less importance is filtered out. This gate takes in two inputs; h_{t-1} and x_t . h_{t-1} is the hidden state from the previous cell or the output of the previous cell and x_t is the input at that particular time step. The given inputs are multiplied by the weight matrices and a bias is added. Following this, the sigmoid function is applied to this value. The sigmoid function outputs a vector, with values ranging from 0 to 1, corresponding to each number in the cell state. Basically, the sigmoid function is responsible for deciding which values to keep and which to discard. This vector output from the sigmoid function is multiplied to the cell state.

Input Gate The input gate is responsible for the addition of information to the cell state. This addition of information is basically three-step process as seen from the diagram above.

1. Regulating what values need to be added to the cell state by involving a sigmoid function. This is basically very similar to the forget gate and acts as a filter for all the information from h_{t-1} and x_t .
2. Creating a vector containing all possible values that can be added (as perceived from h_{t-1} and x_t) to the cell state. This is done using the tanh function, which outputs values from -1 to +1.
3. Multiplying the value of the regulatory filter (the sigmoid gate) to the created vector (the tanh function) and then adding this useful information to the cell state via addition operation.

Once this three-step process is done with, we ensure that only that information is added to the cell state that is important and is not redundant.

Output Gate The job of selecting useful information from the current cell state and showing it out as an output is done via the output gate. The functioning of an output gate can again be broken down to three steps:

1. Creating a vector after applying tanh function to the cell state, thereby scaling the values to the range -1 to +1.

2. Making a filter using the values of h_{t-1} and x_t , such that it can regulate the values that need to be output from the vector created above. This filter again employs a sigmoid function.
3. Multiplying the value of this regulatory filter to the vector created in step 1, and sending it out as a output and also to the hidden state of the next cell.

LSTMs contain information outside the normal flow of the recurrent network in a gated cell. Information can be stored in, written to, or read from a cell, much like data in a computer's memory. The cell makes decisions about what to store, and when to allow reads, writes and erasures, via gates that open and close. Unlike the digital storage on computers, however, these gates are analog, implemented with element-wise multiplication by sigmoids, which are all in the range of 0-1. Analog has the advantage over digital of being differentiable, and therefore suitable for backpropagation. Those gates act on the signals they receive, and similar to the neural network's nodes, they block or pass on information based on its strength and import, which they filter with their own sets of weights. Those weights, like the weights that modulate input and hidden states, are adjusted via the recurrent networks learning process. That is, the cells learn when to allow data to enter, leave or be deleted through the iterative process of making guesses, backpropagating error, and adjusting weights via gradient descent. The diagram below illustrates how data flows through a memory cell and is controlled by its gates.

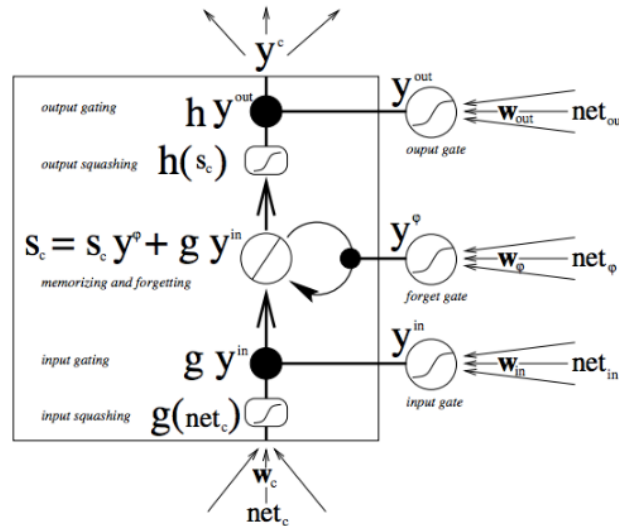


Figure 7.8: Data flow inside LSTM memory cell

Starting from the bottom, the triple arrows show where information flows into the cell at multiple points. That combination of present input and past cell state is fed not only to the cell itself, but also to each of its three gates, which will decide how the input will be handled. The black dots are the gates themselves, which determine respectively whether to let new input in, erase the present cell state, and/or let that state impact the network's output at the present time step. S_c is the current state of the memory cell, and gy^{in} is the current input to it. Remember that each gate can be open or shut, and they will recombine their open and shut states at each step. The cell can forget its state, or not; be written to, or not; and be read from, or not, at each time step, and those flows are represented here. The large bold letters give us the result of each operation. It's important to note that LSTMs' memory cells give different roles to addition and multiplication in the transformation of input. The central plus sign in both diagrams is essentially the secret of LSTMs. Simple as it may seem, this basic change helps them preserve a constant error when it must be backpropagated at depth. Instead of determining the subsequent cell state by multiplying its current state with new input, they add the two, and that quite literally makes the difference. (The forget gate still relies on multiplication, of course.) Different sets of weights filter the input for input, output and forgetting. The forget gate is represented as a linear identity function, because if the gate is open, the current state of the memory cell is simply multiplied by one, to propagate forward one more time step.

Chapter 8

Time Series Forecasting using Random Forest

8.1 Basic Components of Random Forests

Random Forest is an ensemble machine learning method that utilize different decision trees and numerous statistical techniques, such as **Boosting** and **Bagging**, in order to perform either classification or regression forecasting tasks. The ensemble approach is a technique which incorporates the forecasts of several machine learning algorithms to make accurate forecasts than any individual model, thus a model consisting of numerous forecasting models is termed as an ensemble model. Together with boosting, bagging is one of the most commonly used ensemble strategies, in order to overcome high bias and high variance. Boosting bears upon a category of algorithms, that utilize weighted average technique in order to produce strong learners. In simple words, boosting and “teamwork” are synonyms. More specifically, each one of the numerous running models, determines sequentially where the next running model is going to focus on, in terms of feature selection. So each model is learning from the previous one, thus boosting the learning process as the name indicates. On the other hand, **Bagging** is a technique based on random sampling with replacement of a dataset’s subset. In general, bagging is used for variance reducing for algorithms that present high variance, such as decision trees. In the above mentioned technique, each model runs independently and in the end model’s outputs are aggregated without indicating any preference to a specific model. Instead of simply averaging the forecasts of the trees, a Random Forest incorporates the following two core concepts:

1. Random sampling of the training set’s data, when creating the trees.
2. Random feature subsetting concerning node splitting.

More specifically, a RF creates numerous decision trees and combine their forecasts, in order to conclude in a prediction, that is more reliable and accurate in contrary with just using an individual decision tree’s forecast. Each tree incorporated in a random forest model, is trained based on random sampling with replacement procedure, or in other words bootstrapping, so some samples of the training set may be used more than one time in a single tree. The basic concept is that by training each tree on random samples, even though each tree could appear high variance regarding a particular subset of the training data, the random forest will have lower variance overall however not at the expense of increasing the model’s bias. Another principal idea of random forests, is the fact that to each tree in the forest, only a subset of the features is available to make a decision, during the splitting nodes procedure.

A good machine learning model’s property, is the ability to generalize adequately to data that it has never seen before. The phenomenon of overfitting is observed, when the training data are memorized by an elastic model, which leads to fitting it extremely closely. Overfitting can create many complications in forecasting, because when overfitting exists, the model not only memorizes the dataset’s actual relationships but also the model memorizes also the noise that may exist in the dataset. In flexible or elastic forecasts models, high variance is appearing because the parameters or in other words the decision tree’s structure will vary considerably with the training data. To the contrary, concerning inflexible models they present high bias, due to the fact that the models make

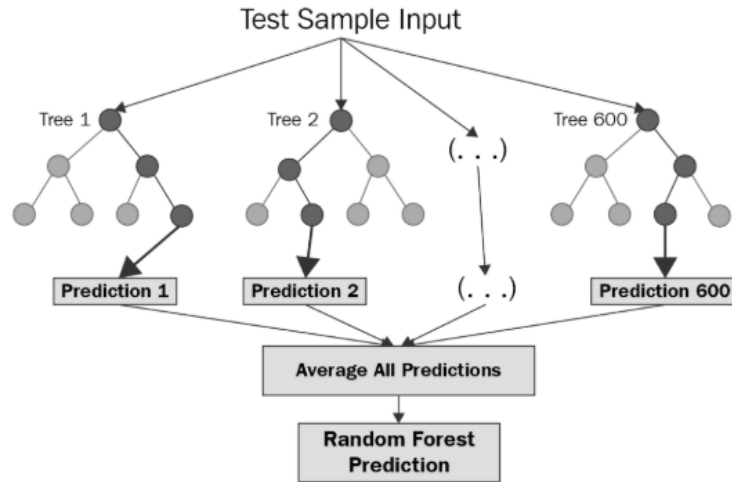


Figure 8.1: Random Forest Structure

assumptions about the training data. Moreover, in the end an inflexible model may not be able even to fit the available training data, thus whether the model presents high variance or whether the model presents high bias, the model is unable to generalize well to data never seen before.

The compromise among building a model that is so elastic that memorizes the training data as opposed to build an inflexible model that cannot be trained on the available data, is referred to as bias-variance tradeoff and is a fundamental concept in machine learning. In the following sections, we will break down the Random Forest algorithm and explain bit-by-bit its process:

- **Stage 1.** First of all, we sample repeatedly from the training set in order each observation in the dataset to have the same likelihood of being selected, having in mind that the sample size must be equal to the training set's size.
- **Stage 2.** Then, based on the bootstrap samples collected in the previous phase, the Random Forest model makes the forecasts for each sample.
- **Stage 3.** Finally, the ensemble forecast is produced by averaging each one of the above tree's forecasts, thus resulting in the model final forecast.

The principal idea behind Random Forest algorithm, as we mentioned earlier, is the combination of the forecasts of numerous decision trees into one forecasting model. Individual forecasts provided by the numerous decision trees may be unreliable and incorrect, but when combined together after averaging the forecast, the final forecast will be closer to the actual observation. Each individual decision tree is trained on a different random sample of the training data, thus each tree contributes with a different way in the forecasting procedure. In case we used only an individual tree, we would incorporate in our forecast only the limited information contain available in that tree's training sample, whereas by combining numerous forecasts of the different available trees, the incorporated information on our final model would be greater thus making our forecasts more accurate. In case each individual tree used the same training sample, then the result would be very vulnerable to outliers, whereas the use of the forest and different training samples results in more accurate overall predictions. Concerning forecasting, the Random Forest regressor averages individual trees forecasts, in order to reach the final forecast.

In Random forests the individual decision trees run in parallel and furthermore no interaction exists between them. Random Forests run by building numerous decision trees during the training process and then produces the forecasts. In case the problem under study is a classification problem, then the Random Forest outputs the mode of the forecasted classes of all the individual trees, whereas when we are referring to a regression problem, the Random Forest outputs the predictions' mean based on the outputs of all the individual decision trees. Random Forests are also called meta-estimators, e.g they aggregate the result of multiple forecasts by combining multiple decision trees, with the following helpful modifications:

1. The number of the available features that can be split on at each node, is constrained based on a certain percentage of the total amount of features. This certain percentage is part of a procedure that is called hyperparameter tuning. The above mentioned procedure guarantees that the ensemble model does not rely solely on any particular feature and that all possible predictive features are used sufficiently.
2. During the splitting procedure, each individual tree selects from a random sample from the training set, thus adding a further aspect of randomness in the model in order to prevent overfitting.

One of the most accurate learning algorithms available in literature is Random Forests, concerning both Classification and Regression tasks. Random Forests run efficiently on large databases and they are capable of handling thousands of input variables without removing features. Furthermore, as the Random Forest building progresses, an unbiased calculation of the generalization error is generated. Last but not least, they are capable of calculating missing data but also producing accurate forecasts even in case a large percentage of the data are missing. On the other hand Random Forests have some disadvantages too. Unfortunately, they tend to overfit when dealing with noisy datasets. Moreover, concerning data that include categorical variables with numerous levels, Random Forests have been observed to be biased towards features with more levels, thus making the variable importance scores unreliable.

8.2 Random Forest Regression

As we have already mentioned, the Random Forest algorithm can also be applied to regression problems. In a classification task, the model allows predicting the existence of observations in a class, based on quantitative and/or qualitative variables. In regression (continuous response variable): The model allows the creation of a predictive model for a quantitative response variable based on quantitative and/or qualitative variables. Regression is a widely used statistical modeling technique which aims to examine the existence of correlation between a dependent variable and one or more independent variables. Regression assumes that certain data match some common types of functions and thereafter determines the function that models the given data optimally. Regression applications are the modeling of tasks such as load demand forecast, wind speed concerning temperature, humidity and other weather factors etc.

As we mentioned in the beginning, Regression is a widely used statistical modeling technique which aims to examine the existence of correlation between a dependent variable and one or more independent variables. Regression models include the following variable types:

- Unknown correlation parameters declared as b (vector).
- Independent X variables (vector).
- The dependent variable Y or else the response variable.

A regression model associates Y dependent variable with X and b based on the regression function, $H = F(X, b)$. Regression Analysis helps us understand the change in response variable Y when one of the independent variable X changes, while the other independent variables are kept constant. Usually, the causal effect of one variable on another is to be determined. Such an example, is the investigation of the impact of product price increase on supply/demand. In other words, we collect data concerning the variable of interest and then we run regression analysis in order to measure the quantitative effect of independent variables on the response variable. The statistical significance of the estimated correlation coefficients, e.g the confidence that the actual correlation value is close to the correlation that we estimated, is also assessed.

Random Forests algorithm, uses a combination of the bagging algorithm and the random selection of variables in order to de-correlate individual decision trees. In the figure below, we present the pseudo-code of Random Forests in order to understand bit by bit the different stages of the algorithm. Firstly, the algorithm determines the number of m samples to be aggregated and thereafter all these m forecasting models are aggregated thus providing a reliable forecasts with lower variance. Instead of selecting all the available features during the node splitting procedure, a random selection of k features among the total number of features at each split is performed. With this procedure we ensure that only the feature with the best splitting performance feature

```

1 Select the number of models to build,  $m$ 
2 for  $i = 1$  to  $m$  do
3   Generate a bootstrap sample of the original data
4   Train a tree model on this sample
5   for each split do
6     Randomly select  $k$  ( $< P$ ) of the original predictors
7     Select the best predictor among the  $k$  predictors and
       partition the data
8   end
9   Use typical tree model stopping criteria to determine when a
     tree is complete (but do not prune)
10 end

```

Figure 8.2: Random Forests pseudo-code

can be chosen to split the data, thus de-correlating the individual trees by introducing this kind of randomness when building the individual trees. Concerning model tuning, there are two primary parameters in the Random Forest algorithm and more specifically these are: 1) the number of aggregated samples m and 2) the number of randomly selected features, k . In general, the bigger the number of trees, the heavier the computational burden. Intuitively, a Random forest with a large number of trees ($m \geq 1000$) is suggested with $k = \sqrt{p}$ or $k = \log(p)$, where p is the total number of the dataset's features. This randomly selected features is a subset of the total number of available features and despite the fact that the number of trees m in Random Forest algorithm is much bigger than that in the bagging tree, the Random Forest is still considered a more efficient algorithm. Moreover, the Random Forests are also considered to be efficient, accurate and highly reliable and are capable of dealing with datasets containing numerous variables. Furthermore, the relative importance of predictors can be deduced, even if the correlation among predictions and the tuning parameter k affects heavily the forecast. Last but not least, the Random Forest algorithm can efficiently handle datasets with large number of missing data and provide highly accurate forecasts. On the other hand, Random Forests are not able to provide accurate forecasts when the response variable value to be predicted is out of the range of the observed values in the training data.

The Random Forest Regression (RFR) algorithm was proposed and applied to quantify complex nonlinear relationships without leading to the obvious overfitting risk. RFR has been demonstrated to be more computationally cost-effective than many popular machine-learning algorithms. It has been widely used for predictions of stock price, electricity load demand, property sale price etc. The Random Forest Regression is a non-parametric ensemble learning algorithm that constructs numerous standard decision trees during the training process and outputs mean prediction of the individual trees, as seen in the following figure.

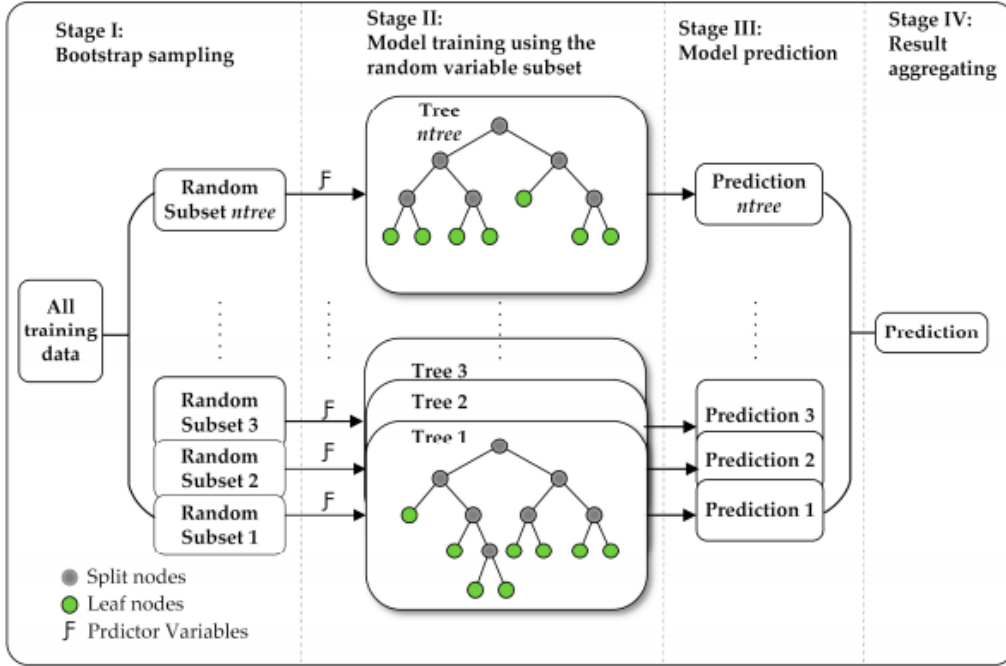


Figure 8.3: Random Forest Regression Algorithm

A decision tree is a hierarchical analysis diagram where each internal (split) node represents a test function on one independent variable, each branch represents the test outcome and each terminal (leaf) node represents a decision. Specifically, at each internal node, the algorithm searches the values of the incoming dataset and recognizes a threshold for one predictor variable to split the dataset such that the homogeneity of dependent variable values in each branch is maximized. In the RFR, each decision tree is trained using a subset of data randomly sampled with replacement from the original training dataset, which can increase the robustness against overfitting. In order to inject an additional layer of randomness, instead of using all variables, only a subset of randomly selected variables are considered to form the split nodes of each tree. The reason to add this randomness is to reduce the redundancy of predictor variables while increasing the diversity of the individual trees in a forest. The final result of RFR is decided by aggregating predictions of each individual tree. The RFR algorithm has four steps described as follows:

1. Use the bootstrap method to produce n_{tree} subset samples from the original training dataset, where n_{tree} is the number of trees to grow.
2. Grow regression trees on each bootstrap sample, during which, randomly draw a subset containing m_{try} predictor variables at each splitting node and determine the optimal split based on this subset of variables only. This process is conducted recursively until a stopping criterion (nodesize) is reached.
3. Obtain regression predictions over n_{tree} decision trees. For each individual tree, the prediction is the mean of the response variable's values at the corresponding leaf nodes.
4. Compute the final prediction by averaging n_{tree} predictions in the forest.

In the field of machine learning, a hyperparameter is a parameter whose value is tuned either to improve model performance or to decrease the time and memory cost of running. The model performance is the only aspect considered in this paper. The three main hyperparameters in RFR that can influence the model's performance are n_{tree} , m_{try} and nodesize. As mentioned earlier, n_{tree} is the number of trees in a forest. Larger n_{tree} can increase the stability of models and predictor importance estimates while decreasing the degree of overfitting, but requires more computational resources. The second hyperparameter m_{try} is the number of variables available for splitting at each tree node, which determines the diversity among decision trees. Lastly, nodesize is the minimal size of the leaf nodes, which is used as a stopping criterion and controls the depth of the trees.

Two strategies are widely used to validate models when tuning hyperparameters in machine learning. First is k-fold cross validation (CV), in which multiple rounds of validation are performed using different partitions of a dataset, and the final result is averaged over the k rounds to evaluate the model performance. The second strategy is to evaluate trained RFR using out-of-bag (OOB) data, which is the data (approximately one third of total data) not drawn by bootstrapping for growing an individual tree at the training stage. Usually, CV is more reliable since it overcomes the fact that the model performance can be very sensitive to how training and validation subsets are divided, but OOB requires shorter runtime. To reduce the risk of overfitting, this study combines these two strategies for searching the optimal hyperparameter sets, which has two steps described as follows:

- For each round of CV, tune hyperparameters using the grid search method to minimize the OOB error within the training dataset.
- Assign the hyperparameter combinations with the lowest OOB errors to each of the k models and test their performances using their corresponding validation datasets. The hyperparameter combination that has the best model performance is the final model configuration.

Chapter 9

Gradient Boosting Machines: LightGBM

9.1 Basic Concepts of Gradient Boosting

In literature, Boosting is thought to be a procedure for converting weak learners into strong ones. In other words, Boosting is an ensemble method for increasing model's accuracy independently of associated learning algorithm. The principal idea behind Boosting, is that each tree is fitted on some different part of the training dataset, thus weak learners are trained in a sequential way, while in parallel each learner attempts to correct its predecessor forecast. At every iteration, a new weak learner is trained with respect to the residual error of the whole ensemble till that moment. On the other hand, methods mentioned in the previous chapters, like Random Forests and Bagging are based on the concept of constructing an ensemble of models, where every individual model generates the forecast and then the ensemble of models just estimates the average of the forecasted values. Boosting technique authorizes ML models to increase their forecasting accuracy, nonetheless boosting algorithms are easy to implement and allows different model designs which makes the selection of loss function a matter of trial and error. In general, the selection of the loss function may be arbitrary and it's the researcher's responsibility, but we should have in mind that when the loss function is very simple - e.g squared-error- , the learning procedure could result in continuous error-fitting.

The family of boosting techniques is based on numerous constructive strategies of ensemble formation and the underlying engine incorporated for Boosting algorithms can be one of the following:

- AdaBoost (Adaptive Boosting)
- Gradient Tree Boosting
- XGBoost

In **Gradient Boosting Machines (GBM)**, the training procedure continuously fits models in order to generate more accurate forecasts. The principal concept of this algorithm is to construct new base-learners which are greatly correlated with the negative gradient of the ensemble's loss function. GBMs are highly flexible and customizable and furthermore they show substantial accuracy not only in real world scenarios but also in numerous machine-learning competitions. Although Random Forests construct an ensemble of deep independent trees, GBMs construct an ensemble of weak sequential decision trees, where each individual tree learns and improves on the previous tree. In the end, all these sequential weak learners (trees) form a powerful "committee" that in general provides very accurate forecasts.

The Gradient Boosting algorithm can be easily interpreted by firstly introducing the AdaBoost Algorithm. Gradient Boosting and AdaBoost, both add sequentially predictors to an ensemble where each one is correcting its predecessors. The main difference between Gradient Boosting and AdaBoost, is the way that the algorithms identify the shortcomings of the decision trees. AdaBoost model identifies the shortcomings by using large weight data points, whereas Gradient Boosting performs the same procedure by incorporating gradients in the loss function $y = ax + b + \epsilon$ where

ϵ is the error term. A major difference is that Gradient Boosting fits the newly added predictor to the residual errors generated by the preceding predictors, where on the other hand AdaBoost adjusts the weights for each incorrectly forecasted observation at each iteration. Gradient Boosting repetitively leverages the patterns in residuals and when reaching a point that residuals do not follow any pattern that could be modeled, the modeling stops (in order to avoid overfitting). Last but not least, one of the unique features of Gradient Boosting is the fact that allows the incorporation of a user-specified loss function, which may be more appropriate in real case scenarios. In the following section, we will break down the GBM algorithm and explain step-by-step its process:

- Fit a model, $F1(x) = y$
- Fit a model to the residuals, $h1(x) = y - F1(x)$
- Construct a new model, $F2(x) = F1(x) + h1(x)$
- Combine more and more weak learners sequentially, thus producing the final model able to account for a large amount of the original model's error and reduce this error as time passes by.

9.2 Gradient Boosting Decision Trees

Gradient Boosting Decision Trees (GBTS) combine the forecasts of numerous decision trees by adding them together, in order to provide forecasts that generalize well. GBTS is a computationally efficient technique to capture correlations/interactions among model's variables by utilizing decision trees. More specifically GBDTs are trained iteratively in a way that the GBDT forecasts the response variable by firstly training weak decision tree on the data. The above mentioned trees are parameterized by the number of splits, or in other words the interaction depth. Every individual split denotes the splitting rule over each explanatory variable, thus modeling the interactions among the input variables. Furthermore, each decision tree aims to minimize the specified error function by recursively splitting the data in such a way to maximize some criterion's threshold, like the tree's depth. That criterion is selected based on the idea, that the loss function is minimized in each split. Last but not least, estimating the best available split requires the decision tree to test various splits and calculate the criterion's value for each split.

A special case of decision tree with only one split or two terminal nodes is the so called tree stump, which many practical applications small trees and tree-stumps provide considerably accurate results. Additionally, there is experimental evidence in literature that complex trees (e.g tree depth > 20) provide hardly any benefit in contrary with compact trees (e.g tree depth ≈ 5). It's worth noticing that we can fit an additive model of weak tree base-learners by utilizing tree stumps. A salient feature of DTs is that by default a single decision tree always extrapolates the loss function with the constant value. There are two approaches that can be incorporated when training each individual decision tree and splitting the data: **level-wise** and **leaf-wise**. The level-wise approach generates a balanced tree, where on the other hand the leaf-wise approach make the splitting that mostly reduces the error function. Furthermore, the level-wise training approach resembles a form of regularized training since leaf-wise training method is able to construct any tree that level-wise training can ,thus making the model more flexible but also the risk of overfitting increases, whereas the opposite is not possible. To conclude, the main challenge in training a Gradient Boosting Decision Tree is the procedure of identifying the best split for each leaf, although no known method is able to identify the best split without having to go through all the feature's available observations.

9.3 LightGBM

The Gradient Boosting Decision Tree (GBDT) and more specifically one implementation of the GBMT algorithm - called XGBoost – is thought to be one of the best performing algorithms in Kaggle’s machine learning competitions. Although XGBoost seemed to be the king of algorithms in Kaggle for the time being, a new challenger is rapidly gaining momentum and this is the **light-GBM**. This algorithm was released by Microsoft and is claimed to provide better forecasting performance for the same running time than XGBoost. LightGBM is an outstanding implementation that is very similar to XGBoost, but the main difference is in particular the way trees are formed. Light GBM is a fast, distributed, high-performance gradient boosting framework based on decision tree algorithm, that can be used for regression, classification and numerous other machine learning tasks. Since LightGBM is based on decision trees, it splits the tree by utilizing the leaf-wise approach having the best fit, whereas other boosting algorithms split the tree level wise. The leaf-wise approach can reduce the loss to a greater extent than the level-wise approach, thus provides very accurate forecasts. Furthermore, LightGBM, as it’s name Light suggests, is **surprisingly fast**. The utilization of Leaf-wise splits, increases model complexity and raise the risk of overfitting. As mentioned earlier, overfitting can be avoided by setting a threshold to the max-depth parameter (e.g the depth where splitting takes place).

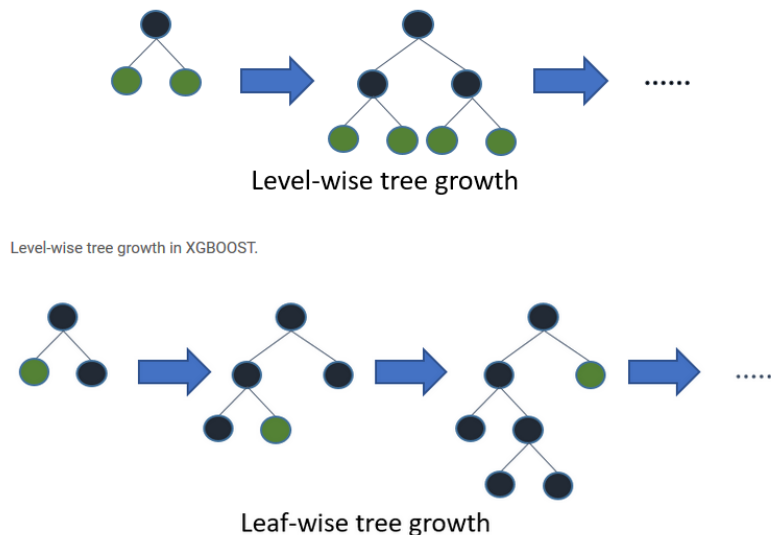


Figure 9.1: Splitting data: Leaf-wise VS Level-wise

Numerous advanced users on Kaggle already incorporate LightGBM for each new competition, as it gets more and more coverage. Concerning parameter tuning, LightGBM offers some different parameters. but in general the majority of them are similar to their XGBoost counterparts. Concluding, we are going to list some of the LightGBM biggest advantages and these are:

- Faster training and increased efficiency: LightGBM buckets continuous variables into discrete bins which makes the training procedure faster. In other words, LightGBM utilizes a histogram based algorithm.
- Reduced memory usage: Replacing continuous values to discrete bins, results in memory usage reducing.
- Higher accuracy: Due to the use of leaf-wise approach, it generates much more complex trees and this makes it more accurate than any other boosting algorithm by providing very accurate forecasts. Although, it should be noted that this approach can sometimes lead to overfitting which can be avoided by specifying the max depth parameter.
- Compatibility with large datasets: It’s very efficient with large datasets, with a significant reduction in training time comparing to XGBoost.
- Parallel learning.

Chapter 10

Forecast Performance Measures

10.1 Basic Concepts of Prediction Accuracy

In simple words, when we are referring to "forecast accuracy", it's a measure to estimate how similar are the forecasted values in contrary with the actual ones. Concerning electric load demand time-series, this measure of "closeness" is calculated by forecasting load demand at referenced time. During the forecasting procedure, the actual values are unknown, but in order to select the optimal available algorithm and build our forecasting model, the model's accuracy is estimated by forecasting load demand for a past time interval, because actual historical data are available for the past. The above described procedure is called in-sample performance or ex ante analysis, in contrary with the out-of-sample performance or a posteriori analysis where the model's accuracy can only be evaluated after the forecasted fact takes place. In general, there are numerous factors that affect forecast's accuracy and most of them are listed below:

- **Volatility** of the forecasted phenomenon: Higher volatility index means reduced forecast accuracy.
- **Model Uncertainty-Overfitting**: When the same data used for fitting and testing, then bias exist in the model and the phenomenon of overfitting arise.
- **Aggregation**: According to literature, aggregating data leads to lower volatility in data, thus aggregating electric load demand over longer intervals (e.g. hourly instead of 15-minute intervals) leads to higher forecast accuracy. The above mentioned aggregation procedure is referred to as temporal aggregation.
- **Forecast horizon length**: The further away we move from the forecast origin (the current time step), the more the forecast's accuracy decreases.
- **Algorithm selection**: A forecast model based on an algorithm that describes better the scenario under study, yields in more accurate forecasts.

The uncertainty around a forecasted value is expressed through the prediction interval. In simple words, the prediction interval defines in a probabilistic way the boundaries where the future observed values are expected to fall within. For example, for a 98% prediction interval it is expected the actual observations to fall within, with a 98% probability. A model's forecast evaluation, can be realized based on different accuracy metrics that consider forecast errors over the specified forecast horizon, summarized into a single value, which is called forecast performance measure or forecast accuracy metric. Counter intuitively we can understand that a lower accuracy measure means better prediction performance for the model.

10.2 Forecast Performance Measures

In the previous chapters, we studied various state of the art algorithms for time-series forecasting. Now the next step, is to apply these methods for building forecast models and making predictions. After creating a forecasting model but before applying the model to time-series data, we have to partition the data into two independent parts, the Training Set and the Test Set. The observations contained in the training set are used for constructing or in statistical language for fitting the selected model. In some cases, a small subpart of the data contained in the the Training set is kept for model validation purposes and this subpart is referred to as the Validation Set. Most of the times, after data transformations and preprocessing -such as log transformation, data scaling etc - is done the constructed model is used for making forecasts. The data contained in the Test set will be used for evaluating the accuracy of the fitted model when forecasting the observations in the test set. In order to evaluate the model's forecasting performance when comparing different models based on different algorithms, their relative forecasting performance on the test set is considered. We meet time-series tasks in many business situations, thus the importance of time-series forecasting and the optimal modeling of time series data is fundamental. Having that in mind, various forecasting accuracy measures, also referred as **performance metrics**, are proposed in literature in order to be able to compare forecasting models based on different algorithms. More specifically, these metrics are based on some function of the time-series' observed and forecasted values. In the following sections, we provide a brief description on the most common performance metrics used in literature and their notable features. In each of the following formulas, y_t is the observation's actual value at time t , f_t is the forecasted value at time t , $e_t = y_t - f_t$ is the forecasting error and n is the size of the test set.

10.2.1 MFE-Mean Forecast Error

The Mean Forecast Error(MFE) is the simplest scale-dependent accuracy metric and is formulated as

$$MFE = \frac{1}{n} \sum_{t=1}^n e_t$$

MFE detects bias in the forecasts (e.g overfitting), but when bias is not present in the forecast, it tends to ignore huge individual errors completely, due to the averaging process that takes place. Some of the MFE's key features are:

- MFE measures the average deviation between the actual values and the predicted ones.
- MFE does not penalize large errors and moreover it is heavily depended on data scale and any data transformations.
- MFE cancels out the sign of errors and we can not estimate their exact amount. In other words, a zero MFE does not mean that predictions contain no error but that errors cancel out.
- Last but not least, a good forecast means a MFE close to zero.

10.2.2 MAE-Mean Absolute Error

Metrics that use absolute values of the forecasting errors, have the ability to detect the magnitude of errors in a more efficient way. Such a metric is the mean absolute error which is defined as follows:

$$MAE = \frac{1}{n} \sum_{t=1}^n |e_t|$$

Some of the MAE's key features are:

- MAE measures the average absolute deviation between the actual values and the predicted ones. Thus MAE sometimes may be referred as Mean Absolute Deviation (MAD).
- MAE presents the magnitude of overall forecasting error, so positive and negative errors do not cancel out.

- With MAE, due to absolute values, we cannot infer anything about forecasting error direction (over-predicting or under-predicting)
- A good forecast means a MAE as small as possible.
- Last but not least, like MFE, MAE does not penalize large errors and it's heavily affected by data scale and data transformations.

10.2.3 MAPE-Mean Absolute Percentage Error

The mean absolute percentage error metric at time t_i is scale-independent, so it's a suitable metric for evaluating a model's forecasting performance independently of the data under study. Furthermore, the MAPE, by using the absolute that we mentioned earlier, does not suffer from the deficit of Mean Percentage Error, where unless biases arise during the forecast, signed percentage errors will cancel out, resulting in inability to detect large percentage errors. This measure is one of the most useful measures for practical purposes and its given by

$$MAPE = \frac{1}{n} \sum_{t=1}^n \left| \frac{e_t}{y_t} \right| * 100$$

Some of the MAPE's key features are:

- MAPE denotes the percentage of average absolute forecasting error.
- MAPE is scale-independent, although it is heavily affected by data transformations.
- With MAPE again due to absolute values, we cannot infer anything about forecasting error direction.
- MAPE does not penalize extreme errors.

10.2.4 MSE-Mean Squared Error

The mean squared error (MSE) metric, measures the average squared difference between the forecasted values and the actual ones, or in other words MSE estimates the average of the squares of the forecasting errors. The Mathematical formulation for this metric is expressed as

$$MSE = \frac{1}{n} \sum_{t=1}^n e_t^2$$

Some of the MSE's key features are:

- MSE measures the average squared deviation between the actual values and the predicted ones.
- MSE gives an abstract idea concerning the forecasting errors, because the opposite signed errors do not cancel out.
- MSE due to forecast error squaring, is heavily affected by large errors and penalizes such occurrences during the forecasting procedure.
- With MSE we cannot infer anything about forecasting error direction and moreover MSE is sensitive to data scaling and transformations.
- Last but not least, MSE may be a decent measure to estimate overall forecasting error, but it is not easily interpretable.

10.2.5 SSE-Sum of Squared Error

SSE is the sum of the squared errors between the value of each observation and the observation's group mean value. SSE is mainly used as a measure of variation between observations belonging in the same cluster. In case all observations within a cluster are identical, then the SSE would then be equal to zero. The Mathematical formulation for this metric is expressed as

$$SSE = \sum_{t=1}^n e_t^2$$

SSE's key features are:

- SSE measures the total squared deviation between the actual values and the predicted ones.
- Concerning other properties, SSE presents the same properties as MSE.

10.2.6 SMSE-The Signed Mean Squared Error

The Signed Mean Squared Error metric is mathematically formulated as follows

$$SMSE = \frac{1}{n} \sum_{t=1}^n \left(\frac{e_t}{|e_t|} \right) * e_t^2$$

and its properties are:

- SMSE measures the same thing as MSE, but this time the error's sign is preserved for each individual squared error.
- SMSE penalizes large forecasting errors, but in contrary to MSE, from SMSE error metric we can infer the forecasting error's direction (over-predicting or under-predicting)
- Last but not least, just as with MSE, SMSE measure is sensitive to data scaling and transformations.

10.2.7 RMSE-Root Mean Squared Error

The root mean squared error (RMSE) is a quadratic scoring function that measures the forecasting error's average range, by calculating the square root of the average of squared deviations between predictions and then actual observations. The RMSE is expressed with the following formula

$$RMSE = \sqrt{\frac{1}{n} \sum_{t=1}^n e_t^2}$$

The root mean squared error is just the square root of the calculated MSE, thus all the properties of MSE hold for RMSE as well.

The right choice of the forecasting accuracy metric which will be later used to evaluate the forecasting models is highly significant. First of all, based on that metric we will choose the optimal forecasting model algorithm for our case study and furthermore it is important to mention that for different problems there may be different forecast accuracy metrics more suitable. The Mean Absolute Error (MAE) and the Root Mean Squared Error (RMSE) are the most commonly used and the most reliable measures concerning forecast evaluation of continuous variables, such as the electric load. Both MAE and RMSE express the model's average forecasting error in the same unit as the response variable's unit, thus making the result rather intuitive and easily interpretable. Both of these error metrics range from 0 to ∞ , e.g they are almost always positive and they are indifferent concerning the forecasting error's direction. Last but not least, both of them are negatively-oriented error metrics, thus lower forecasting error means better forecasting performance. On the other hand, estimating the square root of the average of squared errors presents interesting indications concerning the RMSE metric. Since the forecasting errors are squared before they are averaged, thus the RMSE function assigns relatively large weights to large forecasting errors. In other words, RMSE has the benefit of penalizing large errors more efficiently and moreover outputs a forecasting error in the same unit as the response variable's unit. Thus, we find the RMSE to be the most appropriate evaluation measure for our models and this is the one that we are going to use in our experiments.

Chapter 11

Experiment-ML Models for STLF in Greek Electricity Network

11.1 Goals & Environment

The real value of this thesis is the development of a forecast model that could be used by Energy Analysts in the daily activity of hourly load declaration to the IPTO's platform. More specifically, private company's energy analyst is obliged by the regulation authority to declare the next days load demand per hour based on their forecasts for their market share. OoEM publishes daily the next day's forecast concerning the hourly load demand for the whole Greek electricity grid. This forecast is comprised of 24 numbers in KWh, denoting the load demand for each hour of the day. This hourly forecast value, represents the network's total load demand, which could be graphically represented as a curve, where on the y-axis is the load in KWh and in the x- axis are the 24 hours of the day under study. In general, a plot depicting the variation in electric load demand with respect to time is known as **load curve**. In other words, the load curve is the graphical representation of electric load demand in time sequence. When this curve is plotted over a 24 hour period it's called a daily load curve, whereas when it's plotted for a week, month or a year it is called weekly, monthly and yearly load curve respectively. A power system's load curve is not static and thus not the same during each time interval, as it presents differences and fluctuations (daily, seasonal etc). In general, a load curve is categorized in two types, the summer load curve and the winter load curve.

The load curve depicts the electrical load demand over a specified period and gives an insight into the consumption/ load demand pattern for the power system under study. Currently, the knowledge of a power system's load curve is considered to be a fundamental factor in order to ensure the system's efficient operation. Based on the load curve, we can accurately determine aspects such as planning and controlling a system's function, accurately calculating the electricity distribution associated costs etc. Last but not least, the electric load demand curve should be adjusted to the electric energy generation curve to avoid power quality problems due to the wave quality and supply reliability. In general, from a load curve we can obtain the following information:

- A load curve determines the diurnal load demand fluctuations.
- By observing the load curve, we can infer the peak load demand value, which also determines the power system's maximum demand time.
- The area under the load curve, denotes the total amount of energy generated in the period under study. Moreover, if we divide this value by the total number of hours in the period under study, we obtain the power system's load.

More specifically, the network's load curve -as we will also see in the datasets part later- could be broken down to three other voltage curves, which denote the voltage type of the electricity meters under study, the High Voltage load curve, the Medium Voltage load curve and the Low Voltage load curve. All three sum up to the total load curve, which is the one that OoEM forecasts and publishes on daily basis. High Voltage meters in Greece are considered very demanding -in terms of consumption- industries, wind farms etc, thus there are very few HV meters in the network.

Medium Voltage Meters are considered industries, airports, big offices etc and right now there are approximately 800 MV meters in the network, represented by PPC or other energy providers. Last but not least, there is the Low Voltage meters that are the most tricky ones, when it comes to load forecasting/declaration as we will see later on. In this type of voltage are categorized the vast majority of the Greek electricity grid's meters, containing home meters, small and medium enterprises etc and in general LV meters **DO NOT use telemetry**.

Concerning the High Voltage load curve forecasting, we could state that it is a non-troublesome task, because HV meters use telemetry measurement system and in general we do not expect large discrepancies between forecast and actual load values in High Voltage. As for the Medium Voltage load curve, almost all of the Medium Voltage meters in Greece use telemetry measurement system, so again we have accurate data that could lead in a very good forecast, depending on the estimation approach used, because Medium Voltages are mainly industries, companies' offices etc. which are highly affected by seasonality and Holidays. Apart from the fact that telemetry is used, for Medium Voltage, IPTO also sends to the energy providers historical data of this meters load consumption in 15 minute intervals, thus giving the opportunity to the provider to analyze each individual meter's behaviour, see how much is affected by Seasonality or Holiday and thus making better forecasts. With telemetry, utility companies and electricity providers get previously unavailable valuable insights to make data-driven business decisions. By utilizing power consumption telemetry systems, we are able to perform real-time usage monitoring, identify and analyze /demand patterns, develop utility plans and detect abnormalities in power supply. For example, we expect a hotel to have the highest consumption in summer months when is "high season" than in winter, thus we have to adjust our forecast on load demand based on seasonality.

As we said earlier, thanks to machine learning and power consumption telemetry, energy analysts are able to:

- Identify trends and patterns in power consumption and research the electricity usage behavior.
- Identify energy usage patterns and build data-driven approach in order to optimize power consumption and save resources.
- leverage historical data concerning load demand and usage, in order to obtain a broader picture and statistics for research purposes.

Nevertheless, the real struggle is when we want to make a forecast concerning the daily Low Voltage load curve. As we underlined earlier, among others, energy providers have to make the forecasted load declaration specifically for low voltage per hour, but in this case analysts encounter two major difficulties. First of all, Low Voltage meters are the vast majority of electricity meters in Greece as under this category there are meters such as homes, small and medium enterprises etc. which are obviously greatly affected by seasonality, weather and Holiday. In the total grid's load, load demand for Low Voltage meters is on average the 62% of daily total load demand, so it's easy to conclude that is very crucial to be able to accurately forecast low voltage curve on daily basis. Furthermore, in contrary with the Medium Voltage meters, there aren't many historical data concerning the consumption of Low Voltage meters -and if there are we dispute the validity of them-, because LV meters do not use the telemetry measurement system. Nonetheless, as we said earlier energy analysts in energy providers are obliged to declare next day's load demand forecasts for all the three voltage type categories separately, only for their market share (meters that are being represented by each one of the available energy providers in Greece). Thus to have an accurate estimation for Low Voltage load curve, we will apply the following approach. We will extract information and calculate the daily Low Voltage curve for the whole electricity grid based on past **actual** data and then we will make 24 forecasts - one for each hour of the day- for the next day's Low voltage load demand. The only known information to the provider for low voltage will be the forecasted load curve for the whole network and the percentage that each provider owns in the market share for low voltage, thus by applying this percentage to the total low voltage curve, each provider can make the hourly declaration for low voltage.

11.2 Development Platform

For this master thesis the Python Language was selected as the development software programming language. The Python 3.7.3 version with Jupyter Notebook as IDE platform. The thesis main

goal is to construct & compare different Machine Learning forecasting models, to analyze Greek network's Low Voltage electricity load demand and provide accurate predictions. The code for this master thesis can be found on GitHub (<https://github.com/Darzan21/LoadPrediction>).

11.3 Datasets

The datasets used in our case study originates from 3 different sources.

- The IPTO's Actual Load demand datasets.
- The OoEM's Load Prediction datasets.
- A calendar containing the Greek Holiday.

11.3.1 IPTO's Actual Load demand dataset

The dataset concerning the actual electricity load demand in Greece, can be found from the Greek Independent Power Transmission Operator (IPTO). More specifically, IPTO is responsible for Greece's electricity supply in an efficient and reliable manner, while also promoting the competitive development of Greek electricity market by creating value for all stakeholders. The IPTO's Actual Loads datasets are in hourly format and span from 2018-01-01 to 2019-07-31. More specifically, each of these excel files (one for each month) contain the following columns as shown also in the figure below.

- **DeliveryDay:** The date that the load value is referring to.
- **Hour_EET:** The hour that the load value is referring to.
- **Gen_Aux:** The load demand for the Generator Auxiliaries.
- **HV:** The load demand for High Voltage.
- **Pump:** The load demand for Pump systems.
- **Network:** The load demand for Low Voltage and Medium Voltage.
- **TEL_MV:** The load demand for Medium Voltage that uses telemetry.
- **Total:** The actual load demand in total for the whole Electricity Greek network grid.

deliveryDay	Hour_EET	Gen_Aux	HV	Pump	TEL_MV	Network	Total
2/1/2019	1	32.512	920.678	0.0	963.603	4,561.354	5,514.544
2/1/2019	2	32.272	955.990	0.0	940.037	4,024.992	5,013.254
2/1/2019	3	32.255	986.614	0.0	926.340	3,956.893	4,975.762
2/1/2019	4	30.350	996.793	0.0	919.220	3,803.450	4,830.593
2/1/2019	5	29.874	1,002.054	0.0	927.531	3,717.842	4,749.770
2/1/2019	6	36.490	1,001.668	0.0	961.822	3,823.480	4,861.638
2/1/2019	7	28.826	976.842	0.0	1,090.304	4,361.878	5,367.546
2/1/2019	8	22.119	949.259	0.001	1,288.426	5,193.399	6,164.777
2/1/2019	9	20.617	865.308	0.0	1,445.959	5,667.775	6,553.70
2/1/2019	10	28.902	819.796	0.0	1,514.567	6,032.183	6,880.881
2/1/2019	11	27.520	751.936	0.0	1,519.486	6,084.209	6,863.665
2/1/2019	12	24.854	717.710	0.0	1,508.517	6,030.643	6,773.207
2/1/2019	13	25.517	710.822	0.0	1,495.242	5,950.453	6,686.792
2/1/2019	14	28.685	705.833	0.0	1,468.537	5,831.897	6,566.416
2/1/2019	15	27.104	724.128	0.0	1,413.674	5,648.342	6,399.574

These data because there are actual historic data, are provided by IPTO on monthly basis but concerning the **Previous Month**. Thus, around the middle of month x, IPTO will provide the actual measurements for month x-1. Then IPTO himself but also the Greek energy providers (private companies), will know the error in their load declaration for the x-1 month. The TSO investigates the possibility of Demand Side Management (DSM) of electric load demand in the future, because the grid's peak load is going to be multiple times greater, in contrary with the grid's base load. The above described situation, will lead to unnecessarily high requirements for OoEM's distribution infrastructure and moreover yields in fines by RAE.

11.3.2 OoEM's Load Prediction dataset

OoEM (LAGIE in Greek) stands for Operator of Electricity Market in Greece and is the independent organisation that applies the rules of operation of the Electricity Market and more specifically it is responsible for the daily electric load demand planning. In other words, OoEM forecasts the next day's electricity load demand for the whole Greek electricity grid. This process is vital because OoEM needs to have an accurate estimation of next day's hourly load demand in order to prepare electric power units to produce the requested amount of energy or to be ready purchase energy on demand from the neighbouring countries. The forecasts are provided in daily basis and an excel file is filled with the load demand forecasted values, which contains the following columns as shown in the figure below.

- **Date:** The date that the load forecast value is referring to.
- **Hour:** The hour that the load forecast value is referring to.
- **Weekday:** The day of Week (starting from Monday) that the date is referring to.
- **Holiday:** Indicator of whether this day is a Holiday or not.
- **[1...24]:** 24 columns each one of them indicating the hour of the day and containing the forecasted IPTO's load demand value, **for the whole Electricity Greek network grid.**

Month	Weekday	Holiday	Date	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	1	0	Monday, January 1, 2018	5,315	4,963	4,838	4,515	4,280	4,182	4,202	4,277	4,573	5,152	5,765	6,130	6,163	5,808	5,237	5,265	5,359	5,446	6,117	6,237	6,198	6,021	5,756	5,513
1	2	0	Tuesday, January 2, 2018	5,023	4,612	4,573	4,389	4,250	4,319	4,629	5,072	5,417	5,782	5,973	6,051	6,116	6,108	5,946	6,047	6,126	6,491	6,927	6,918	6,831	6,517	6,106	5,717
1	3	0	Wednesday, January 3, 2018	5,211	4,761	4,721	4,547	4,438	4,506	4,945	5,504	6,057	6,430	6,518	6,599	6,642	6,619	6,476	6,626	6,736	7,026	7,553	7,554	7,409	6,949	6,411	5,962
1	4	0	Thursday, January 4, 2018	5,307	4,788	4,721	4,525	4,427	4,476	4,865	5,450	5,995	6,518	6,753	6,953	7,049	7,096	6,909	6,977	6,949	7,168	7,648	7,649	7,483	7,014	6,476	6,038
1	5	0	Friday, January 5, 2018	5,420	4,932	4,865	4,718	4,621	4,688	5,039	5,534	6,149	6,725	6,821	6,809	6,829	6,795	6,605	6,749	6,773	6,973	7,528	7,550	7,453	6,984	6,455	6,037
1	6	0	Saturday, January 6, 2018	5,430	4,952	4,885	4,680	4,514	4,474	4,531	4,550	4,818	5,163	5,492	5,812	6,027	5,916	5,420	5,464	5,493	5,706	6,380	6,469	6,401	6,117	5,755	5,550
1	7	0	Sunday, January 7, 2018	5,348	4,890	4,813	4,617	4,421	4,333	4,459	4,619	4,942	5,433	5,889	6,221	6,338	6,188	5,751	5,799	5,801	6,041	6,765	6,912	6,863	6,648	6,130	5,710
1	1	0	Monday, January 8, 2018	5,170	4,711	4,674	4,516	4,419	4,545	5,102	5,932	6,387	6,732	6,774	6,733	6,710	6,672	6,500	6,627	6,655	6,863	7,543	7,591	7,543	7,055	6,439	5,952
1	2	0	Tuesday, January 9, 2018	5,343	4,903	4,856	4,680	4,592	4,700	5,205	6,028	6,470	6,829	6,872	6,876	6,878	6,798	6,557	6,682	6,714	7,075	7,667	7,736	7,588	7,150	6,514	6,029
1	3	0	Wednesday, January 10, 2018	5,304	4,836	4,788	4,640	4,552	4,670	5,215	6,026	6,493	6,923	7,035	7,036	7,029	6,952	6,763	6,901	6,895	7,065	7,518	7,597	7,540	7,207	6,494	6,027
1	4	0	Thursday, January 11, 2018	5,310	4,851	4,794	4,648	4,559	4,685	5,181	6,014	6,553	7,042	7,141	7,197	7,190	7,048	6,718	6,821	6,816	7,052	7,534	7,574	7,545	7,076	6,403	5,964
1	5	0	Friday, January 12, 2018	5,337	4,889	4,810	4,654	4,576	4,673	5,160	6,003	6,510	6,851	6,928	6,880	6,875	6,801	6,602	6,775	6,802	7,000	7,442	7,482	7,434	7,035	6,322	5,932
1	6	0	Saturday, January 13, 2018	5,424	4,945	4,858	4,672	4,556	4,585	4,837	5,220	5,762	6,349	6,637	6,766	6,847	6,835	6,587	6,767	6,791	6,998	7,433	7,443	7,296	6,867	6,290	5,902
1	7	0	Sunday, January 14, 2018	5,405	4,946	4,897	4,702	4,556	4,536	4,700	4,901	5,312	5,900	6,491	6,875	7,027	6,892	6,486	6,542	6,576	6,692	7,189	7,335	7,189	6,868	6,360	5,990
1	1	0	Monday, January 15, 2018	5,258	4,769	4,731	4,574	4,507	4,621	5,157	6,068	6,608	7,177	7,369	7,410	7,459	7,400	7,146	7,296	7,240	7,318	7,859	7,928	7,919	7,430	6,718	6,201
1	2	0	Tuesday, January 16, 2018	5,623	5,136	5,087	4,941	4,824	4,920	5,437	6,291	6,773	7,224	7,321	7,301	7,301	7,189	6,942	7,095	7,083	7,269	7,865	7,991	7,933	7,387	6,724	6,188
1	3	0	Wednesday, January 17, 2018	5,525	5,048	4,989	4,814	4,716	4,833	5,339	6,226	6,708	7,054	7,107	7,033	6,986	6,903	6,731	6,864	6,850	7,082	7,620	7,737	7,669	7,211	6,518	6,088
1	4	0	Thursday, January 18, 2018	5,310	4,831	4,791	4,646	4,557	4,654	5,132	6,034	6,500	6,999	7,200	7,273	7,285	7,145	6,889	7,007	7,005	7,192	7,725	7,823	7,736	7,230	6,605	6,185
1	5	0	Friday, January 19, 2018	5,445	4,959	4,910	4,753	4,685	4,781	5,298	6,175	6,660	7,074	7,143	7,069	7,017	6,898	6,627	6,809	6,759	6,959	7,571	7,659	7,542	7,055	6,441	6,040
1	6	0	Saturday, January 20, 2018	5,557	5,078	5,049	4,863	4,736	4,755	4,968	5,328	5,837	6,366	6,585	6,606	6,626	6,573	6,294	6,426	6,459	6,649	7,193	7,272	7,108	6,707	6,181	5,840
1	7	0	Sunday, January 21, 2018	5,237	4,798	4,720	4,524	4,369	4,359	4,474	4,667	5,030	5,539	5,973	6,310	6,423	6,245	5,857	6,015	6,094	6,302	6,867	7,005	6,919	6,684	6,214	5,786

11.3.3 Weekdays & Holidays

As we will see later, load forecasting presents Hourly but also Weekly seasonality. Moreover, load demand is greatly affected by whether the Day under study is a Holiday or not (for example on Holiday, stores industries etc are closed, thus the load demand is decreased). Based on this, we also provide a calendar with the Greek Holiday.

11.4 Preprocessing & Feature Engineering

After fetching all the datasets, one vital step in any forecasting task is the preprocessing of the available data. First of all, we have to join the datasets in order to create one dataframe object containing only the appropriate columns. After doing some unpivoting to the dataset containing the OoEM's daily forecast in order to create the Hour column -instead of having 24 columns in the dataset-, the common fields among all the datasets are the Date and Hour. So an inner join in the common field which is "date - hour" was performed in order to join the two datasets. By looking at the IPTO's dataset which contains the actual load demand data, we do not observe a column for the Low Voltage load because it is not possible to be accurately measured by IPTO, although the energy suppliers are obligated in a daily basis to declare forecast for next days Low Voltage load demand. Thus, the aim of this thesis is to develop a model for forecasting the load demand for Low Voltage, that the energy provider's analysts could use, as-is for daily Low Voltage load declaration. So to sum up, we will focus on Low Voltage load forecasting.

As we underlined earlier there is no column for Low Voltage load demand on the dataset, but based on the experts guidance, we can estimate it by calculating the difference of the dataset's

columns Network and TEL MV. The Network column's load demand values, represent the load demand for Low Voltage and Medium Voltage, whereas TEL MV column contains the load demand for Medium Voltage that uses telemetry (almost all Medium Voltage meters in Greece use telemetry measurement system), thus the difference is the actual load demand for Low Voltage stored in the column LV, which is the column of interest that we want to forecast. In order to have a metric to evaluate our model concerning the Low Voltage demand, we are going to compare our forecasts with the daily forecasts provided by OoEM. Although, the OoEM publishes on a daily basis the next day's load demand forecast for the whole Greek electricity grid. To get an accurate forecast for the Low Voltage we will use the following approach. After creating the column with the actual values of the LV load demand value, we calculate the percentage of load demand corresponding to the daily LV curve (Low Voltage demand), in comparison to the total load demand for that day. The actual values of load demand curves are not known a-priori and are made available only one month after the time of occurrence, or in other words after the refinement of measurements has been made by IPTO. After calculating that percentage, we will apply it to the total load demand forecast given by OoEM in order to be able to make a good approach, of what would be the forecast of the OoEM's model if OoEM knew a-priori the exact proportion of LV load demand and forecasted specifically the Low Voltage load demand and not the total networks load demand. So that is the metric that we will use to compare our model's accuracy to OoEM's model accuracy for Low Voltage load demand. The datasets structure in it's premature form, after the joins, the feature engineering and keeping the columns of interest concerning only the Low Voltage load demand is shown in the following figure.

LV Weekday Holiday LVForecastEstimation				
Timestamp				
2018-01-01 00:00:00	3933.209685	1	1	3928.618104
2018-01-01 01:00:00	3615.862294	1	1	3584.743969
2018-01-01 02:00:00	3531.408073	1	1	3481.997027
2018-01-01 03:00:00	3250.926682	1	1	3171.205588
2018-01-01 04:00:00	3027.204711	1	1	2874.047836
...
2019-10-31 19:00:00	4050.675054	4	0	3961.302143
2019-10-31 20:00:00	3998.556006	4	0	3901.933643
2019-10-31 21:00:00	3681.134314	4	0	3598.459103
2019-10-31 22:00:00	3229.579195	4	0	3135.288660
2019-10-31 23:00:00	2814.278120	4	0	2808.415708

Last but not least, as we mentioned earlier, OoEM everyday provides a forecast for next day's load demand concerning the whole network. By examining OoEM's forecast (ForecastTotalLoad variable) for the whole network historically, we identified that this forecast is fairly accurate for the total load demand, thus we decided that we should incorporate this variable in our models as explanatory variable, as there is plenty of information there. Although as we mentioned earlier, our aim is to predict the next day's LV load demand which is not available until the refinement of the network's measurements that are made available at month n concerning month's n-1 electricity load demand. Based on that, we propose the following mechanism in order to extract from the daily OoEM's forecast the proportion that represents the load demand for Low Voltage. The proposed mechanism encounters the LV load demand forecasting as a regression problem and creates a LightGBM model, which includes variables that affect load demand, the OoEM's daily total load forecast and aims to predict with the highest accuracy the percentage of LV demand inside the OoEM's total load forecast. Thus we created a new dataset containing the following columns as seen in the picture below:

- Weekday: The day of Week (starting from Monday) that the date is referring to.

- **Holiday:** Indicator of whether this day is a Holiday or not.
- **ForecastTotalLoad:** The daily forecast of OoEM concerning the total network's load demand. It's the only near-live information that we have and OoEM publishes this every day.
- **Month:** The Month that the load forecast value is referring to.
- **Hour:** The Hour that the load forecast value is referring to.
- **%LVofTotal:** This is our response variable, and denotes the percentage of OoEM's total load forecast that concerns Low Voltage.

Timestamp	Weekday	Holiday	ForecastTotalLoad	Month	Year	Hour	%LVofTotal
2018-01-01 00:00:00	1	1	5315.0	1	2018	0	0.739157
2018-01-01 01:00:00	1	1	4963.0	1	2018	1	0.722294
2018-01-01 02:00:00	1	1	4838.0	1	2018	2	0.719718
2018-01-01 03:00:00	1	1	4515.0	1	2018	3	0.702371
2018-01-01 04:00:00	1	1	4280.0	1	2018	4	0.671507
...
2019-10-31 19:00:00	4	0	6011.0	10	2019	19	0.659009
2019-10-31 20:00:00	4	0	5913.0	10	2019	20	0.659891
2019-10-31 21:00:00	4	0	5523.0	10	2019	21	0.651541
2019-10-31 22:00:00	4	0	5005.0	10	2019	22	0.626431
2019-10-31 23:00:00	4	0	4723.0	10	2019	23	0.594625

Load demand declaration for Low Voltage is a daily procedure for energy analysts. As we mentioned earlier, the actual values of Load demand are made available, one month after the time of occurrence, or in other words after the refinement of measurements has been made. Although energy analysts need information right now in order to be able to provide a decent forecast for the next day's Low Voltage load demand. The proposed mechanism for fetching Low Voltage load demand data, before its time is based on historical data and we calculated the %LVofTotal which is our response variable. Every day based on OoEM's fairly accurate forecast we will estimate the percentage of LV in contrary with the daily OoEM's total load forecast. As we mentioned earlier, to do that we used the LightGBM algorithm and we calculated the LV percentage of OoEM's total load forecast, for the 24 hours we want to predict. In other words, we obtained 24 percentages and then we multiplied these percentages to the OoEM's hourly total network's load forecast, thus getting 24 predictions as if OoEM had predicted the LV load demand specifically. In the following picture, we provide a graph that shows the actual LV load demand for the day 15/9/2019 - which was made available to us at around middle of October- versus the results obtained from the procedure described above, e.g the adjustment of daily OoEM's total load forecast to Low Voltage. In other words, the blue line(ActualLVLoad) shows the actual LV load demand value for that day (which is an unknown and we were able to calculate it only after the data made available with approximately one month lag) versus the OoEM's Low Voltage Adjusted forecast, which calculated as the proportion of LV that we forecasted multiplied with the OoEM's hourly forecast appropriately. In the figure below We can clearly see that we are fairly close to the actual LV load demand and we quantified the deviation at 131.3 KWh, using the RMSE error.

So to sum up the aim of this thesis is to create a model for daily Low Voltage load demand, which is not available by OoEM but the analysts have to declare it nonetheless. The above described algorithm will be used in order to adjust OoEM's daily forecast for the whole network to the LV load forecast, as if OoEM provided on daily basis load demand forecast, and incorporate this information in the daily task of Low Voltage demand forecasting in our model's later.

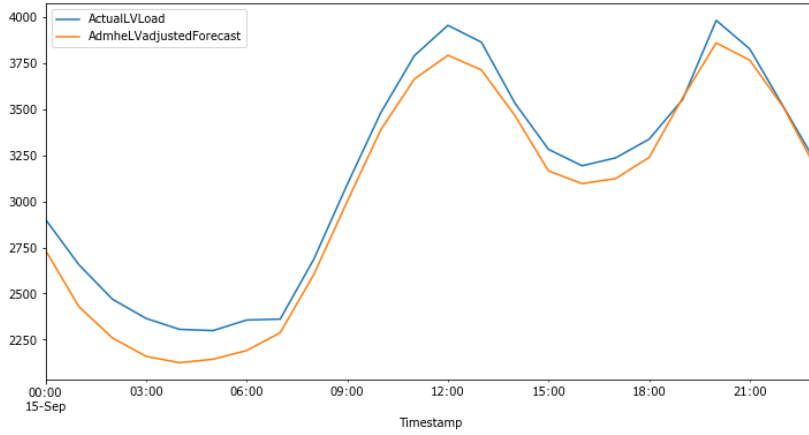


Figure 11.1: Actual LV VS OoEM's LV Adjusted Forecast

11.4.1 Outliers-Missing Values

Having in mind that the datasets are certified load demand data and moreover that the IPTO's dataset presents the actual load demand, we don't expect to have outliers, missing values or bad data etc. The only case where we have to deal with "missing values" is on the Daylight saving time (DST) days. Daylight Saving Time is the procedure of setting the clocks forward one hour from standard time during the summer months, and back again in the fall, in order to make better use of natural daylight. This phenomenon is depicted in hour dataset as follows: Once a year there is a day with "23 hours" and also 23 hourly load demand values when local time is about to reach e.g Sunday, 31 March 2019, 03:00:00 clocks are turned forward 1 hour to Sunday, 31 March 2019, 04:00:00 local daylight time instead. On the other hand, there is one day with 25 hours and 25 load demand values when the local daylight time was about to reach e.g Sunday, 27 October 2019, 04:00:00 clocks are turned backward 1 hour to Sunday, 27 October 2019, 03:00:00 local standard time instead. To deal with this situation, after communicating with an energy analyst, working for one of the biggest energy providers in Greece, we will fill the missing values with the previous' hour load demand and in the case of the extra hour, we will calculate the average load demand value for these two hours that the hour change occurs e.g in hour example, on 31 March we will take the average load demand of hours 3 and 4.

11.4.2 Summary Statistics & Data Visualizations

Any data science problem, starts with the investigation of the available data to examine their properties and identify underlying correlations, so some exploratory data analysis was done on our dataset to this end. As we mentioned earlier, daily electricity load demand is greatly affected by seasonality, Holiday etc. In this task, our response variable is the LV e.g the load demand for load meters in hourly basis. In the following graphs, we will plot our response variable LV in order to identify interaction with time itself, as we are dealing here with time series, but also we will examine our variable of interest for any correlation with the rest of the explanatory variables. In the following graph, we can observe that Low Voltage load demand is a bit lower in Holiday, in contrary with normal days. This is fairly logical, because under the category of Low Voltage meters exist home meters, SME meters, office meters etc, so it is expected on Holiday when offices and SMEs are closed to expect lower load demand.

In the next graph, we plot the Low Voltage load demand based on the Weekday, e.g Monday, Tuesday...etc, in order to identify if and how the Weekday under examination affects the load demand. By observing the plot, we can infer that electric load demand is significantly lower during the Weekend because of reduced economic activity, as SMEs and offices are closed. Lower levels of demand are also observed on Mondays compared to the other "normal" - e.g non Holiday and non Weekend - days of the week, due to inertia caused by the reduced economic activity of Weekend. Here we create a boxplot for our response variable and we can observe that Low Voltage load demand most of the times, or in other words in the 2nd and 3rd quantile is approximately between 2800 and 4500 KWh, but as we can see there are values below 2000 KWh and also outliers over

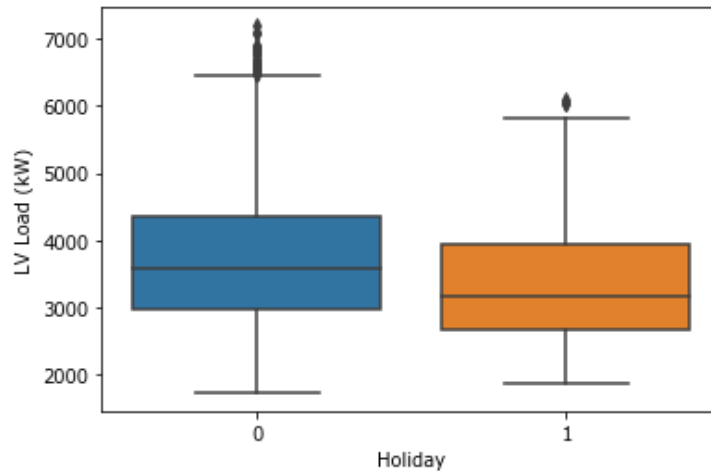


Figure 11.2: Low Voltage Load Demand vs Holiday

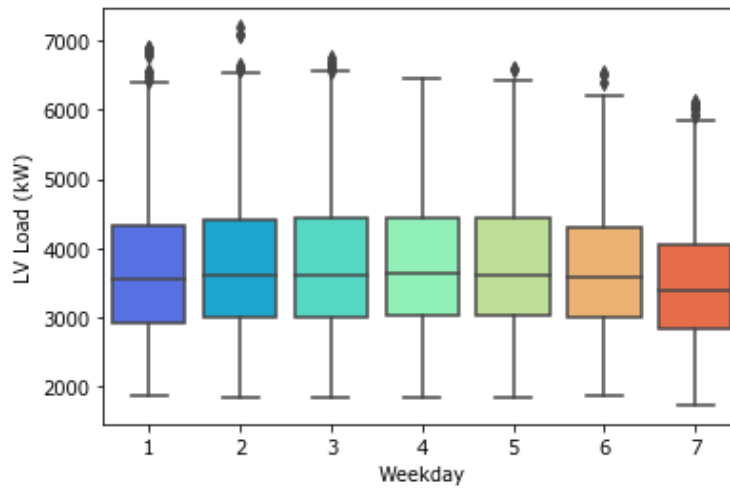


Figure 11.3: Low Voltage Load Demand vs Weekday

6500 KWh per hour. Electricity load demand presents fluctuations during the course of the day,

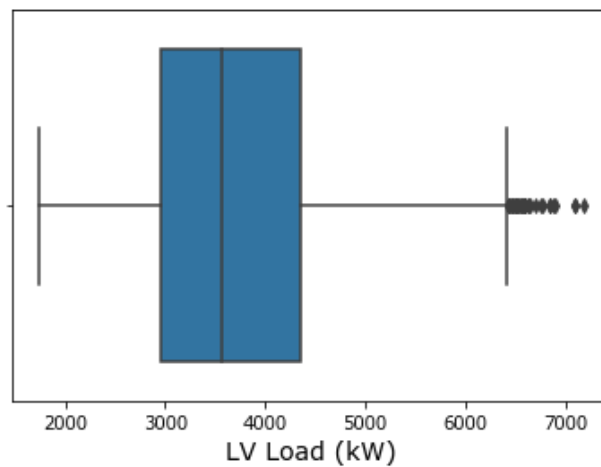


Figure 11.4: Low Voltage Load Demand Boxplot

based on daily human activity. Given that our dataset contains hourly electric load demand values,

which show diurnal variation in electric load demand. In the next graph, we plotted the low voltage load demand on a random day and as we can see, the hourly variation during the 24 hour period is evident. More specifically, we can observe that the first load demand maximum value appears close to midday due to extensive use of electricity both for household activities (e.g cooking, heating etc) and businesses activities (office lighting, PC usage etc) during the working hours. The second load demand maximum is observed during the late afternoon and early evening hours and this is due to the extensive use of lighting and heating or cooling.

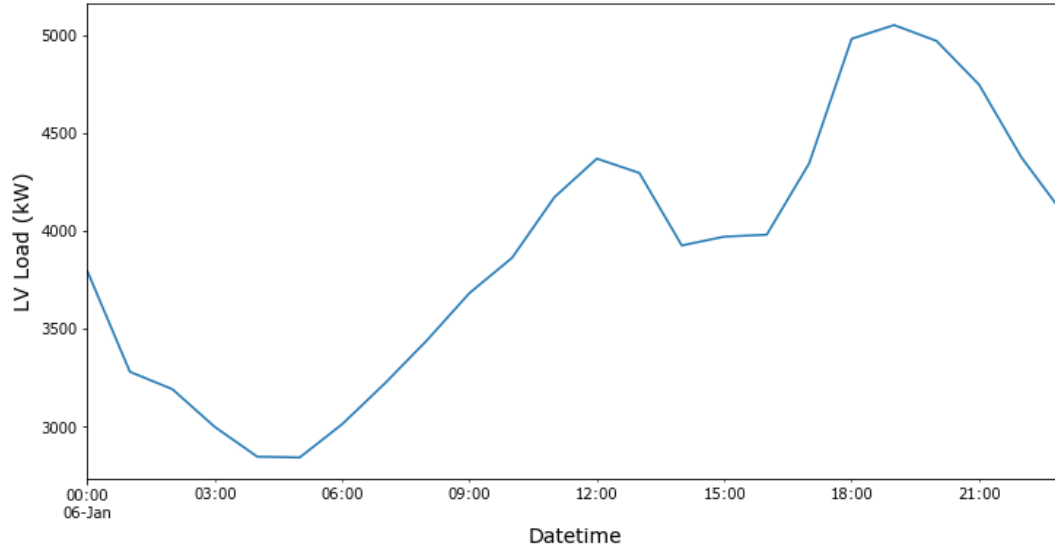


Figure 11.5: Daily Low Voltage Load Demand per Hour

In literature, there is a lot of discussion concerning electricity load demand and seasonality, thus we plotted the Low voltage load demand over one year time. More specifically, we used the actual LV load demand data of 2018 and plotted them over time. By looking at our graph, we can easily identify the strong seasonality appeared in the electric load demand behaviour, where there is observed higher load demand in winter in contrary with the summer. More specifically, we can observe higher electricity load demand in January, which thereafter gradually decreases until May. The above mentioned decrease in demand, moving towards the Spring as the weather gets warmer, coincides with a decrease in demand for heating. As we move into the Summer, we observe a gradual increase in electricity demand, due to extensive use of air conditioners as the temperature gets higher and higher. This trend continues throughout the period of Summer till September, with the exception of August when a significant decrease in load demand is obvious. This decrease in August can be explained by the fact that the majority of people is on summer vacation during this month. A slight decrease in electricity load demand is also observed in October, which is representative of the transition from summer to winter, just before starting to gradually increase again in the following months to reach a maximum in December. The electricity load demand in December is higher than in January and February (which are Winter's coldest months), due to increased load demand during the festive Christmas period.

So from our exploratory analysis we can conclude the following:

- Electricity load demand curve is seasonal and presents fluctuations across the week and throughout the day.
- Irregular events (e.g extreme weather) can heavily affect electric load demand.
- Seasonal trends and fluctuations observed in load curve are due to weather conditions or factors such as weekend and holiday effects.
- Load demand minimum values are observed in the transient seasons (Spring & Autumn), whereas load demand maximum values are observed in Winter and Summer months.
- Lower electric load demand values are observed during holiday and weekends, in contrary with the normal weekdays (working days).

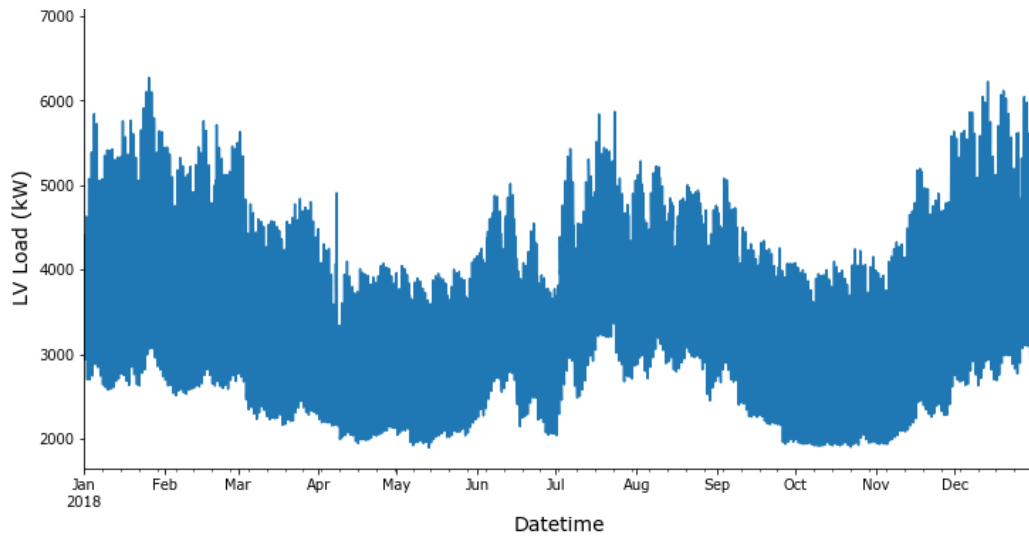


Figure 11.6: 2018 Low Voltage Load Demand (Historical Data)

- Last but not least, concerning daily variation, we observed that there is higher load demand during midday and late afternoon, regardless of season.

11.5 SARIMAX Model

Fistly, we will apply the SARIMAX model and try to forecast Low Voltage load demand one day ahead, or in other words for the next 24 hours. As observed in the exploratory data analysis, there is trend and seasonality in our response variable and also our response variable is heavily affected by exogenous variables such Weekday, Holiday etc, so concerning ARIMA models, we will use the Seasonal AutoRegressive Integrated Moving Average with eXogenous factor (SARIMAX (p, d, q)(P, D, Q, m)). To obtain the optimal order values, we used Python's *auto arima()* function with m=24, where m is the period for seasonal differencing and refers to the number of periods in each season, thus because we have hourly frequency in our time series we set it to 24. The auto arima function returned us the following optimal values for the SARIMAX orders as seen in the following figure. After obtaining the optimal values (**SARIMAX(1,1,1)x(2,0,0,24)**) from the

Statespace Model Results

Dep. Variable:	y	No. Observations:	16056			
Model:	SARIMAX(1, 1, 1)x(2, 0, 0, 24)	Log Likelihood	-88508.909			
Date:	Thu, 16 Jan 2020	AIC	177033.818			
Time:	13:00:46	BIC	177095.288			
Sample:	0	HQIC	177054.145			
	- 16056					
Covariance Type:	opg					
	coef	std err	z	P> z	[0.025	0.975]
intercept	-0.2356	0.643	-0.366	0.714	-1.497	1.026
x1	2.5371	0.558	4.547	0.000	1.443	3.631
x2	15.6888	3.236	4.848	0.000	9.346	22.031
ar.L1	0.3601	0.006	60.268	0.000	0.348	0.372
ma.L1	0.4019	0.007	61.620	0.000	0.389	0.415
ar.S.L24	0.6684	0.005	147.943	0.000	0.660	0.677
ar.S.L48	0.2891	0.004	64.722	0.000	0.280	0.298
sigma2	3599.6471	16.098	223.604	0.000	3568.095	3631.199
Ljung-Box (Q):	1893.18	Jarque-Bera (JB):	135539.17			
Prob(Q):	0.00	Prob(JB):	0.00			
Heteroskedasticity (H):	0.74	Skew:	0.57			
Prob(H) (two-sided):	0.00	Kurtosis:	17.19			

Figure 11.7: SARIMAX optimal model summary

function we split the dataset into train and test set, where the test set is the dataset's last day or in other words the last 24 hours that we want to predict. Then we fit a SARIMAX model with the appropriate order values and the Holiday and Weekend variables as exogenous factors. After fitting the model we will make the predictions and in order to evaluate the model we will use the Root Mean Square Error metric. As said earlier, the object of this thesis is to create a model that we be more accurate than the OoEM's forecast model, thus every forecast will be compared with that forecast which is the value that we created earlier LVForecastEstimation. Again we should note that the LVForecastEstimation is the best case scenario, because OoEM does not know a-priori the proportion of Low Voltage in his daily total load forecasts. We were able to calculate this percentage based on historical data with one month lag as mentioned earlier. In the following figure, we plotted our result to get a graphical representation of our model's accuracy where the blue line **LV**, is the dataset's actual LV load demand as calculated earlier, the green line **LVForecastEstimation** is the OoEM's forecast LV load demand as calculated earlier and the orange line **OurPrediction** is the SARIMAX model's predicted values.

By looking at the graph, we can conclude that our prediction is very decent, because as we can see the model follows very accurately the daily seasonality, although the OoEM's forecast is slightly better. In order to quantify the error, we will use the RMSE error as we said earlier and calculate our's prediction RMSE and the OoEM's prediction RMSE, as seen on the table below.

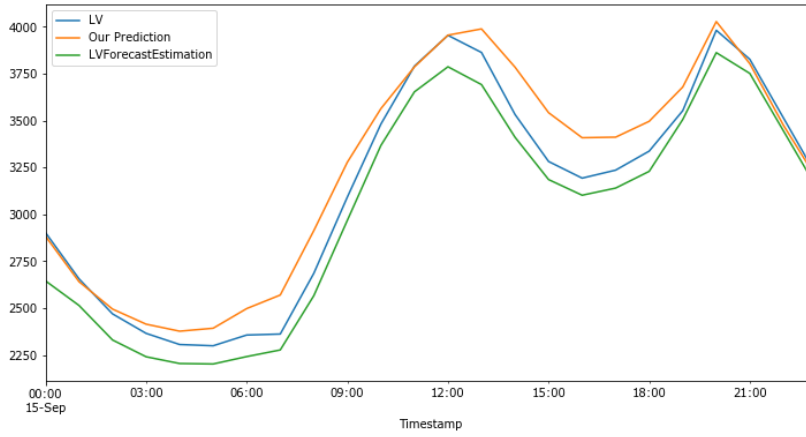


Figure 11.8: SARIMAX LV load forecasts vs Actual load vs OoEM's load forecast

Their RMSE Error	Our RMSE Error
123.23	136.03

11.6 Prophet Model

In this section, we will use Facebook's Prophet algorithm to make a forecast on the same day as before in order to be able to make the comparison between models easier. This time we will import and then use the Python's Prophet library, to create our Prophet model. This algorithm, requires some data preprocessing in order to work, as we mentioned earlier. Because we have holidays and weekday in our dataset that we would like to model as it affects load demand in a great way, we must create a dataframe for them, which should include the holiday occurrences and the weekday sequence, both in the past (historical data) and in the future (forecast horizon). Then again, we fit the Prophet model and then we will make a forecast again for the last 24 hours as before, and again in order to evaluate the model we will use the Root Mean Square Error metric. Furthermore, in order to produce even better forecasts, we tuned some of the model's parameters as seen below.

- *growth*: We set this parameter as "linear" because by observing our time-series plot we identified that there is a trend that keeps on growing with no real saturation insight. In any other case, e.g when the response variable must saturates, we would set this parameter to "logistic".
- *holidays_prior_scale*: This parameter concerns the holiday effect on our time-series data. In other words, determines how much of an effect holidays should have on our predictions. When we know that holiday have a massive impact on our data, we should try large values. In our case after several tests, we set the *holidays_prior_scale* equal to 30.
- *changepoint_prior_scale*: This parameter indicates the changepoints flexibility value. Here large values denotes more flexibility in the model, although you may end up overfitting. Several tests made and we found the optimal *changepoint_prior_scale* value to be 50.
- *seasonality_mode*: This parameter illustrates the way that seasonality terms should be incorporated into the predictions. The available options here are additive or multiplicative. In general, when the forecasted measure is affected by seasonalities we should set this parameter to multiplicative. As we saw in our EDA, load demand is greatly affected by seasonality thus we will use multiplicative in order to strengthen the seasonality effect in our model's forecasting procedure.
- *seasonality_prior_scale*: This parameter indicates the seasonality flexibility value. Again large values allow more seasonality flexibility in the model and after several tests we set the *seasonality_prior_scale* value equal to 50.

- *yearly_seasonality*: Indicates whether yearly seasonality is present on the data. We set this value to true, as observed in the EDA section.
- *weekly_seasonality*: Indicates whether weekly seasonality is present on the data. We set this value to true, as observed in the EDA section.
- *daily_seasonality*: Indicates whether daily seasonality is present on the data. We set this value to true, as observed in the EDA section.

Last but not least, to improve even more the model's forecasting ability, we added two additional regressors concerning the type of day e.g whether the day under study is Holiday or not and the day of week respectively. As we saw in the EDA part, these two independent variables affect in a massive way load demand, so we decided to incorporate them in our model as well. Both of these independent variables values must be known a-priori both in the past and in the future. According to literature, adding more regressors is crucial concerning accurate forecasting in Prophet algorithm, as it's a way to tune the model and make the forecasting procedure more clear and user-friendly.

To visualize individual forecast components, we used Prophet's built-in `plot_components` function: As we observe in the following component visualizations, Prophet was able to accurately model the underlying trend in the data, while also accurately modeling weekly, yearly and daily seasonality. The components plot consists of e sections: the trend, the extra regressors, and the seasonality. The sum of those components account for the entirety of the model in fact. The trend is simply what the data is showing if you subtract out all of the other components. The `extra_regressors` plot shows the effect of all of the regressors (Weekday & Holiday) included in the model. Additional regressors are like holidays in that they cause the trend to deviate from the baseline, except that the trend will stay changed after the event. The weekly seasonality component shows that load demand is constantly increasing throughout he week, but with a steep decline on the Monday. Moreover, the yearly and daily seasonality is quite wavy as can be observed in the plots. These plots are created with Fourier transforms, essentially stacked sine waves.

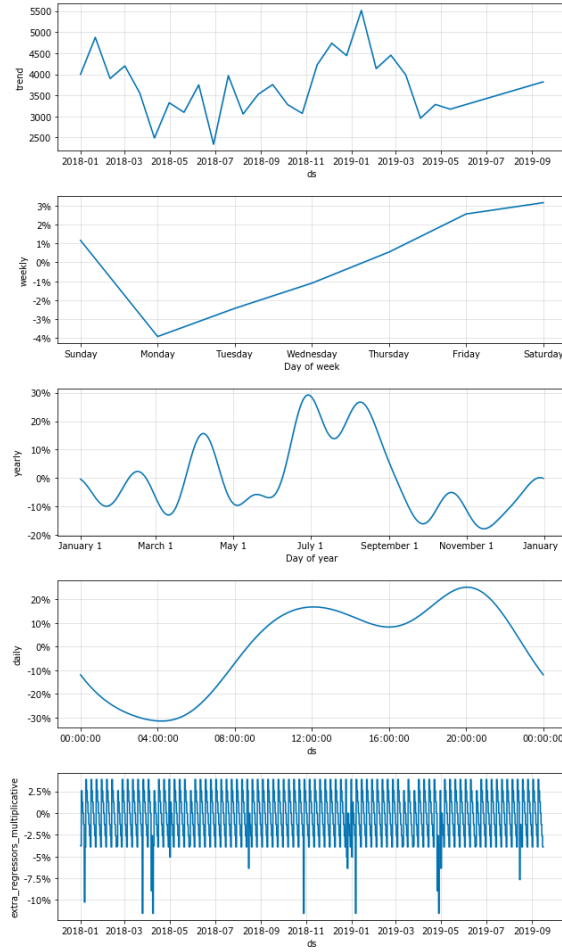


Figure 11.9: PROPHET's components

Last but not least, after generating the forecasts using the Prophet model for the same day as before in order to be able to compare, we plotted our predictions. In the following figure, we plotted our result to get a graphical representation of Prophet model's accuracy where the blue line **LV**, is the dataset's actual LV load demand, the green line **LVForecastEstimation** is the OoEM's forecast LV load demand as calculated earlier and the orange line **OurPrediction** is the Prophet model's predicted values. By looking at the graph, we can clearly see that our prediction

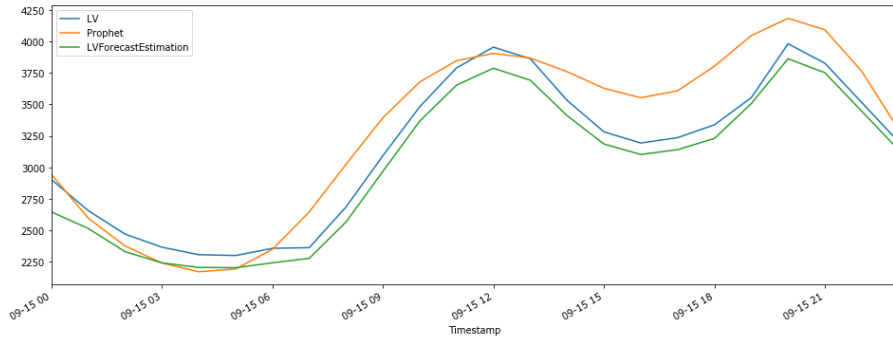


Figure 11.10: PROPHET LV load forecasts vs Actual load vs OoEM's load forecast

is not as good as before, and although the model seems to follow the daily seasonality the model is not accurate, and of course the OoEM's forecast is much better. Again to quantify the error, we will use the RMSE error as we said earlier and calculate our prediction's RMSE as seen on the table below.

Their RMSE Error	Our RMSE Error
123.23	235.20

11.7 Multivariate Multi-Step LSTM Model

In this section, we are going to investigate Neural Networks and we will create a special type of NN which is the LSTM neural network. As we saw earlier, we created an hourly observation of LV load demand based on the total network's actual values. Thus for each day, we will have 24 observations one for every hour of the day. Generally, in Deep Learning it is crucial to scale features before training a neural network. More specifically, before creating the model we have to do some data preprocessing, which involves framing the dataset as a supervised learning problem and normalizing the neural network's input variables. Concerning the continuous variables we will use Python's function `MinMaxScaler()` to do the scaling, whereas for the categorical variables we will use One-Hot-Encoding. After doing the scaling of the data, we will split it in train/test dataset and we will use the 33% of the dataset as test dataset.

In a multi-step prediction model, based on the past history values (whose length in the past is defined by the user), the model is asked to predict a range of future values. In other words, the multi-step neural network model needs to learn to predict a sequence of values in the future, in contrary with a single-step neural network model, where only a single future point is predicted. The above mentioned framing of the problem is also known as a multi-step time series forecasting problem, given the multiple forecast steps and furthermore a model that uses multiple input variables is scientifically called a multivariate multi-step time series forecasting model, which this is our case-study in this experiment. For the multi-step model, after doing some experiments we chose the training data to consist of hourly load demand observations over the past fifteen days. Thus, we have to create a time window containing the most recent 288 (15x24) observations to train our neural network model. These time intervals from now on will be called windows, and there are the windows of time containing the appropriate data for the model to train on. The window *past history* denotes the size of the past window of data observations, whereas the *target size* denotes how far in the future the model need to learn to predict. Here according to the business rules, the model needs to learn to predict the load demand value for the next day. Since an observation is recorded every hour, the output consists of 24 predictions e.g one for every hour of the day.

As mentioned earlier, the dataset needs to be prepared accordingly before training, thus the first step is just to re-create it, but this time with the appropriate pre-specified *past history* and *target size*. Concerning the LSTM neural network architecture, we used Keras library and after doing some experiments, we concluded in creating a sequential model with 4 Layers. The first and the second layer have 100 LSTM neurons each, whereas the third one has 50 neurons. The last layer, is a Dense layer and since 24 predictions are made (forecasting 24 hours in the future), the dense layer has 24 neurons. The model was trained for 20 epochs with a batch size of 256 and concerning

```
Model: "sequential"
```

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 360, 100)	73600
lstm_1 (LSTM)	(None, 360, 100)	80400
lstm_2 (LSTM)	(None, 50)	30200
dense (Dense)	(None, 24)	1224
Total params: 185,424		
Trainable params: 185,424		
Non-trainable params: 0		

Figure 11.11: Multi-Variate & Multi-Step LSTM Model Summary

the loss function we used the Root Mean Square Error (RMSE) and the Adam optimiser also. The Adam optimization algorithm is described in literature as an extension of the stochastic gradient

descent algorithm and recently has seen broader adoption in the scientific community concerning neural network applications. Adam optimization algorithm or Adam optimizer is used in neural networks to update network weights in an iterative way based on the training data. In the next plot, we present our neural networks training and validation loss curves, that were extracted after finishing with the training of our model.

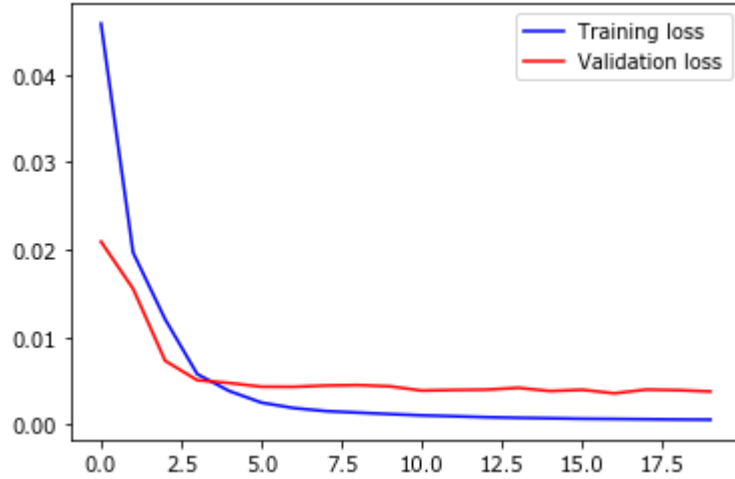


Figure 11.12: LSTM Model's Train & Validation Loss

To evaluate how well our neural network learnt to predict the future, a **rolling-forecast** scenario will be used, which is also technically described as walk-forward model validation. In simple words, each time step of the test dataset will be walked one observation at a time. More specifically, the proposed neural network model will make a forecast for a specific time step and then, after the forecast, the actual expected load demand value for the next time step of the test dataset will become available to the neural network in order to be incorporated in the forecast for the load demand value of the next time step, e.g the load demand value for the next hour. In other words, when the neural network model is required to make a 24 hour forecast in the future, then after the forecast, the actual load demand data for that 24 hour period is made available to the neural network model, in order to be used for making a forecast on the subsequent 24 hour period. The above described procedure is realistic concerning the way that the model will be used in practice and also beneficial, by allowing the neural network model to use the most recent available data. After making the forecasts and obtaining the predicted values, we have to invert the data transformations made earlier, in order to return the predicted load demand values back into their original scale. We can do this directly, using Python's *MinMaxScaler* object that offers an inverse transformation function (*invert_transformation()*). All collected forecasts made on the same test dataset and then we implemented an error score function - which used once again the RMSE metric - in order to summarize the model's forecasting skill (for each time step) and output an average of all the prediction errors.

Furthermore, in the following figure, we plotted our result to get a graphical representation of our model's accuracy where the blue line is the **past history**, e.g the dataset's actual LV load demand for the declared history size window -in our case 15 days of previous observations-, the green line is the **true future**, the current actual LV load demand for the specified target size e.g how far in the future does the model need to learn to predict. Last but not least, the red line **Predicted Future** is the predicted values forecasted by the LSTM Neural Network for the specified target window size e.g 24 hours in the future based on the specified past history window size e.g using 15 days history data to make the prediction. By looking at the graph, we can clearly see that our prediction is again worse than before, and although the model seems to recognise the hourly seasonality -as it is denoted by the spikes and the ups and downs in the graph- but the model is not accurate enough. To quantify the error, we will use the RMSE error as we said earlier and calculate our prediction's RMSE as seen on the table below. As we can see once again, the OoEM's forecast is still better.

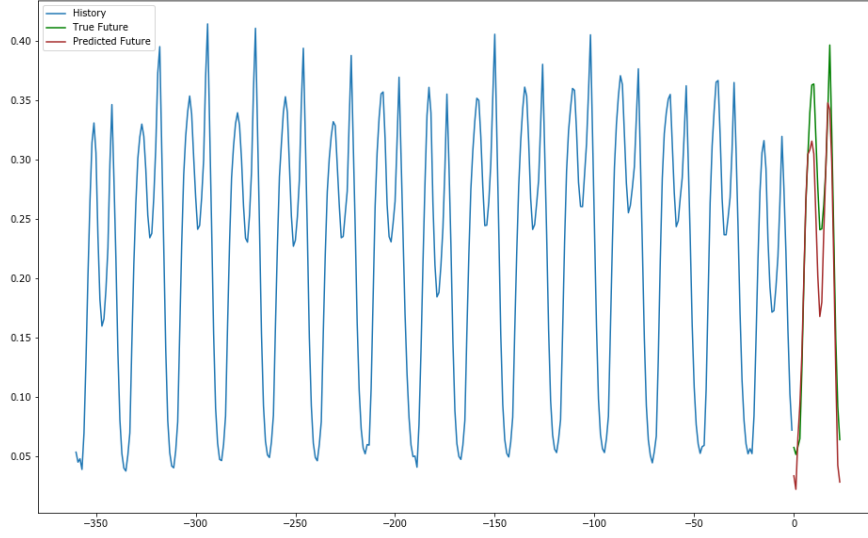


Figure 11.13: LSTM model's load forecasts vs Actual Load

Their RMSE Error	Our RMSE Error
123.23	194.10

11.8 Time Series as a Regression Problem

As we mentioned earlier, in literature there are numerous publications concerning Electricity Load Demand Forecasting, which state that this forecasting problem should be approached as a regression problem and as confirmed in the EDA section, there are numerous exogenous factors that affect load demand and also it's affected greatly by seasonality. Thus in this section, we will try to approach this problem as a regression problem, we will apply some of the state-of-the-art algorithms and evaluate their performance. To do that first of all, we have to make some transformation in our dataset. First of all we will create four more columns, which will denote the Day, Hour, Month and Year of the observation respectively. Moreover as we saw, earlier the LV load demand presents apart from great seasonality, indicates autocorrelation also. **autocorrelation** metric is used to measure the linear relationship between lagged values of a time series, just as correlation metric used to estimate the extent of linear relationship between two variables, by identifying and measuring their interaction. In other words, autocorrelation phenomenon occurs when time-series is linearly related to a lagged version of itself.

The estimation of this linear relationship between lagged values is realized by the **Autocorrelation coefficients**, as shown in the following formula. More specifically, r_1 measures the linear relationship between lagged values y_t and y_{t-1} , r_2 measures the linear relationship between lagged values y_t and y_{t-2} and so on and so forth. The value of r_k can be formulated as where T is the

$$r_k = \frac{\sum_{t=k+1}^T (y_t - \bar{y})(y_{t-k} - \bar{y})}{\sum_{t=1}^T (y_t - \bar{y})^2},$$

time series length. When time-series have trend, the autocorrelation value for small lag intervals

tend to be large and positive, because nearby observations in terms of lag intervals, tend to be similar in size and small discrepancies are observed. Autocorrelation plots of time-series which have trend, present large positive autocorrelation values that slowly decrease as the lags time intervals gets bigger. On the other hand, when seasonality is observed in the time-series, autocorrelations will be larger for the seasonal lag intervals (multiples of the seasonal frequency). In cases such electric load curve when data present both trend and seasonality, a combination of these effects is observed as shown in the following plot. In the next graph, we present the autocorrelation plot of our response variable, e.g the LV load demand.

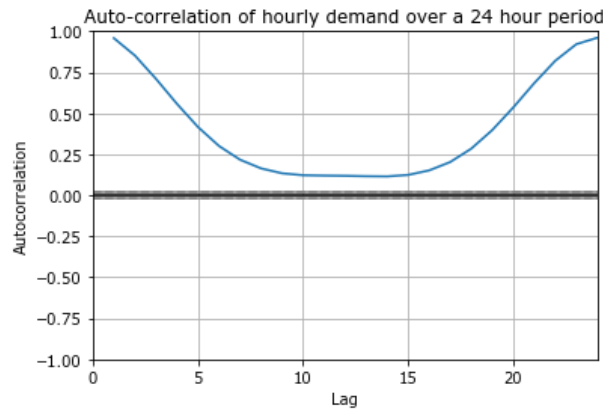


Figure 11.14: Auto-correlation plot over 24 hour period

The autocorrelation plot below shows the extent to which the demand variable correlates with itself at different intervals (lags). This plot shows that demand is highly autocorrelated over the closest 6 hour period. To incorporate this information in our dataset, we created a function to generate for each observation in the dataset, the LV load demand value for closest 6 lags. Thus now our dataset contains the following columns,

- Year: The year that the load value is referring to.
- Month: The month that the load value is referring to.
- Hour: The hour (t) that the load value is referring to.
- Weekday: The day of Week (starting from Monday) that the date is referring to.
- Holiday: Indicator of whether this day is a Holiday or not.
- LV: The load demand for Low Voltage for that hour of the specific day.
- LV_lag1: The load demand for Low Voltage on hour (t-1).
- LV_lag2: The load demand for Low Voltage on hour (t-2).
- LV_lag3: The load demand for Low Voltage on hour (t-3).
- LV_lag4: The load demand for Low Voltage on hour (t-4).
- LV_lag5: The load demand for Low Voltage on hour (t-5).
- LV_lag6: The load demand for Low Voltage on hour (t-6).

as shown in the picture below.

Then again, we are going to split the dataset in train/test datasets and apply one of the most state-of-the art algorithms for regression, LightGBM, to build our model and fit it. LightGBM Regressor is a Gradient Boosting Machine that has many parameters that can be tuned, thus we will use a technique called Grid Search in order to find the optimal parameters for our proposed model. Grid-searching is the process of scanning the data to configure optimal parameters for a given model, because certain parameters are necessary to be tuned depending on the algorithm

Timestamp	Weekday	Holiday	Year	Month	Hour	LV	LV_lag1	LV_lag2	LV_lag3	LV_lag4	LV_lag5	LV_lag6
2019-09-30 14:00:00	1	0	2019	9	14	3700.472680	3838.257197	3804.698167	3669.041729	3504.153083	3267.180635	2917.487389
2019-09-30 15:00:00	1	0	2019	9	15	3470.274327	3700.472680	3838.257197	3804.698167	3669.041729	3504.153083	3267.180635
2019-09-30 16:00:00	1	0	2019	9	16	3327.570305	3470.274327	3700.472680	3838.257197	3804.698167	3669.041729	3504.153083
2019-09-30 17:00:00	1	0	2019	9	17	3326.882177	3327.570305	3470.274327	3700.472680	3838.257197	3804.698167	3669.041729
2019-09-30 18:00:00	1	0	2019	9	18	3427.843535	3326.882177	3327.570305	3470.274327	3700.472680	3838.257197	3804.698167
2019-09-30 19:00:00	1	0	2019	9	19	3826.308438	3427.843535	3326.882177	3327.570305	3470.274327	3700.472680	3838.257197
2019-09-30 20:00:00	1	0	2019	9	20	4075.101694	3826.308438	3427.843535	3326.882177	3327.570305	3470.274327	3700.472680
2019-09-30 21:00:00	1	0	2019	9	21	3763.093504	4075.101694	3826.308438	3427.843535	3326.882177	3327.570305	3470.274327
2019-09-30 22:00:00	1	0	2019	9	22	3311.041237	3763.093504	4075.101694	3826.308438	3427.843535	3326.882177	3327.570305
2019-09-30 23:00:00	1	0	2019	9	23	2927.471492	3311.041237	3763.093504	4075.101694	3826.308438	3427.843535	3326.882177

that the model utilizes. Grid-searching can be applied across machine learning to calculate the best parameters to use for any given model. During the process of Grid-Search, a model on each parameter combination possible will be built. The algorithm iterates through every possible parameter combination - among the values specified for each parameter- and stores the model for each combination.

So after running the GridSearch algorithm we reached a LightGBM Regressor model with the following parameters.

- **boosting_type:** gbdt
- **objective:** regression
- **num_leaves:** 1200
- **learning_rate:** 0.17
- **n_estimators:** 700
- **max_depth:** 6
- **metric:** rmse
- **bagging_fraction:** 0.8
- **feature_fraction:** 0.8
- **reg_lambda:** 0.9

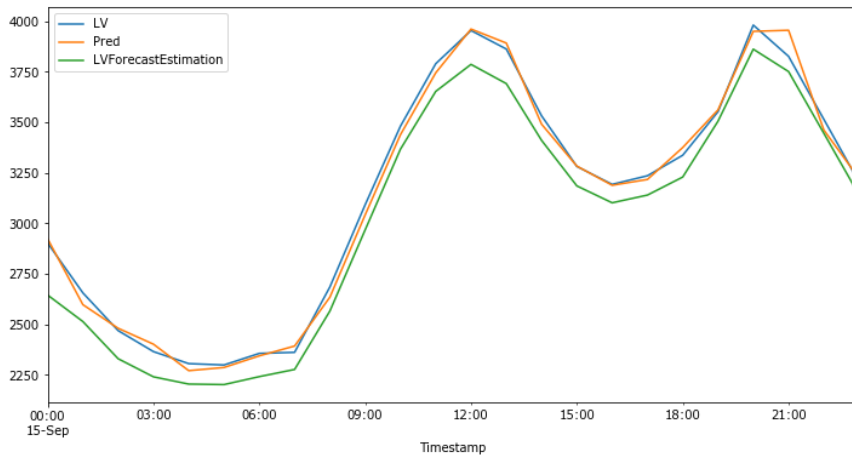


Figure 11.15: LightGBM LV load forecasts vs Actual load vs OoEM's load forecast

As we can see in the provided graph our model generated great forecasts and even surpassed OoEM's forecast. Moreover, we used the RMSE error to quantify our results and we obtained a

Their RMSE Error	Our RMSE Error
<i>123.23</i>	<i>41.86</i>

forecast with RMSE equal to 41,86 in contrary with OoEM's forecast which has an error of 123.23. Last but not least, to be even more confident about our model's general forecasting ability and also ensure that the above results were not random, we ran one more experiment as we tried to predict a whole month and more specifically the Sempتمبر of 2019. Again we perform the same procedure, as we split the dataset in train and test sets and used the LightGBM model constructed above. Below we can see our results concerning the forecasting of one whole month. In the first figure, with the blue line are represented the actual Low Voltage load demand for September, where on the other hand with the orange line we present the evaluation criterion, or in other words OoEM's adjusted Low Voltage forecast. In the second graph, with the blue line again we present

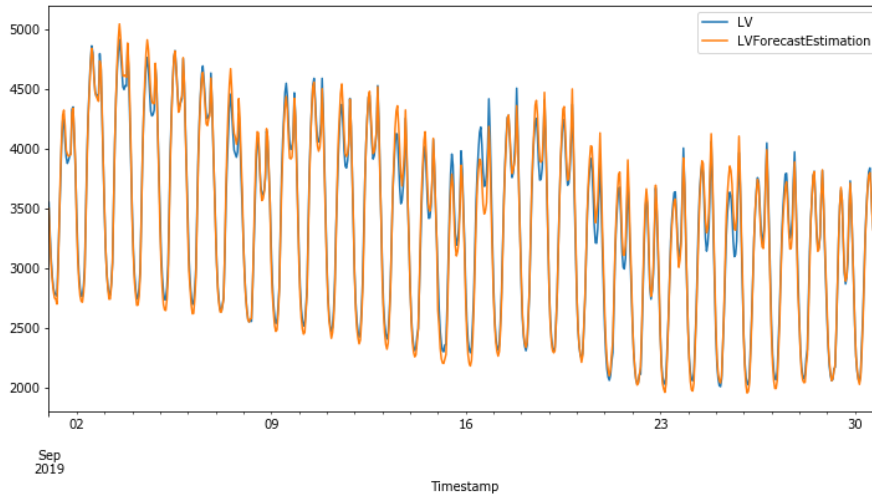


Figure 11.16: Actual LV load demand vs OoEM's LV load forecast

the actual Low Voltage load demand for September, where on the other hand with the orange line we present our model's prediction for the Low Voltage load demand during September. By just looking at the two graphs, it's easy too understand that our model again performed better again and we quantified the error's again using RMSE which is our evaluation metric. The results are presented in the following table, where OoEM's forecast deviation where measured with an RMSE error of 88.14, where on the other hand our model generated an RMSE error of 55.73, thus outperforming OoEM's forecasts. Last but not least, to enhance our conclusion visually with

Their RMSE Error	Our RMSE Error
<i>88.14</i>	<i>55.73</i>

greater detail, we provided graphs that visualize the forecast of three random days in September as forecasted by our models. In the following graphs, we present with the blue line the actual Low Voltage load demand, with the green line the OoEM's Low Voltage adjusted forecast and with the orange line, we present our model's forecasts.

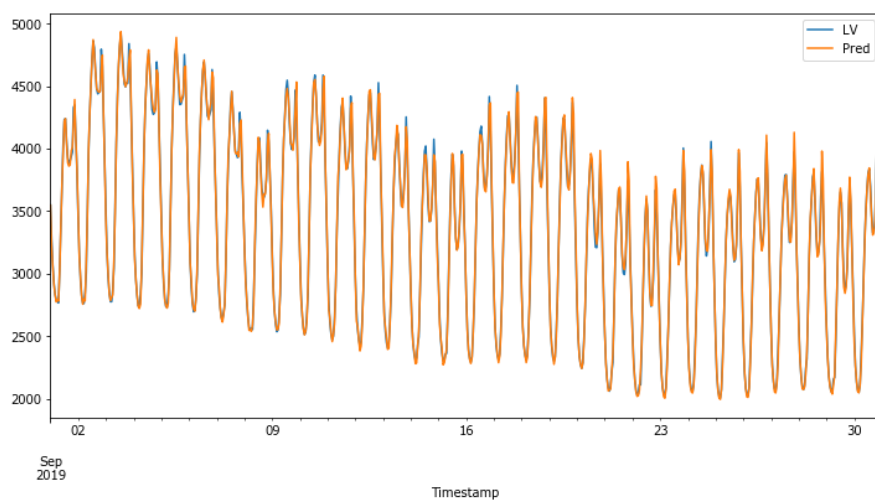


Figure 11.17: Actual LV load demand vs LightGBM model's LV load forecast

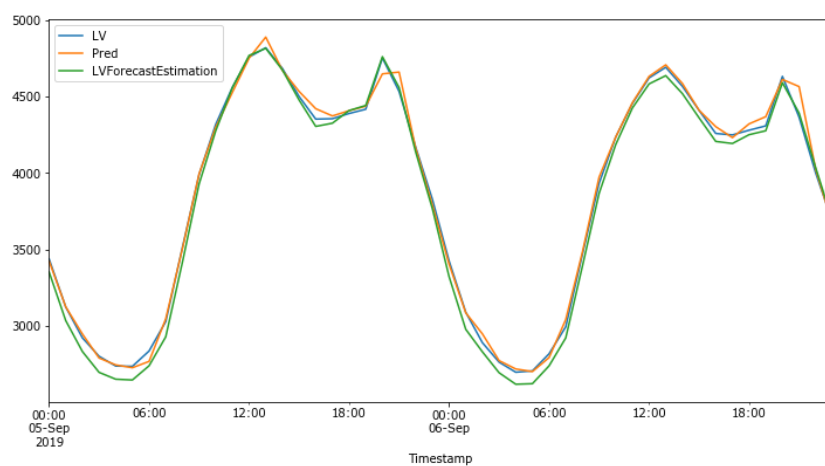


Figure 11.18: LightGBM LV load forecasts vs Actual load vs OoEM's load forecast

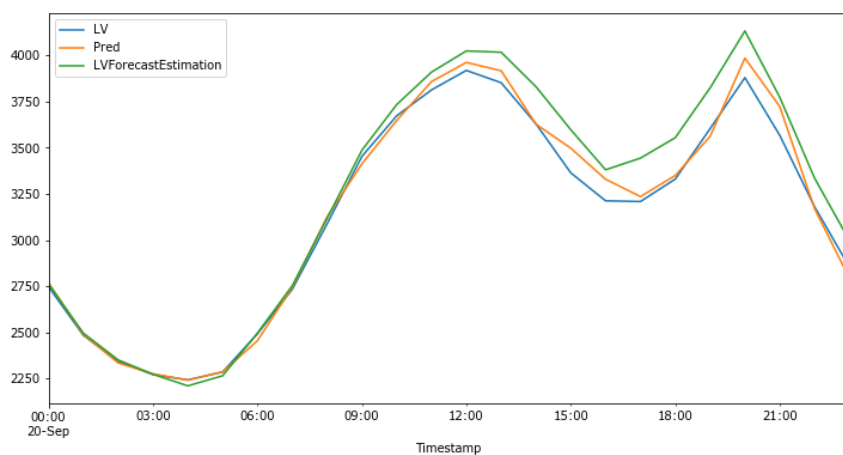


Figure 11.19: LightGBM LV load forecasts vs Actual load vs OoEM's load forecast

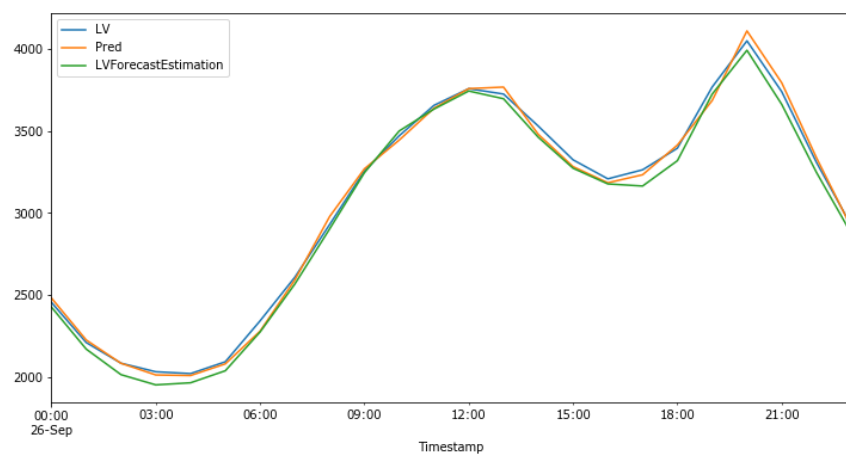


Figure 11.20: LightGBM LV load forecasts vs Actual load vs OoEM's load forecast

Chapter 12

Conclusion & Future Work

Various models utilizing time-series methods and machine learning techniques for Short Term Load Forecasting have been proposed in the literature, over the past years. There is still an essential need for the Independent Power Transmission Operator S.A (IPTO) and market participants (energy providers) in electricity markets, to develop accurate and robust forecasting models, that can capture the uncertainties in the load demand, the seasonal load demand variations and the load demand variations of special days, such as holiday. Furthermore, these models should take care of volatility and non-stationarity issues in the electric load demand, especially when the whole network is progressing towards smart power systems with a lot of smart grid technology systems already part of modern power grid. As mentioned earlier, the main objective of this work is to provide IPTO with an accurate and convenient short-term load forecasting (STLF) model, in order to increase the power system reliability and reduce the system's OPEX. On the other hand, this model should be used by energy providers for making more accurate forecasting concerning the daily Low Voltage electric load declaration. In the electricity market nowadays, the energy trading system and the spot price establishment are based on accurate load forecasting. Energy providers should optimize this process and constantly improve their models for more accurate forecasts. As seen in the literature, many machine learning, deep learning and statistical modeling approaches exist, concerning STLF.

In this research, we proposed a short-term electricity load forecasting model using some state of the art algorithms and using the original hourly load demand data from 1-1-2018 to 31-09-2019, which obtained from the Independent Power Transmission Operator S.A (IPTO). Applying four of them (Time-Series SARIMAX, Prophet, Neural Networks (LSTM) and Light Gradient Boosting Machine-LightGBM Regressor), the forecast mean squared error was smaller in contrast with the OoEM's rmse, with the best prediction coming from ensemble models and more specifically from LightGBM Regressor. In the last chapters, the forecasting results of our experiments are presented and we observed that the proper selection of the model orders (concerning SARIMAX), the architecture and the hyperparameter tuning (concerning LSTM Neural Network) and the optimal parameter's value choice (concerning LightGBM Regressor) is thought to be crucial for successful electric load demand forecasting. Finally, the proposed model (LightGBM) provides more accurate forecasts concerning Low Voltage hourly load demand. The performance measures evaluated in this thesis are, to the best of the author's knowledge, offer a good indication of how each one of the proposed algorithms can be optimally utilized.

Time series forecasting in general and especially Short Term Load Forecasting, is a fast growing research area, thus providing a vast majority of alternatives for future work. Moreover, concerning our experiments, in some cases, significant deviation is observed between the actual observations and our predicted values. Thus, we can suggest that a more suitable data preprocessing may improve the forecast performance. Moreover, other factors affecting electric load demand such as weather data -which was unavailable for the period under study- could be included in the building and improvement of the model. Furthermore, the integration of some economic metrics could be facilitated, as the economic environment has a significant influence on consumer behavior, for example economic growth lead to increased demand for electric load. Last but not least, future work may include actions such as modeling the influence of historical data length on forecast accuracy, studying alternative forecasting algorithms and combining time series forecasts in hybrid models e.g combining different algorithms and models in order to improve forecasting accuracy.

More specifically, many techniques proposed in the literature have not been developed in this work and more importantly this thesis still has some future work to do, concerning the problems faced by the entire STLF area, concerning the learning and training procedures. The results and the forecasting algorithms proposed in this research concerning short term load forecasting should be compared in a realistic setting before any dependence on the forecasts can be justified.

Appendix A

Appendix

Bibliography