Athens univeristy 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 26, 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, Gradient Boosting, 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

List of Figures

Chapter 1

Introduction

Nowadays, it is quite clear that the current lifestyle imposes a greater need for electrical energy supply. In our modern urbanized world, population is rapidly increasing and technology is developing in a fast pace leading to an era of industrialization. Such reality imposes a greater need for electrical energy, however, current technological advances do not support its storage from sources like renewables (solar power, wind power etc.), natural gas, coal or nuclear power and therefore it is still not possible to provide a high quality, fast and efficient storage, distribution and supply network. Additionally, studies have shown that the high demand on energy has a significant impact on the world economy. Hence, a lot of research has been conducted on sustainable sources of electricity, ways to balance demand and supply without unfortunate slow-downs as well as stead electricity production. Due to the continuous demand for electricity, the lack of storage and constraints in resources but also the need for immediate electrical consumption the moment it is produced, electricity market actors have started to search and develop action plans and practices for an equilibrium in electrical demand and supply.

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. In this context, planning includes practices aiming at the cost of losses along with the cost minimization of the substations, the sub-transmission networks, the laterals, and so on.

Even though supply and demand play a significant role is this situation, especially in financial terms, there are still other key factors. In detail, the external environment given that electricity generation is depending on external resources, or the legislation and state regulations that lead to limitations on electricity consumption. Having all these in regard, the concept of planning is integrated in activities while producing, transmitting and distributing energy for the prevention of energy shortage or surplus. Hence, it is very important to be able to forecast accurately the load demand, not only for consumers and suppliers but also policymakers so as to plan, regulate and control the operation of electricity power systems. Thus, load forecasting is an indispensable section of designing, planning and operation of electric utilities, and moreover necessary so as to allocate considerable amount of electric energy in order to increase the economy significantly by electric power manufacturers. In general, electricity load forecasting offers an estimation of the amount of electric load in a given period according to the system data provided. In a broader sense, it seems that numerous areas in power system energy management 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 distribution system planning focuses on ensuring that the increasing load demands along with 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:** It seems that the load growth in a particular geographic area served by a particular utility significantly influence the distribution system expansion. 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

Over the last decade, the energy market in the European Union in general and more specifically the electricity market, have changed significantly. During this period and especially from 2007, the energy market in the EU has been liberalized; it has moved from theory into practice with the activation of relative provisions of the second liberalization directives ("Parliament Council Directive concerning Internal market in electricity's common rules", 2003). The traditional market activities have been enriched with the concepts of energy supply and energy production, with competition on market shares or trading, including the exchange of relevant energy derivatives and products. Practically, the process of energy liberalization has not been completely implemented yet around the EU members and has to overcome many restraints.

2.1 The Traditional Model of Energy Market

Traditionally the common model in this market could be described by state-owned companies that were involved in all stages of electrical energy supply chain: they were producing, transmitting, distributing and retailing but also they were responsible for the supply of all customers in a specific region. In fact, they were considered to be autonomous Electric Power System (hereinafter EPS) but in their own right; they were vertically integrated. Also they acted as a Transmission System Operator (hereinafter TSOs), being responsible for the forecasting of the cumulative load, the central distribution and balancing any differences among energy consumption and production. Given that under these conditions, market players had no motivation or interest in up-to-date technologies and innovations or find ways to minimize the cost of production. Therefore, a state of monopoly has been formed, with a direct impact on KWh pricing and especially the end consumer [1].

2.2 EU and electricity market liberalization

Market liberalization has started in 1999 in the EU and ten years later the relative legislation was adopted. The Directive of the European Parliament and Council "concerning internal market in electricity's common rules" (hereinafter "Directive 2009/72/EC"), also known by the title "Third Energy Package", focus on the gradual liberalization of the market. Its main goal is to offer space for more and new players to enter the market, boost competition and as a result minimize price limitations; in other words, the establishment of a sustainable, competitive but also reliable network of energy supply based on new technology and innovation. According to Serena (2014), the introduction of this directive is significant, however, there are great challenges to overcome as well as benefits to be offered for energy-related partners and consumers [2].

It appears that EU energy market presents high concentration of power. The EU has attempted to reduce it and set limits and clear distinction among sections like production, distribution, sales or transmission through a different perspective in ownership: private companies. Additionally, this distinction has been set through vertical separation in the same field of activity. For instance, competition and liberalization in the whole sale and retail energy market offer more space for business models and new entries of producers. Therefore, the TSO and Distribution System Operators (hereinafter DSOs) have become more independent and have turned to more commercial activities

like sales and production. Concerning DSOs, it seems that there is deviation among EU states as in some regions they are outnumbered whereas in others there are very few. On the other hand, though, there is equal access to the transmission system for all market participants, which is supervised by the independent National Regulatory Authority which each member state establishes, under the framework of the Agency for the Cooperation of Energy Regulators (hereinafter ACER) (Market legislation, 2009) [3, 4].

As we have previously mentioned, the new conditions in the energy market have enable the emerge of new business models. It seems that investments into energy infrastructure have increased and have boosted interest in other energy sources as well as cross-border trading. We could say that there is a new vision for the produced energy, of a European Internal Electricity Market, however, for now, it is traded in regional markets, including:

1. **Day-ahead market:** It refers to electrical energy bought a day in advance (according to DSO's prediction) and even though it is insufficient and at elevated price, it is bought so as to ensure loss minimization. This type is regarded as a spot market type.

2. **Intra-day market:** This type is identical to day-ahead market, also a spot market type, however in this case, retailers buy electrical energy at a very short notice with an accordingly higher price and have it delivered within an hour or two. difference that energy is bought at very short notice and delivered one or two hours later.

3. **Long-term market:** This term refers to standardized electrical energy products which are traded within a period up to 3 years in advance. These products can be provided every week, every month, every three months or every year and can be related to peak, base and off-peak energy. This type of trading offers the opportunity to retailers to buy more energy and have less risk in terms of price.

4. **Balancing market:** In order to describe in brief this type, it is worth to mention that after "gate closure", supply and demand imbalances are settled by a TSO, approximately in real time. More details on this type will be discussed later.

Generally, these four markets represent places where electricity is delivered physically as a commodity (physical market). However, there are other markets called derivative markets in which producers' prices (derivative parts) are dependent on the offered products' prices. Nevertheless, the reasons for participating into trading processes in electricity market may vary a lot; from attempt to minimize and secure price risk to speculation.

2.3 Balancing Market

Although electricity is regarded as a commodity, it differs a lot from other traded goods. Currently, energy has to be consumed as soon as it is produced because the available technologies are insufficient in terms of large quantity storage ("bulk storage"). It is necessary to maintain a balance of power supply and demand for the grid frequency and stability for the whole power grid. In the unfortunate event of the smallest deviation from the 50 Hz, the whole system may become unstable or lead to black-outs. For this reason, there are very strict balancing limitations and penalties in case they do not fulfil the requirements. Furthermore, there are more challenges for the energy operators as the Renewable Energy Sources (hereinafter RES), including solar and wind power, are integrated into their activities. RES display intermittent behavior, and have an impact on the flows within the grid by making it difficult to secure energy supply, available margin in case of emergency and the general smooth operation of the market.

TSOs are legally obliged to maintain the grid frequency continuously. Additionally, frequency is controlled by means of manual and automatic regulation mechanisms. In this context, producers provide an active power capacity which is maintained by the TSO in-store made and it is possible to activate it when needed so as to return the grid into a balanced state. In the EU, these reserves can vary in terms of response time and function. The first level of reserve aims at ceasing the frequency drift if something unexpected occurs like a power plant to go down. Then the second level of reserve is responsible for restoring the frequency to normal levels and the last reserve focuses on repairing imbalances that can be fixed within hours [5].

The last two reserves are considered as ancillary services and they are included in a broader collection of measures. These services were the unique way for supply and demand balance, before

energy market liberalization. In this case, all costs for the procurement of these services were integrated in the end users through the price and, in part, the network and imbalance charges. But the balancing market was introduced through the new legislation framework. This new market balances supply and demand of electrical energy and trading has changed into a form of bidding for those who are interested in selling or buying energy. In the next section it will be discussed the origin of this balancing energy (demanding or supplying) in more detail.

In the European Union, there are currently available only balancing/reserve markets of a national state. However, the EU envisions a pan-European Exchange of Balancing Energy. However, it seems to be a very complex and up-scaled task, so member states have adopted a gradual implantation plan. It appears that the first step includes the regional coordination which will lead to merged balancing market of these regionals. Regarding limitations, it is true that each country presents a variety of balancing cost and it is influenced by various factors, among others the share of RES in the grid or even the size of the energy market. The plan of this pan-European balancing market is to be beneficial for all member-states and provide the ability for a balancing cost reduction as well as an accurate demand and supply balance.

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. Since the establishment of the wholesale electricity market in Greece, in 2005, it has been given a form of an obligatory energy pool. During a five-year period where it has gradually improved into a transitional market model, in September 30, 2010, took its final form. This last form represents the complete implementation the 2005 Grid and Market Operation Code and has the title "5th Reference Day" [6].

The Regulatory Authority for Energy (RAE) is the Greek independent regulatory authority which was 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 is 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 natural gas and electricity markets. Under Law 2773/1999. 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 electrical energy from sources like natural gas, renewable energy sources or/and conventional fuels.

Moreover, this regulatory authority has acquired a specific role in the petroleum market, with fixed responsibilities. With the adoption of Law 3851/2010, there were substantial changes in relation to the existing legislative regime governing Renewable Energy Sources, along with 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 on regulating of the natural gas and electricity 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 September 3, 2009, the European Union adopted the so-called "Third Energy Package", which consists of the Regulation (EC) 713/2009 on "Establishing an Agency for the Cooperation of Energy Regulators", the Directive 2009/72 / EC, "on common rules for the internal market in electricity and repealing Directive 2003/54 / EC", the Regulation (EC) 714/2009, "on conditions for access to the network for cross-border exchanges in electricity and repealing Regulation (EC) 1228/2003", the Directive 2009/73 / EC, "on common rules for

the internal market in natural gas and repealing Directive 2003/55 / EC" and the Regulation (EC) 715/2009, “on conditions for access to the natural gas transmission networks and repealing Regulation (EC) 1775/2005” [7].

Under EU law, European regulations be applied directly by the Member States, without the need for their incorporation into national law. On 3.3.2011, these Specific Regulations were put into force. 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). According to the 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 natural gas and electricity are:

- **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 along more than one supplier at the same time, putting limitation the supplier’s choice, the compatibility of the terms of electricity and natural gas contracts with the possibility of interruption, or even 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 aiming at ensuring 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 demand and supply equilibrium on the Greek energy market, the foreseeable demand’s level in the future, the projected supplementary generation, transmission but also distribution capacity of electricity and natural gas under planning or under construction. Moreover, RAE observes Transmission Systems and Distribution Networks, their quality, level of maintenance and reliability, the implementation of actions to meet peak demand and the state of ag Ras energy in relation to the possibility of developing new capacity in terms of production. It also monitors the implementation of actions and regulations related to the possibility of a sudden energy market crisis. Specifically, for natural gas, RAE is approinted as the Competent Authority, responsible to implement the European Gas Security Regulation 994/2010 of the European Parliament and of the Council of 20 October 2010 (L 295).10 and the measures presented in it.

- **Licensing:** RAE decides on the granting, modification and revocation of permits for the pursuit of energy activities based on the specifications of the Energy Law, provided that

equal treatment and transparency, as principles, are respected and considering the unique features 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 country's Long-Term Planning concerning energy, 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 to certify electricity and natural gas 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, need to be approved beforehand by RAE. In addition, it 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 long and short-term incentives for Transmission System Operators and Distribution Networks so as to improve the efficiency of their networks, to encourage the development of the energy market and the supply security.

- **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 requirement to allow third parties to have access or from the obligation to own ownership. To this end, it shall cooperate with the other member states' Regulatory Authorities, 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 distribution and decommissioning and management mechanism of congestion 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 other countries that have an energy interconnection with ours and their Regulatory Authorities.

- **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.)

The separation of the Distribution Department from PPC S.A., was the trigger for the establishment of HEDNO S.A. (Hellenic Electricity Distribution Network Operator S.A.), according to L.4001/2011 and in compliance with 2009/72/EC Directive relative to the electricity market organization with the goal to undertake the tasks of the Hellenic Electricity Distribution Network Operator. HEDNO S.A.'s operation and management is independent, according to the requirements of the legislative framework mentioned above, but it is considered as a complete subsidiary of PPC S.A. In detail, it is responsible to operate, maintain and develop the network for power distribution in Greece. Additionally, it is responsible to assure impartial and transparent access for all users of this network and be a reliable power supplier with continuously quality service improvements.

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. Generally, in countries with mandatory energy pools, the market design presented a new perspective of the day-ahead market and the related balancing mechanism. Therefore, it is easier to distinguish the elements affecting prices and the risks and uncertainties implied. In detail, in the period of the implementation of the transition market the day-ahead market presented a reference spot prince and unit commitment schedule (SMP forecast). The cash-flows were derived by ex-post SMP prices where instead of inserting day-ahead forecasts metered values of inputs were chosen (focusing on plant availability, demand, plant availability and output of renewables); in other words, it was an adaptation of the algorithm for cost minimization applied in the day-ahead. These ex-post prices were applied to the actual quantities produced, meaning the TSO's real-time dispatch, or the quantities consumed.

However, the current market design seems to follow a different direction compared to the total market in terms of settlement via ex-post SMP prices. This market design combines two main characteristics: a) the plant schedules are organized by the submitted load declarations along with the suppliers' charges which are based on the SMP prices and b) the issue of deviations which depend on the TSO's dispatch order and according to their status different or similar they are charged or compensated respectively. Additionally, in case specific limits of deviations' frequency and magnitude are surpassed, then penalties are imposed. But despite the above, in the day-ahead market, it is applied a uniform pricing practice. With this practice the predicted demand is satisfied as it reflects the expensive unit dispatched offer.

On the other hand, pricing in zones has not been applied yet. This practice of two zones for generators could possibly show areas of new capacity as well as problems of the congestion and work as an indicator. Even though participators can make CfDs (bilateral financial contracts), still there are no such contracts for any physical delivery transactions within the pool along with a 150 €/MWh limit imposed to producers' offers. There are also other supplementary mechanisms and rules. For example, compared to the current model where there is a tendency to suppress wholesale price, there is a lower limit for offers, equal to the minimum variable cost of each unit in each trading period. Moreover, there is a mechanism for cost recovery to ensure remuneration for generators' dispatched by the TSO according to the minimum variable cost they declare plus a margin of 10%. In this way, it works as a safeguard measure but also a drawback for those who participate as it diminishes their interest on the level of prices. Furthermore, in order to partially recover the capital costs, there is an obligation for suppliers concerning capacity certificates provided by the generators, titled Capacity Adequacy Mechanism. Before November 2010, these certificates had a value of 35.000 €/MW and after were risen to 45.000€/MW, so as to relieve of generators from any issues caused because of low demand. Another possible future measure is considered to be the adjustment of the certificates' value according to the environmental impact of the facilities and the

technical flexibility.

In terms of market structure, although PPC retained its dominant position over 2011, its market share in supply and generation declined significantly. Regarding generation, in 2010, it focused on a more deconcentrated structure with two new IPP units that operate commercially. This move got supported a year later with two more IPP facilities. On the other hand, it seems that thermal capacity is far from lasting in the future, given that all private facilities have been completed and also any new will be PPC's. There is though a chance for change in case Troika or other measures on PPC's capacity allocation impose plant divestments in the near future. Besides traditional generation, renewables have penetrated the market. The structure of the market is changing and PPC has only a slight share in it. It appears the according to TSO, many plans for investments were suspended due to economic recession. Additionally, the demand for electricity was fairly stabilized at 51872 GWh in 2011, in the interconnected system, presenting a slight decline of less than 1% compared to the year before.

Similarly, it seems that the general national demand was around 61834 GWh (compared to 61817 GWh in 2010), meaning that it was not significantly affected [8]. The last three months present a recovery of the demand with an 2% increase relatively to the data of the last year, due to the environmental conditions and the use of air-conditioning for heating, instead of oil, as it was regarded as cheaper by the consumers. In those two years PPC's market share was significantly low, given the addition of substantial capacity. Also its volume share in the interconnected system, dropped from 85% to 75% in local production, along with its natural gas production which declined by 816 GWh (13.5%). Because of an increase in natural gas prices of 18.4% and a €21 million tax contribution due to the levy imposed on natural gas, the decline did not affect PPC and its cost for natural gas increased by €10 million. Overall, along with the non-connected regions, PPC with its production covered 70.1% of total demand (2010), with a corresponding share of 77.3% and 85.6% in the previous two years. As a result its share in the market was even more suppressed at 67.5% in the first three months of 2012. Additionally, its production along import activity was reduced by 4451 GWh, while in 2010 this quantity had already been reduced by 5123 GWh and the imports were similarly reduced, by 18%.

Considering renewables generation, the water scarcity affected PPC's small units and its production stayed low (246 compared to 374 GWh) whereas independent renewable production was up to 3958 GWh. Also the wind parks which are connected to the high-voltage grid submitted 2535 GWh. In general sector of renewables is regarded as an attractive investment, with the level of feed-in-tariffs being a major incentive, despite the fact that their levels are adjusted downward for new capacity. Moreover, the bureaucracy for licensing has been gradually diminished. It is interesting that secondary reserve and stand-by reserve are expected to play an important role in the economy of thermal plants as long as the level of wind penetration will be 7500 MW by 2020 and the nature of its operation remains intermittent. Such return was not possible in 2011 because facilities of natural gas was competing for reserve provision, cost-recovery payment and submitting reserve prices were very low. For this reason, supplementary mechanisms influence ancillary prices and need to be regarded within context, not isolated.

Chapter 3

Basic Concepts of Load Forecasting

The overall load of all consumers simultaneously is considered to be the system load and it is related to the STLF system is used for the forecasting of the system load in the future. The in-depth comprehension of their features supports the design of models for forecasting and their appropriate application per case.

3.1 Electric Power System Characteristics

The transformation of energy from different sources including fuels (oil, gas, coal) or nuclear power into electricity is called electrical power. However, the RES, the use of renewables like wind, water, sun etc. is on the rise. The electric power is provided up to the consumer through a network, called the power grid, and includes the transmission grid and the distribution grid. The first one is managed by transmission system operators focusing on transmitting power in long distances by operating at high and extra-high voltage. The second one is managed by distribution system operators focusing on distributing electricity in specific regions, supplying systems of low-level or customers that are connected directly to it. Therefore, it operates partly at high voltage, and mainly at medium and low voltages. The limits between the two grids are determined nationally. The DSOs and the TSOs, as the system operators are called, have the legal obligation to safely and reliably supply the system. So the electric power system or power grid is considered to be the network in which occurs the supply, the transmission and the use of power [9].

The objective of load forecasting in short-term emphasizes on finding out the load variation trend and predicting future development by using correct prediction methods. Its main characteristics are presented below:

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 for a specific period of time scale, like minutes, hours and days. In this way, the temporality is its one main characteristic.

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

related methods are developed. The load forecasting is according to following the real-time data. This 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 forecasting methods and make corresponding adjustments based on the previous model.

3.2 Developed STLF Methods Classification

In the context of lead time, load forecasting is categorized as follows:

- Long-term forecasting (lead time $>$ 1 year)
- Mid-term forecasting (lead time: 1 week – 1 year)
- Short-term load forecasting (lead time: 1 day-1week)
- Very short-term load forecasting (lead time $<$ 1day)

It seems that there are different categories of forecasting for different purposes. The current study emphasizes on the short-term load forecasting which serves the next day unit commitment and reliability analysis. The short-term load forecasting consists of two categories, namely the **artificial intelligence methods** and **the statistical methods**. Beginning with the latter, equations can be obtained showing the relationship between load and its relative factors after training the historical data. On the other hand, artificial intelligence methods focus on the imitation of the way human beings think, retrieve knowledge from their past experience and forecast the future load. The statistical approach uses stochastic time series, multiple linear regression, general exponential smoothing, state space, etc [10]. In recent studies, a very interesting statistical learning method, support vector regression (SVR), has been applied to short-term load forecasting with good results. Generally, statistical methods are good at the prediction of the load curve of ordinary days, but given that they do not have a flexible structure, they are not good in the analysis of the load property of holidays and other irregular days. The artificial intelligence methods include artificial neural network (ANN), expert system and fuzzy inference. Starting with artificial neural network, it does not require the expression of the human experience and its aim is the establishment of a network between the input data set and the observed outputs. It seems effective with the nonlinear relationship between the load and its relative factors, but the shortcoming lies in overfitting and long training time. Moving to expert systems, they try to retrieve the knowledge of experienced operators and express it in an "if. . . then" rule, however, there are cases that the experts' knowledge is intuitive and cannot be expressed in such form. Last but not least, fuzzy inference, is considered as an extension of expert systems. It uses the experts' experience to form fuzzy rules, by creating an optimal structure of the simplified fuzzy inference so as to decrease model errors and the number of the membership functions to grasp nonlinear behavior of short term loads.

In general, artificial intelligence methods have greater flexibility in identifying the relationship between load and its relative factors, especially for the irregular load forecasting [11]. In this section, types of load forecasting are illustrated in the following table in details. The emergency analysis or study of input load flow in the day-to-day operation like scheduling of energy transfer and management of demand need to be considered for short term load forecasting. So as 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 in a significant part of the Energy Management System (EMS) with power markets development over the past decades.

Under normal circumstances, depending on the forecasting purposes and period, load forecasting can be distinguished into **long-term load forecasting**, **medium-term load forecasting**, **short-term load forecasting**, and **ultra short-term load forecasting**. Usually, the power consumption and load can be predicted by adopting these different time-scale forecasts [12]. Regarding the **ultra-short-term load** forecast period is generally within one day (24h) and it is usually targeted to in predicting the load capacity for 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, automatic generation control (AGC) and real-time economic dispatch. Although there are few elements that can influence the ultra short-term load forecasting, only in summer the temperature can be regarded as a main impact factor contributing to the change on the predicted results. In the meantime, the mature methods of prediction are towards allowing for the instantaneous variation rate during the same time interval of previous several days, such as linear extrapolation and exponential smoothing methods.

Concerning the **short-term load** forecast period is generally one or two days or even one week. The target of the prediction is usually the load capacity of a region or the daily and weekly electricity consumption data. Additionally, the forecasting data present daily or monthly periodicity and the same date type of one year following the similar periodic pattern. The main purpose is to organize a day forecast for suspending or restarting power plans or even power generation projects. The short-term load forecasting is mainly affected by weather conditions, week type, national tariff policy etc. Also, the artificial neural network based on the daily data along with the trend extrapolation based on the historical data of the same day are regarded as the mature prediction methods.

In terms of the medium-term load forecast period refers to a period of 5-6 years. Its target is usually the regional load capacity or the electricity consumption per month. The forecast data generally point out a cyclical growth and every month of one year consists with the similar growth pattern. The purpose of the prediction is to organize operation mode, monthly maintenance plan, coal transportation plans and reservoir operation plans. It is mainly influenced by weather conditions, production planning from large users, national tariff policy, industrial restructuring situations etc. Additionally, the mature prediction methods are the time series prediction methods formed by yearly data as well as the trend extrapolation from the historical data of the same month.

Finally, the **long-term load** forecast period is generally 10 to 15 years. The target of the prediction emphasizes on the annual electricity consumption or the regional load capacity. Its purpose seems to identify base data for the power grid planning aiming at the determination of annual maintenance plans and grid operation mode. It is mainly affected by population, national economic development, national tariff policy, industrial restructuring etc. Also, its mature prediction methods include the trend extrapolation and various types of relevant prediction methods considering elements like exponential smoothing, regression analysis, grey prediction and running average.

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 categorized as follows [13]:

- random disturbance
- economy
- time
- weather

Among them, one of the most popular factors is temperature associated with meteorological situations for short term load forecasting. generalized mathematical models are associated with the selection of suitable variables for load forecasting. How accurate such models are, is depended on the quality of input information. Deterministic and stochastic are two different categories of variable. However, 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 safe to say that the system load is complicated and diverse since different social and natural conditions have an impact on it. These social features 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 short-term load forecasting is less influenced by these main factors, including the level of social progress, national policies, the use of energy and the changing of the seasons and other natural factors [14]. 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. Regarding 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 etc. 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 that the accuracy level of short-term load forecasting is related to a combination of factors. 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, it is usually difficult to define the impact of influencing factors on the future load change 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

One of the main research fields in electrical power engineering is considered to be STLF due to its significant role in the power system operation efficiency. 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 [15, 16]. 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 so as 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 this industry has also influenced forecasting and upgraded its level of importance, 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. We have previously mentioned that a number of factors influence the system load including the environment (weather), time, economics, or even unexpected elements. In terms of STLF, it seems that these elements can influence various fields in different ways, forming a new challenge that requires prediction algorithms to be more specified [17]. 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:

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

In the following parts, it will be discussed the way 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 [18].

Time-Series Forecasting Methods

In view of the time series forecasting methods, it seems that present a structure (internal) like autocorrelation or trend or even seasonal variation. So these methods find and examine this internal structure. For decades, they have been applied in electric load forecasting, economics and digital signal processing. For example, the most commonly used time series methods are ARIMAX (Autoregressive Integrated Moving Average with eXogenous variables) and ARMA (Autoregressive Moving Average), ARIMA (Autoregressive Integrated Moving Average). Beginning with ARMA models, they are related to stationary processes but ARIMA, an ARMA extension, is related to nonstationary processes. They both use as their input parameters the load and time. However, ARIMAX appears to be more "natural" regarding load forecasting giaven that load is depending on the time of day and the weather. In this 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

Since 1990, there has been a lot of research on the use of artificial neural networks (ANN or just NN) for load forecasting methods. These networks are actually non-linear circuits that are capable of fitting in non-linear curve. Their results are some non-linear or linear mathematical function of its data. The data could be the result of other network features or even real network data. Actually the elements of the network elements are organized in an approximately small number of connected layers of elements between network inputs and outputs, and sometime they use feedback paths [16]. In order to apply a neural network in load forecasting, it is necessary to choose among the architectures (for example Back Propagation, Hopfield, Boltzmann machine), select the elements and layers' number and connectivity, use uni-directional or bi-directional links and decide the number of format (such as binary or continuous) for outputs and inputs. Generally, back propagation seems to be more frequently encountered among architectures of artificial neural network for load forecasting by using supervised learning and constantly valued functions. Especially, regarding

supervised learning, the actual numerical weights, which are connected with element inputs, are defined by matching historical data like weather or time into desired outputs like historical loads in a pre-operational "training session". But pre-operational training is not required in networks with unsupervised learning.

Similar Day Approach

Similar day approach is related to the search of historical data for a period of time of 1-3 years with common elements to the day of forecast. Such elements are considered to be the date, the day of the week or the weather among others. Also as a forecast can be regarded the load of a similar day. Besides the load of one similar day, a forecast can include various similar day in a linear combination or regression procedure. Additionally, for similar days in the previous years can be used the trend coefficients.

Integration of Different Algorithms

Given the variety of STLF methods that can be used, it is common to add together several methods' results. A way to minimize the risk for a non-satisfying prediction is to find their average value. But another, reasonable and more complicated way is review the results of historical prediction so as to calculate the weight coefficient of every forecasting method. Therefore, a weighted average method leads to a comprehensive result.

3.3.2 Requirements of the STLF Process

It seems that a short-term load forecasting module is present in the majority of energy management systems' modern control centers. A good STLF system should be accurate, fast, detect bad data automatically, have a friendly interface, can access data automatically and generate forecasting results automatically [19].

Accuracy

The most important requirement of STLF process is to accurately predict. Such a characteristic is fundamental for system reliability, economic dispatch and electricity markets. The majority of literature on STLF, along with the current study, is to provide a more accurate forecasting result.

Fast Speed

Accuracy can be increased with the inclusion of weather forecast data and the employment of the latest historical data. By fixing a deadline for the forecasted result, the longer the STLF program is run, the earlier historical data and weather forecast data can be included by the program. As a result, the speed of the forecasting has a significant role for the program. It is suggested not to use programs with an extended training time but to choose new techniques so as to shorten the training time. Therefore, the basic requirement of 24h (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 communication lines transfer the measured data are transferred to the control center. Sometimes the data in the dispatch center are not correct because of the sporadic failure of communication or measurement, so they are recorded incorrectly in the historical database. In the beginning, the STLF systems depended on the power system operators to recognize and dispense these data. Currently, there is a tendency to allow the system itself this process, and not the operators, so as increase the detection rate and minimize operators' work load.

Friendly Interface

Its interface needs to be practical, easy and convenient. Every user has to be able to indicate easily their subject of forecast and the mean for example tables or graphics. Also the output needs to have a numerical and graphical format, so as to be accessible by the users.

Automatic Data Access

Many load-related data like the weather or the historical load are found in the database. The STLF system needs to automatically access the database and locate the requested data. Additionally, it should be able to automatically find on line the forecasted weather, through specific communication lines or Internet so as to minimize the dispatchers load.

Automatic Forecasting Result Generation

Usually, a variety of models is integrated in one STLF system so as to minimize the risk of individual inaccurate forecasting. Due to its characteristics, such a system requires the interference of the operators. This means that the operators ought to choose a weight for every model so as to have a combined result. For the operators' convenience, the system should adapt the final forecasting result based on the forecasting behavior of the historical days.

Portability

It is normal that different power systems will have different properties of load profiles. So, a normal STLF software application matches the area for which it has been developed. The possibility of developing a general STLF software application, able to adapt from one grid to another, could be beneficial especially for the development of different software for other areas. Unfortunately, such a requirement has not been realized yet and it is still regarded as of high-level.

3.3.3 Difficulties in the STLF

Precise Hypothesis of the Input-Output Relationship

The majority of STLF methods have a hypothesis of a regression function (or a network structure, e.g. in ANN) to represent the relationship between the input and output variables. It is very difficult to describe the regression form or the network structure due to the need of a prior knowledge of the problem. The result of the prediction could not be satisfactory in case the network structure or the regression form were not properly selected. For instance, if a problem itself is a quadratic and the linear input-output relationship is supposed, the prediction result will be very poor. In the same context, selecting the parameter can also be a problem. It is not enough to select appropriately the form of the model but also the parameter so as to have a good prediction. In addition, the selection of input variables can be challenging as well, given that exceeding number of variables or selecting less that the adequate could affect the prediction accuracy. Therefore, it should be determined the character of the variables (trivial or influential) for a specific situation. In this case, the variable that do not have any impact on the load behavior (trivial variables) should be abandoned [20].

Generalization of Experts' Experience

Usually the experienced personnel in power grids are quite good at manual load forecasting and they can be better than the computer forecasting. Consequently, it is common to fuzzy inference and use expert systems for load forecasting. However, it is very difficult to transfer the experts' intuition and experience into a database.

The Forecasting of Anomalous Days

When it comes to electric loads of irregular days it is difficult to perform precise prediction given that the load behavior is dissimilar and that there are no sufficient samples as of the other regular days. Such examples are consecutive holidays, public holidays, the days before and after them, special events days and days with extreme weather conditions or sudden weather change [19]. We could say that the load growth through the years might lead to dissimilarity of two sample days, even if the sample number can be greatly enhanced by including the days that are far away from the target day. The result from previous experience show that it is very difficult to forecast days with sudden weather changes. This example day has two kinds of properties namely **the property of the previous neighboring** and **the property of the previous similar days**, so their combination can be very challenging [21].

Less Generalization Ability Caused By Overfitting

Overfitting refers to a technical issue that requires solving during the procedure of load forecasting. Historical training data are employed in the proposed model and it is easy to obtain a basic representation so as to predict the testing data. Regarding the out-coming training module, we identify "overfitting" when the training error for the training data is low but the error for the testing data is high. When a significant disadvantage of neural networks is overfitting, it means that it shows perfect performance for training data prediction and less adequate performance for the future data prediction. So it is suggested that it would be better to avoid overfitting and provide technical solutions as long as STLF aims at the prediction of future and unknown data.

Compulsory Demand-side Management and the destruction of Load Curve Nature

It seems that energy shortage has appeared in various regions due to the financial development and relative lag in power investment. In such situations, compulsory demand-side management is usually applied in order to prevent reliability issues and assure the power supply of very important users. However, this compulsory command intervenes in the natural property of load curve and when this kind of load curve is included in training, it interferes with 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

$$x(t), t = 0, 1, 2, ...$$

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 [22]. 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 $\mathbf{x(t), t = 0, 1, 2, ...}$ is assumed to follow certain probability model which describes the joint distribution of the random variable 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* [23]. 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 [24]. These plots help in determining the order of AR and MA terms. Below we give their mathematical definitions: For a time series $x(t), t = 0, 1, 2, ...$ 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 Coeffient at lag k is defined as:

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

Here μ is the mean of the time series, i.e. $\mu = E[x_t]$. The Autocovariance at lag zero i.e. γ_o 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 fnction (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 [18].

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 stohastic 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 stohastic models have been proposed in literature, nonetheless they are not simplistic and easy-to-understand like the ARIMA model [25].

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 where 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 [26].

- **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 [27].

- **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 [28]. Considering the above mentioned limitations, two different types of models are generally used for Time-Series modelling and these are **Multiplicative and Additive models.**

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

$$AdditiveModel : Y(t) = T(t) + S(t) + C(t) + I(t)$$

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 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 [29] 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 [10]. **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** [30]. Under this principle, the model with the smallest number of parameters among the other candidate models should be preferred, nevertheless

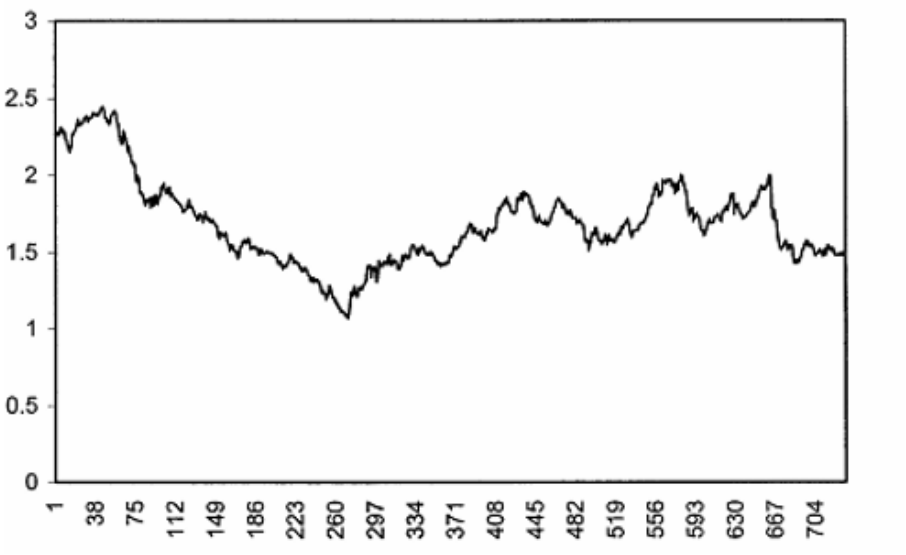


Figure 4.1: Weekly BP/USD exchange rate series (1980-1993)

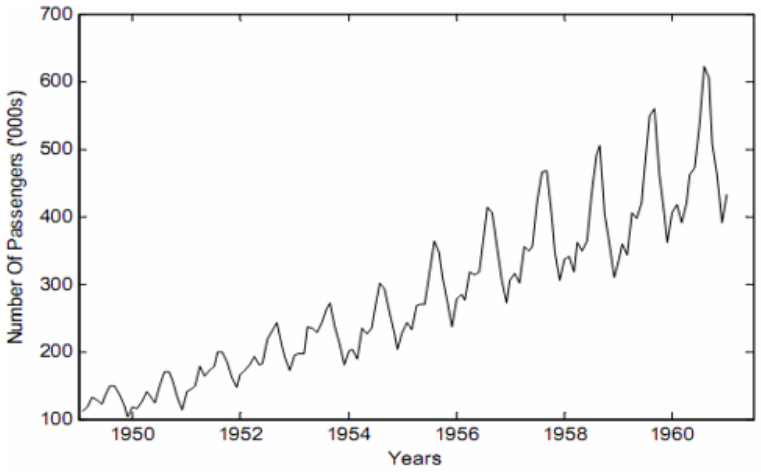


Figure 4.2: Monthly internation airline passenger series (Jan 1949-Dec 1960)

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.

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 [31].

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 [32]:

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

where y_t and t are respectively the actual value and random error in time period t, with i = 1, 2,... 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 .

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.

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

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 considerd to be non-stationary [33]. 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, 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 [34]. 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 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

$$w(B) = w_0 + w_1 B + w_2 B^2 + w_3 B^3 + \ldots$$

and thus defining the Seasonal AutoRegressive Integrated Moving Average with eXogenous factor (SARIMAX (p, d, q)(P, D, Q, m)) [35].

Stohastic 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 stohastic point of view and are based exclusively on the distribution of their past values and, above all, on the most recent ones [36]. 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:

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

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. [37]. 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 it's 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 it's parameters [38]. 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 [37] presents the forecasting method observed to operate efficiently at scale: Prophet has been oftenly used as the

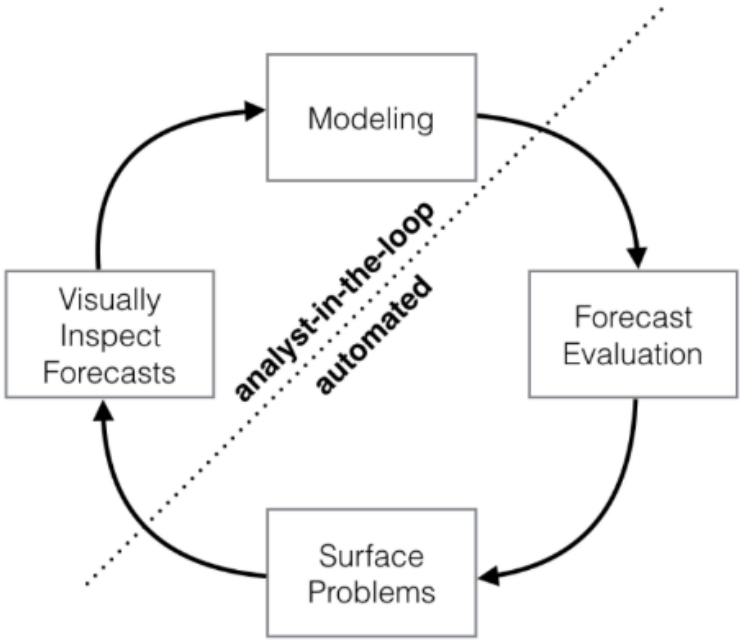


Figure 6.1: Automated forecasting process with analyst-in-the-loop

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 [39]. 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 [37]:

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

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 [40]. 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 [41]. 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 [42], is a simple model for describing the neuron.

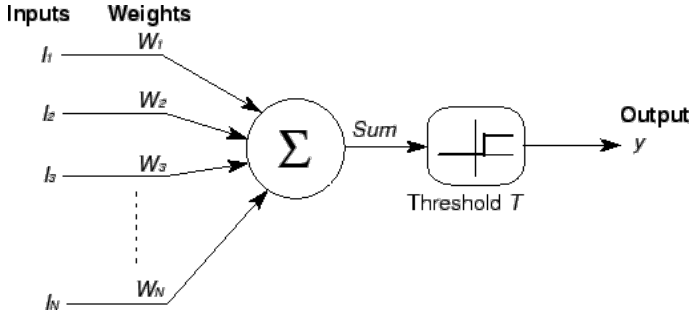


Figure 7.1: The McCulloch-Pitts Neuron model

As seen above, $I_1, I_2, I_3 ... I_N$ are the neuron inputs and $w_1, w_2, w_3 ... w_N$ 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 $wi * xi$ 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 $f(u) = 1/(1+\epsilon^{-u})$. 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 [41]. The layers in between are called hidden layers because they are hidden from the user.

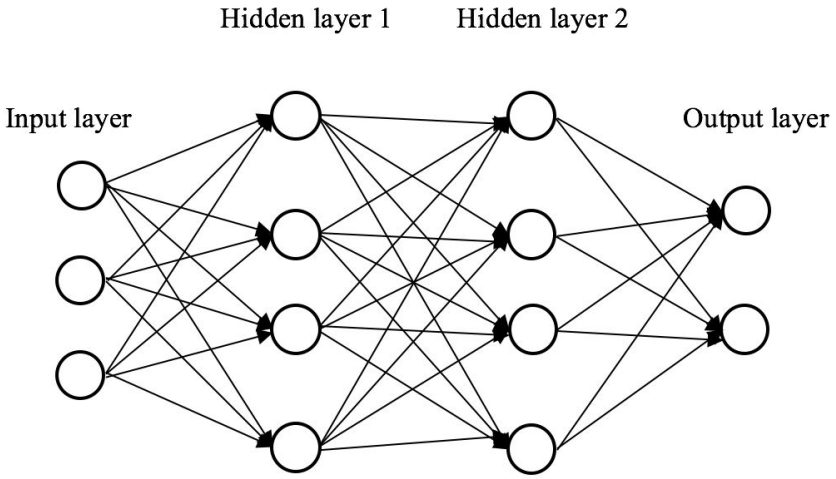


Figure 7.2: Artificial Neural Network's Architecture

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") [43]. 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 [44]. Gradient descent operates under the assumption that we make small steps towards

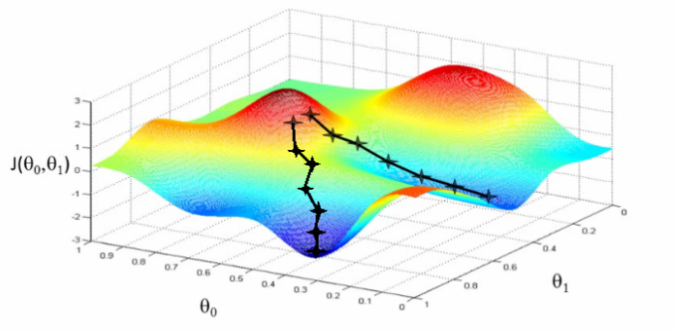


Figure 7.3: Cost function and Gradient Descent

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 [45]. 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 **Feedforward 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 [46].

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 [47]. 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 as shown in the figure below [47].

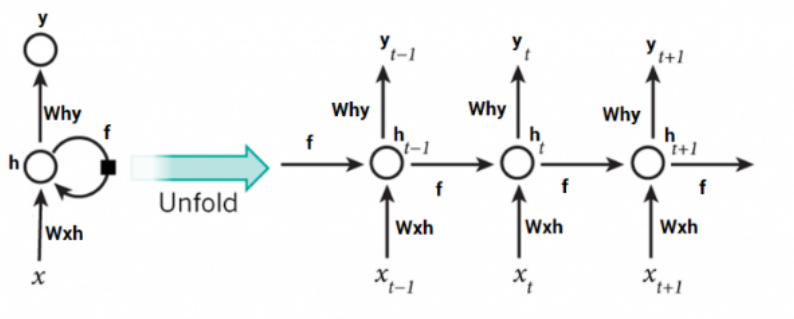
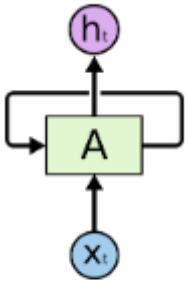


Figure 7.4: Unfolded basic Recurrent Neural Network

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 from feedforward networks, a perception of "memory" as seen on figure 7.5 [48]. **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



Recurrent Neural Networks have loops.

Figure 7.5: Recurrent Neural Network loop

sequential information is 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 [49]. 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 [50].

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 7.6 [48]. 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

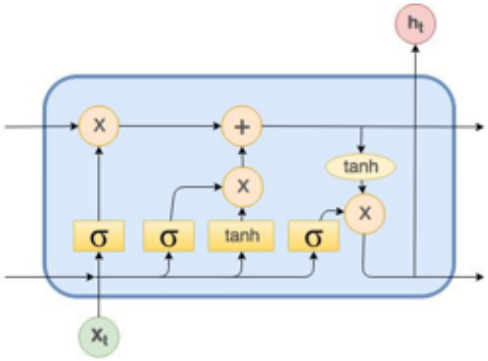


Figure 7.6: Inner structure of an LSTM unit

linear interactions. LSTMs 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 as seen in the following figure [48]: A typical LSTM network is comprised of different memory blocks called cells (the rectangles

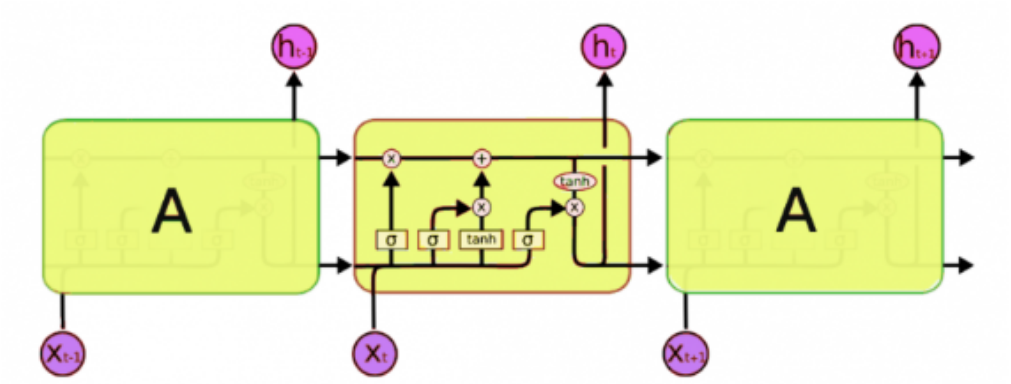


Figure 7.7: LSTM Network Architecture

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 call 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 [51].

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 [52]. 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 figure below [43] illustrates how data flows through a memory cell and is controlled by its gates.

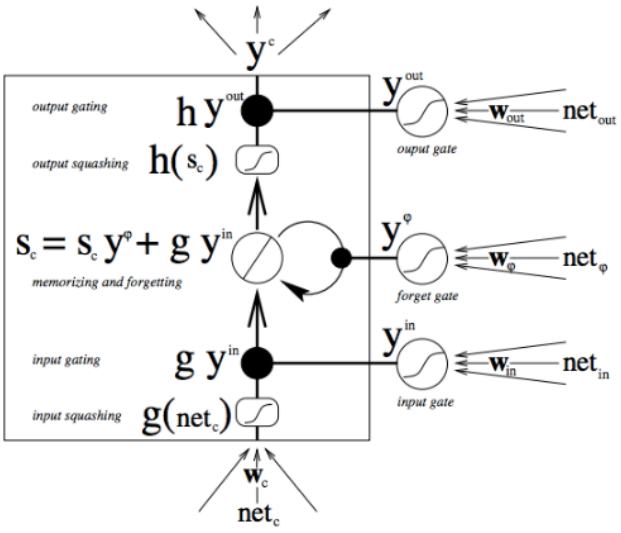


Figure 7.8: Data flow inside LSTM memory cell

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 [20]. 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 [53]. 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 [54]. 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 bootstraping, 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 as shown in figure 8.1 [55].

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 forecsating 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

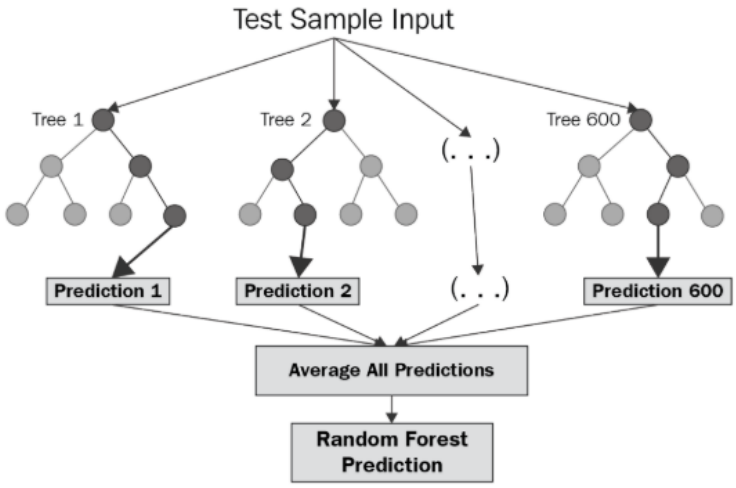


Figure 8.1: Random Forest Structure

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 [56]:

- **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: [57]

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 apect 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 Classsification 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 [58].

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 [59]. 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 [55].

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 section below, we present the pseudo-code of Random Forests in order to understand bit by bit the different stages of the algorithm.

```
Select the number of models to build, m
for i=1 to m do
    Generate a bootstrap sample of the original data
    Train a tree model on this sample
    for each split do
        Randomly select k(<P) of the original predictors
        Select the best predictor among the k predictors
         and partition the data
    end
    Use typical tree model stopping criteria to determine
    when a tree is complete (but don't prune)
end
```

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 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 higly 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 [60]. 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 [61].

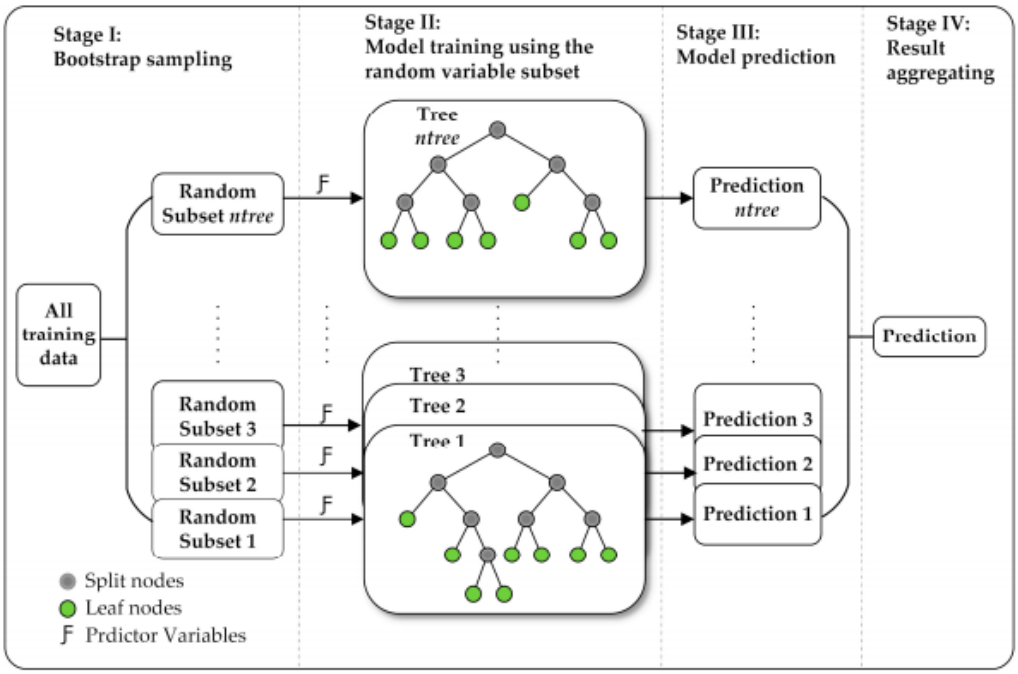


Figure 8.2: Random Forest Regression Algorithm

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 [62]. 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 ntree 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 [63]. 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 others 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 [64]. 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 constract 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 conctruct 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 [65].

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 diffrence 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 nut 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 [66].

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 $\approx$ 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 [67].

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) [68].

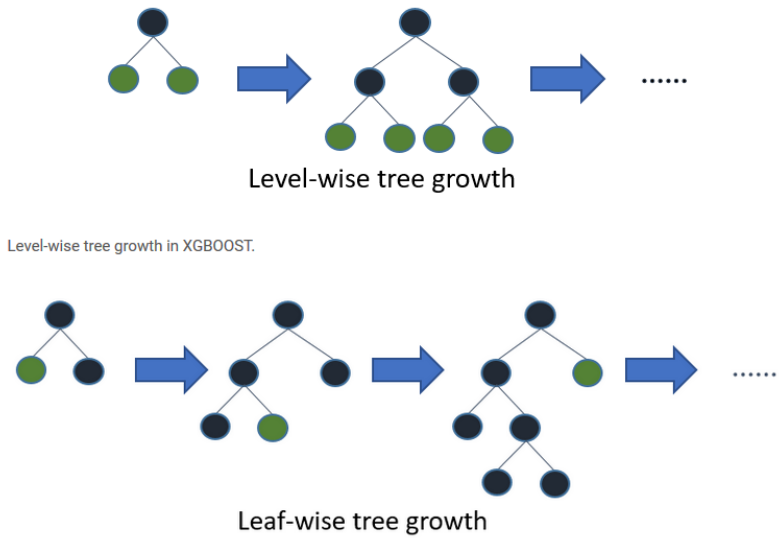


Figure 9.1: Splitting data: Leaf-wise VS Level-wise

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 [69]:

- 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 valuers 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 exante 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 [70]. In general, the 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 [71].

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 [72].

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

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

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 us 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

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

and it's 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. Lust but not least, both of them are negatively-oriented error metrics, thus lower forecasting error means better forecasting performance [73]. 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 mush 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 (on 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.

Date	Hour	Gen_Aux	HV	Pump	TEL_MV	Network	ActualTotal
01-01-18	0	34.42	757.02	0.00	596.56	4529.77	5321.21
01-01-18	1	40.93	760.02	0.00	589.27	4205.13	5006.08
01-01-18	2	39.29	750.62	0.00	585.33	4116.74	4906.65
01-01-18	3	38.68	756.57	0.12	582.21	3833.14	4628.50
01-01-18	4	37.50	731.95	130.18	581.25	3608.46	4508.08
01-01-18	5	37.84	751.93	358.48	592.81	3527.23	4675.48
01-01-18	6	38.72	746.04	365.63	609.29	3600.19	4750.58
01-01-18	7	38.56	729.05	256.67	619.34	3668.00	4692.28
01-01-18	8	37.72	753.26	248.24	614.22	3803.44	4842.65
01-01-18	9	38.38	745.18	11.87	621.18	4196.90	4992.33

These data because there are actual historic data, are provided by IPTO on monthly baisis 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, OeEM 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**.

Weekday	Holiday	Date	1	2	3	...	21	22	23	24
1	1	2018-01-01	5315.0	4963.0	4838.0	...	6198.0	6021.0	5756.0	5513.0
2	0	2018-01-02	5023.0	4612.0	4573.0	...	6831.0	6517.0	6106.0	5717.0
3	0	2018-01-03	5211.0	4761.0	4721.0	...	7409.0	6949.0	6411.0	5962.0
4	0	2018-01-04	5307.0	4788.0	4721.0	...	7483.0	7014.0	6476.0	6038.0
5	0	2018-01-05	5420.0	4932.0	4865.0	...	7453.0	6984.0	6455.0	6037.0
6	1	2018-01-06	5430.0	4952.0	4885.0	...	6401.0	6117.0	5755.0	5550.0
7	0	2018-01-07	5348.0	4890.0	4813.0	...	6863.0	6648.0	6130.0	5710.0
1	0	2018-01-08	5170.0	4711.0	4674.0	...	7543.0	7055.0	6439.0	5952.0
2	0	2018-01-09	5343.0	4903.0	4856.0	...	7588.0	7150.0	6514.0	6029.0
3	0	2018-01-10	5304.0	4836.0	4788.0	...	7540.0	7207.0	6494.0	6027.0

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 occurence, 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.

Timestamp	**LV**	**Weekday**	**Holiday**	**LVForecastEstimation**
01-01-18 0:00	3933.21	1	1	3928.62
01-01-18 1:00	3615.86	1	1	3584.74
01-01-18 2:00	3531.41	1	1	3482.00
01-01-18 3:00	3250.93	1	1	3171.21
01-01-18 4:00	3027.20	1	1	2874.05
01-01-18 5:00	2934.42	1	1	2624.70
01-01-18 6:00	2990.90	1	1	2645.53
01-01-18 7:00	3048.66	1	1	2778.85
01-01-18 8:00	3189.22	1	1	3011.63
01-01-18 9:00	3575.72	1	1	3690.08
01-01-18 10:00	4025.50	1	1	4290.20
01-01-18 11:00	4384.56	1	1	4658.45

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
01-01-18 0:00	1	1	5315	1	2018	0	73.92%
01-01-18 1:00	1	1	4963	1	2018	1	72.23%
01-01-18 2:00	1	1	4838	1	2018	2	71.97%
01-01-18 3:00	1	1	4515	1	2018	3	70.24%
01-01-18 4:00	1	1	4280	1	2018	4	67.15%
01-01-18 5:00	1	1	4182	1	2018	5	62.76%
01-01-18 6:00	1	1	4202	1	2018	6	62.96%
01-01-18 7:00	1	1	4277	1	2018	7	64.97%
01-01-18 8:00	1	1	4573	1	2018	8	65.86%
01-01-18 9:00	1	1	5152	1	2018	9	71.62%
01-01-18 10:00	1	1	5765	1	2018	10	74.42%
01-01-18 11:00	1	1	6130	1	2018	11	75.99%

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 occurence, 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.

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 demannd, 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,

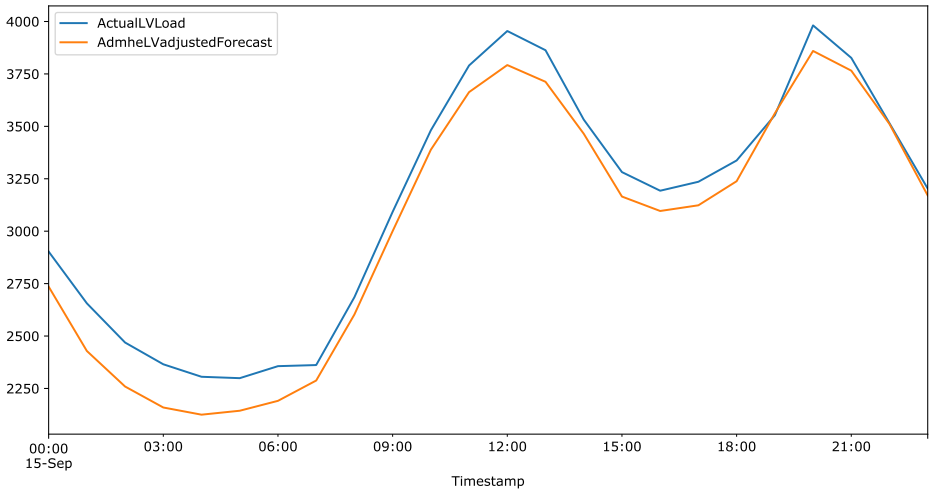


Figure 11.1: Actual LV VS OoEM's LV Adjusted Forecast

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

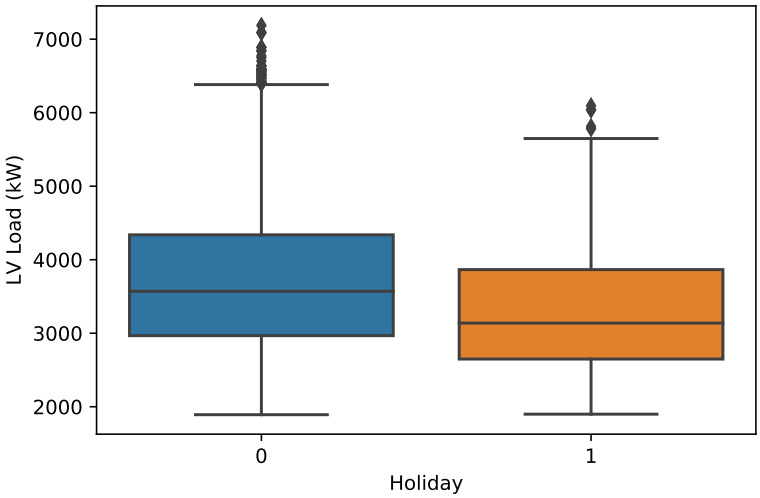


Figure 11.2: Low Voltage Load Demand vs Holiday

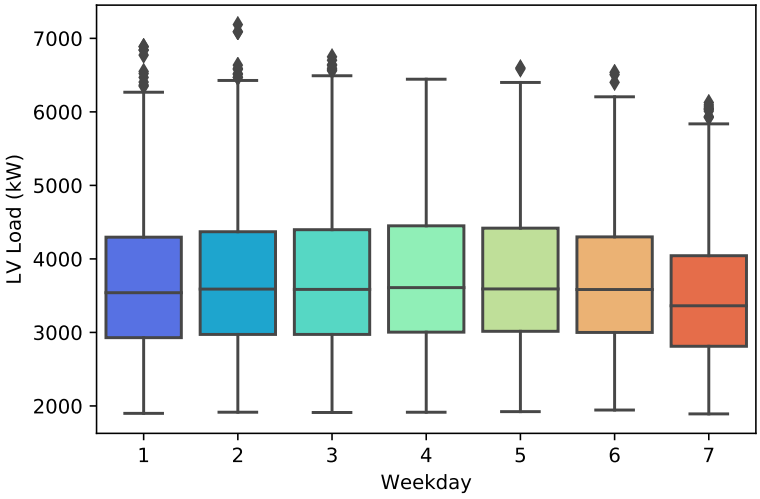


Figure 11.3: Low Voltage Load Demand vs Weekday

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.

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

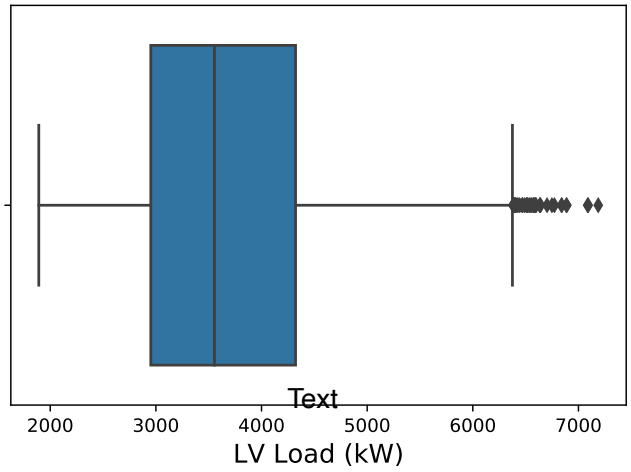


Figure 11.4: Low Voltage Load Demand Boxplot

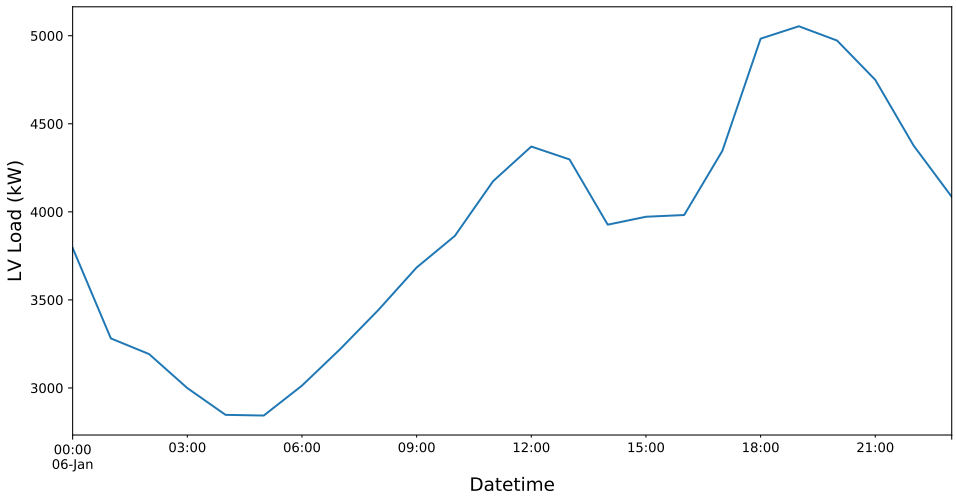


Figure 11.5: Daily Low Voltage Load Demand per Hour

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).

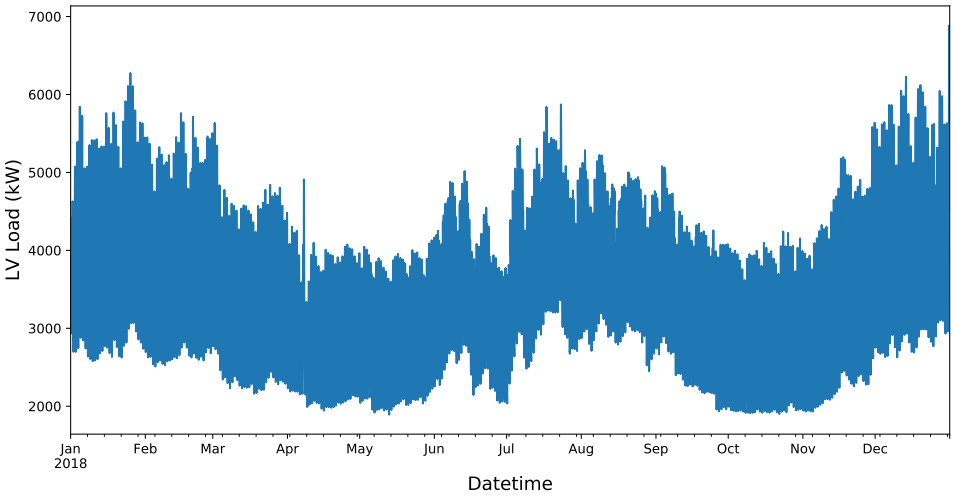


Figure 11.6: 2018 Low Voltage Load Demand (Historical Data)

- 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()* [74] 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)**)

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

from the 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.

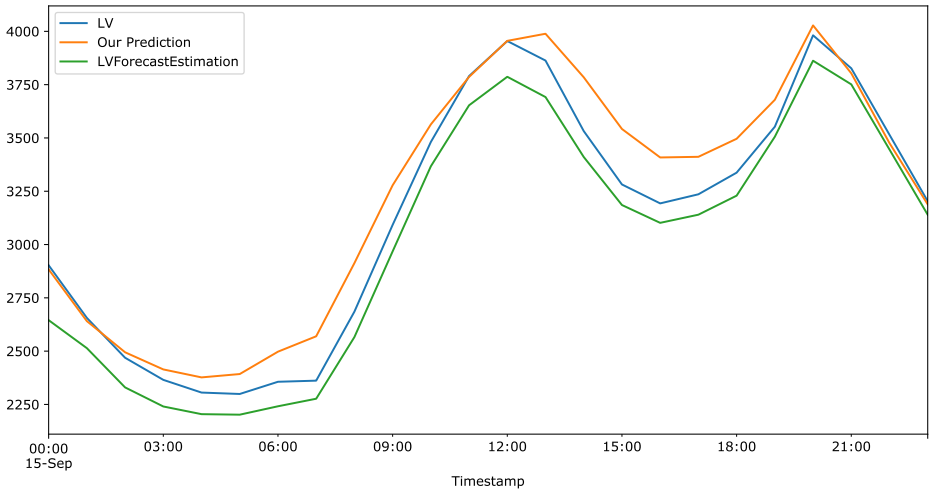


Figure 11.8: SARIMAX LV load forecasts vs Actual load vs OoEM's load forecast

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 [75].

- *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 flexiblity 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 availiable 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.

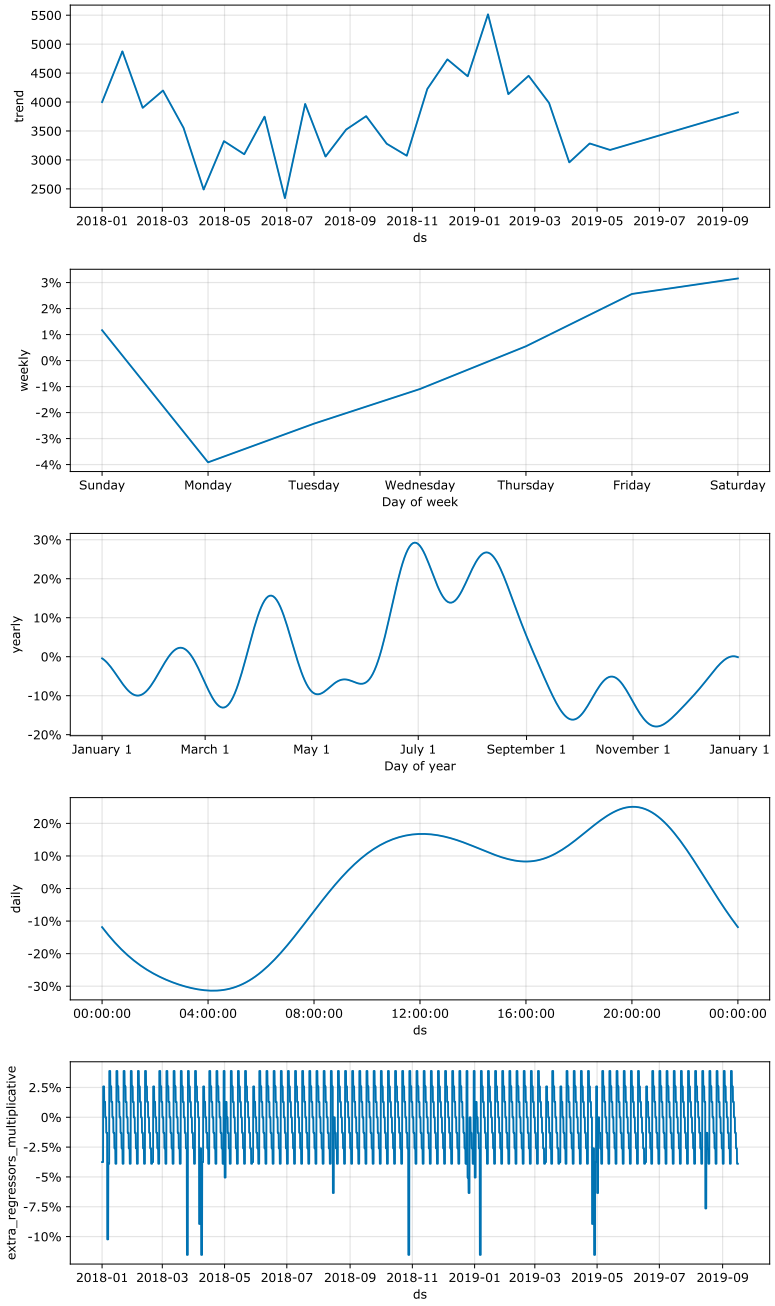


Figure 11.9: PROPHET's components

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 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.

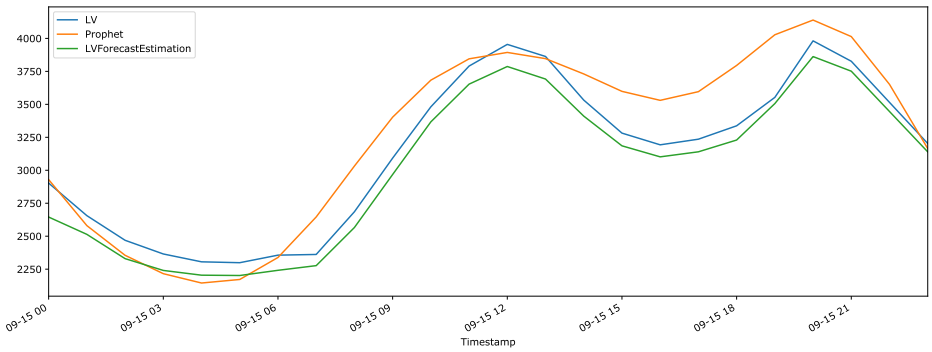


Figure 11.10: PROPHET LV load forecasts vs Actual load vs OoEM's load forecast

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 [76], 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 [77].

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 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 [78]. 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 obversation 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 [77] 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.

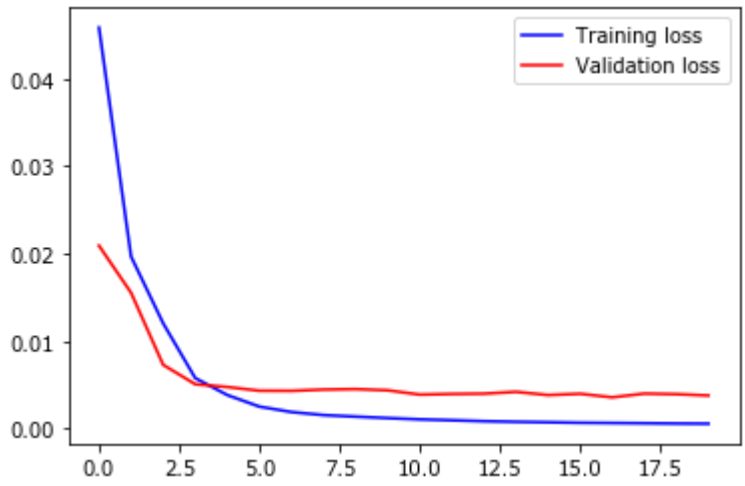


Figure 11.12: LSTM Model's Train & Validation Loss

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 functin (*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 [79]. 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.

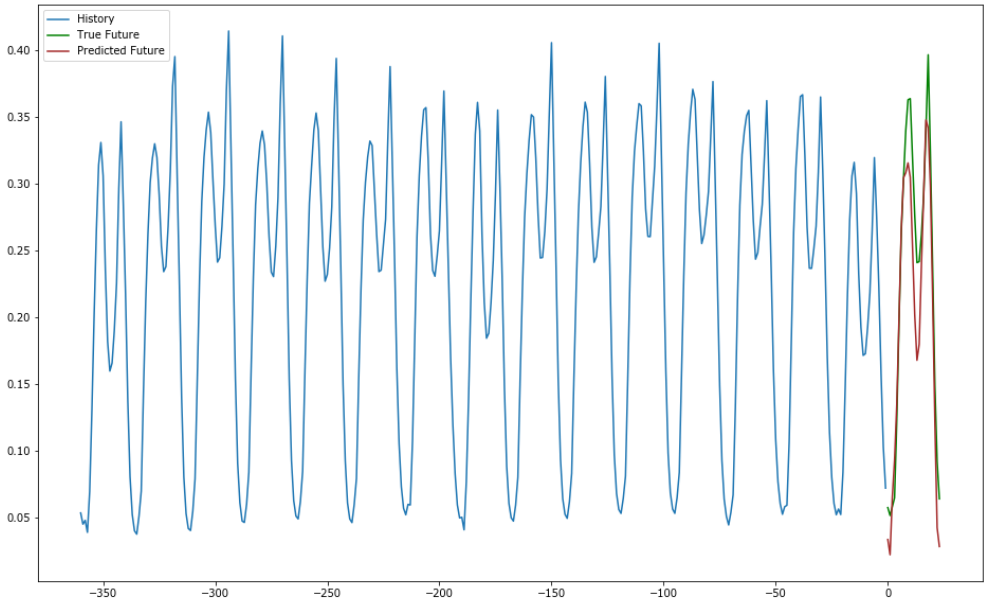


Figure 11.13: LSTM model's load forecasts vs Actual Load

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 [80]. **utocorrelation** metric is used to measures 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 observeved 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.

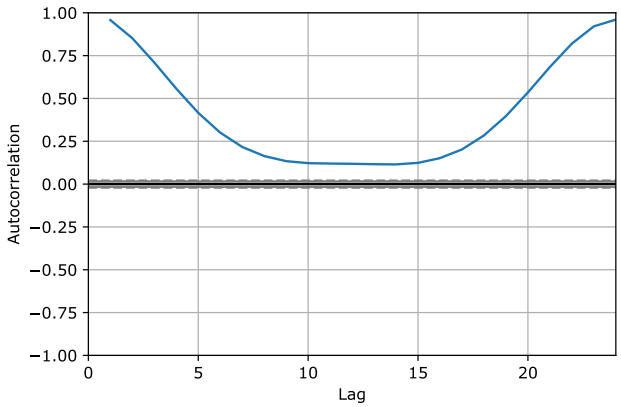


Figure 11.14: Auto-correlation plot over 24 hour period

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.

Timestamp	Weekday	Holiday	Year	Month	Hour	LV	LV_lag1	LV_lag2	LV_lag3	LV_lag4	LV_lag5	LV_lag6
01-01-18 12:00	1	1	2018	1	12	4418.91	4384.56	4025.50	3575.72	3189.22	3048.66	2990.90
01-01-18 13:00	1	1	2018	1	13	4114.95	4418.91	4384.56	4025.50	3575.72	3189.22	3048.66
01-01-18 14:00	1	1	2018	1	14	3588.58	4114.95	4418.91	4384.56	4025.50	3575.72	3189.22
01-01-18 15:00	1	1	2018	1	15	3613.31	3588.58	4114.95	4418.91	4384.56	4025.50	3575.72
01-01-18 16:00	1	1	2018	1	16	3691.87	3613.31	3588.58	4114.95	4418.91	4384.56	4025.50
01-01-18 17:00	1	1	2018	1	17	4014.75	3691.87	3613.31	3588.58	4114.95	4418.91	4384.56
01-01-18 18:00	1	1	2018	1	18	4542.76	4014.75	3691.87	3613.31	3588.58	4114.95	4418.91
01-01-18 19:00	1	1	2018	1	19	4628.66	4542.76	4014.75	3691.87	3613.31	3588.58	4114.95
01-01-18 20:00	1	1	2018	1	20	4618.74	4628.66	4542.76	4014.75	3691.87	3613.31	3588.58
01-01-18 21:00	1	1	2018	1	21	4479.44	4618.74	4628.66	4542.76	4014.75	3691.87	3613.31
01-01-18 22:00	1	1	2018	1	22	4203.66	4479.44	4618.74	4628.66	4542.76	4014.75	3691.87
01-01-18 23:00	1	1	2018	1	23	3931.89	4203.66	4479.44	4618.74	4628.66	4542.76	4014.75
02-01-18 0:00	2	0	2018	1	0	3490.90	3931.89	4203.66	4479.44	4618.74	4628.66	4542.76
02-01-18 1:00	2	0	2018	1	1	3098.82	3490.90	3931.89	4203.66	4479.44	4618.74	4628.66

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 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 [81].

- **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**

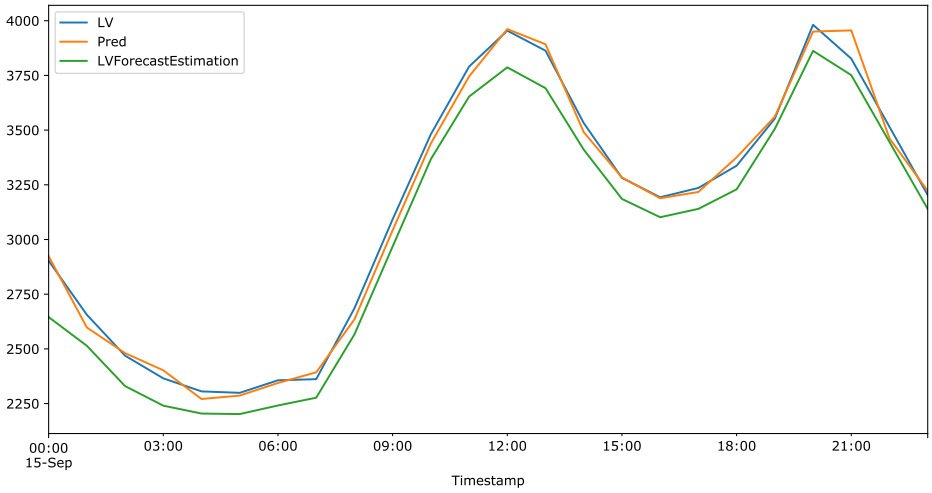


Figure 11.15: LightGBM LV load forecasts vs Actual load vs OoEM's load forecast

Their RMSE Error	Our RMSE Error
123.23	*41.86*

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 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 Semptember 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

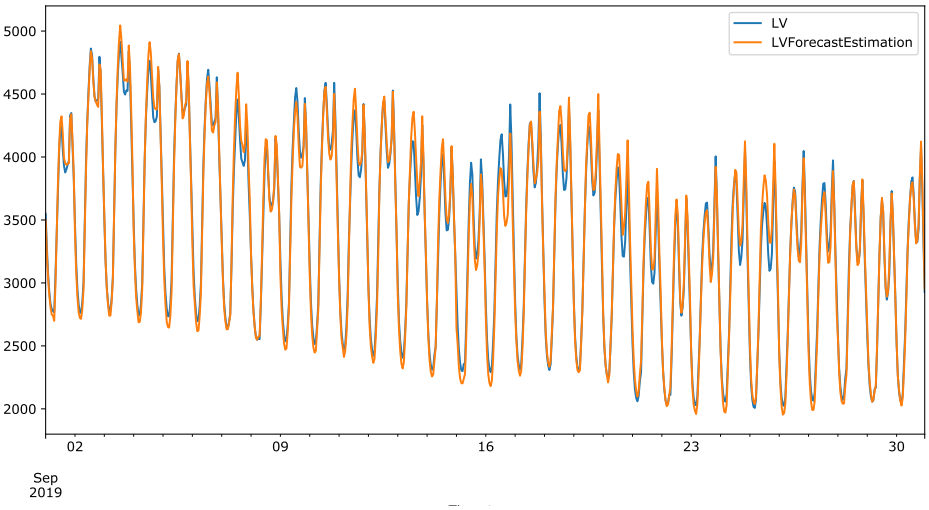


Figure 11.16: Actual LV load demand vs OoEM's LV load forecast

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

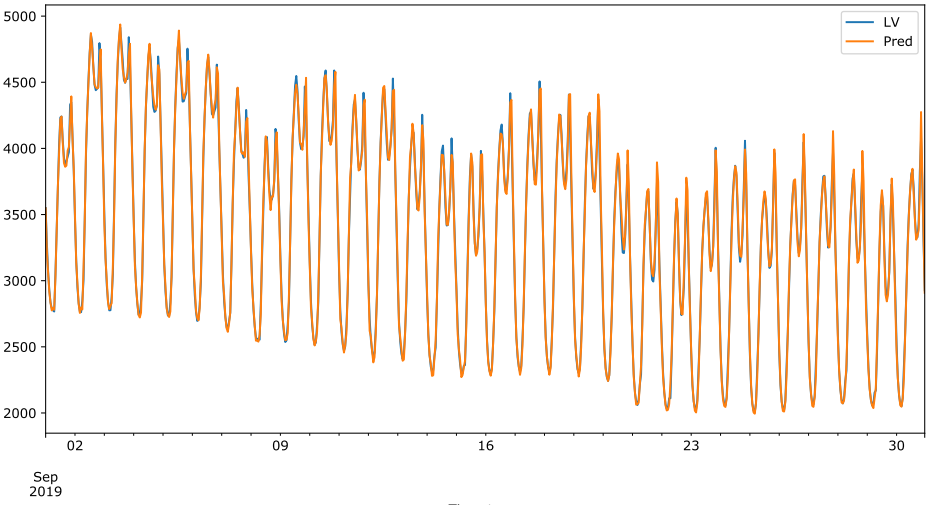


Figure 11.17: Actual LV load demand vs LightGBM model's LV load forecast

Their RMSE Error	**Our RMSE Error**
88.14	*55.73*

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.

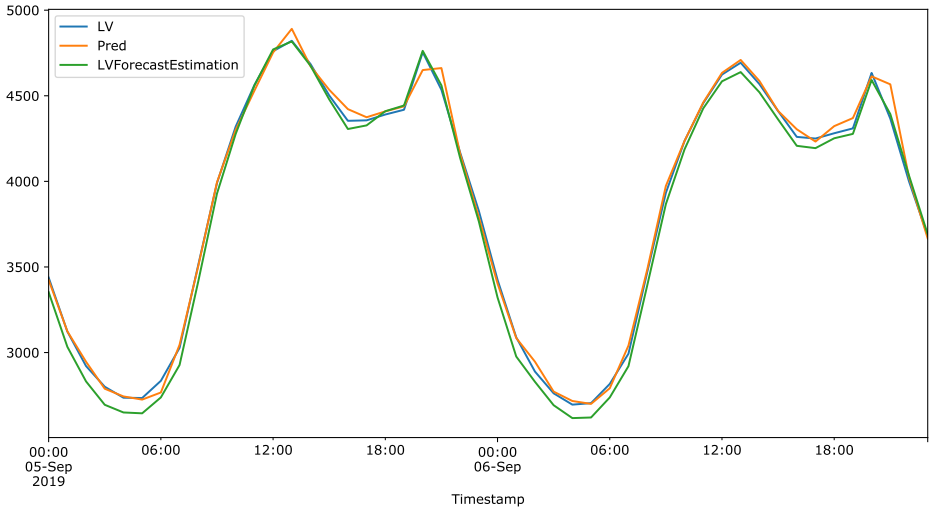


Figure 11.18: LightGBM LV load forecasts vs Actual load vs OoEM's load forecast

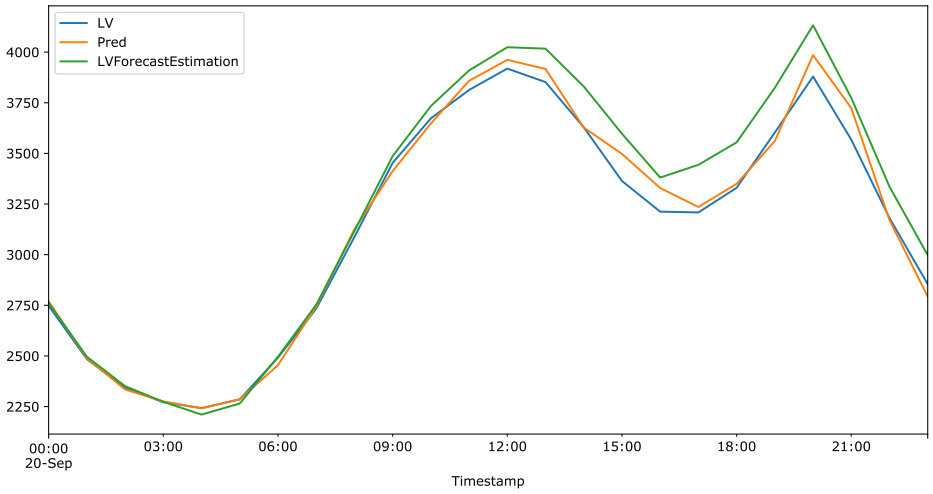


Figure 11.19: LightGBM LV load forecasts vs Actual load vs OoEM's load forecast

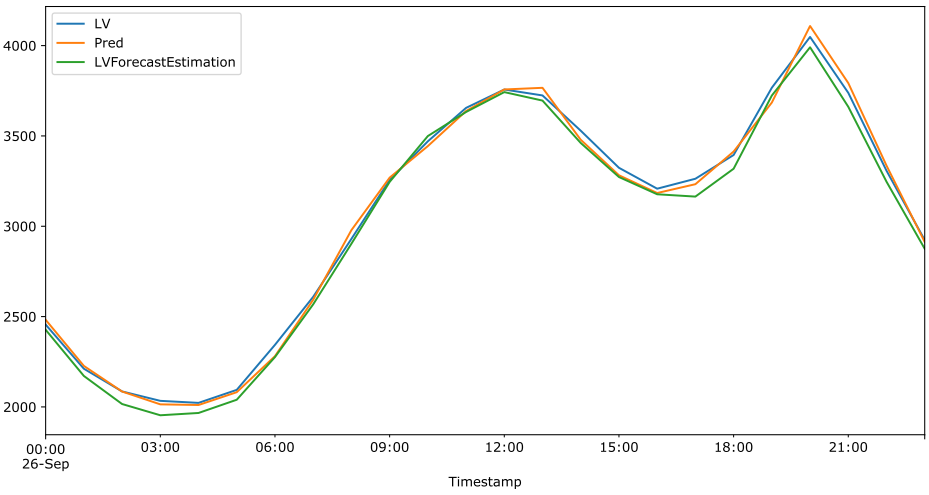


Figure 11.20: LightGBM LV load forecasts vs Actual load vs OoEM's load forecast

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.

Bibliography

[1] energy-community.org.

[2] Welsch Manuel. *Europes Energy Transition-Insights for Policy Making.* Elsevier Science Publishing Company, 2017.

[3] Ulrich Scholz and Stephan Purps. The application of eu competition law in the energy sector. *Journal of European Competition Law & Practice*, 5(2):100–112, 2014.

[4] E Union. Directive 2009/72/ec of the european parliament and of the council of 13 july 2009 concerning common rules for the internal market in electricity and repealing directive 2003/54/ec. *Off. J. Eur. Union L*, 211:55–93, 2009.

[5] Florian Kern and Michael Howlett. Implementing transition management as policy reforms: a case study of the dutch energy sector. *Policy Sciences*, 42(4):391, 2009.

[6] www.energia.gr.

[7] *www.rae.gr.*

[8] *www.deddie.gr.*

[9] VNR VIGNANA JYOTHI. Electrical and electronics engineering. 1999.

[10] Ratnadip Adhikari and Ramesh K Agrawal. An introductory study on time series modeling and forecasting. *arXiv preprint arXiv:1302.6613*, 2013.

[11] Marcelo Espinoza, Johan AK Suykens, Ronnie Belmans, and Bart De Moor. Electric load forecasting. *IEEE Control Systems Magazine*, 27(5):43–57, 2007.

[12] Hesham K Alfares and Mohammad Nazeeruddin. Electric load forecasting: literature survey and classification of methods. *International journal of systems science*, 33(1):23–34, 2002.

[13] Angel Pardo, Vicente Meneu, and Enric Valor. Temperature and seasonality influences on spanish electricity load. *Energy Economics*, 24(1):55–70, 2002.

[14] Heiko Hahn, Silja Meyer-Nieberg, and Stefan Pickl. Electric load forecasting methods: Tools for decision making. *European journal of operational research*, 199(3):902–907, 2009.

[15] Nima Amjady. Short-term hourly load forecasting using time-series modeling with peak load estimation capability. *IEEE Transactions on power systems*, 16(3):498–505, 2001.

[16] Soliman Abdel-hady Soliman and Ahmad Mohammad Al-Kandari. *Electrical load forecasting: modeling and model construction.* Elsevier, 2010.

[17] Chris Chatfield. *Time-series forecasting.* Chapman and Hall/CRC, 2000.

[18] Kasım Zor, Oğuzhan Timur, and Ahmet Teke. A state-of-the-art review of artificial intelligence techniques for short-term electric load forecasting. In *2017 6th International Youth Conference on Energy (IYCE)*, pages 1–7. IEEE, 2017.

[19] Eugene A Feinberg and Dora Genethliou. Load forecasting. In *Applied mathematics for restructured electric power systems*, pages 269–285. Springer, 2005.

[20] Elias Kyriakides and Marios Polycarpou. Short term electric load forecasting: A tutorial. In *Trends in Neural Computation*, pages 391–418. Springer, 2007.

[21] Yan Cao, Zhong Jun Zhang, and Chi Zhou. Data processing strategies in short term electric load forecasting. In *2012 International Conference on Computer Science and Service System*, pages 174–177. IEEE, 2012.

[22] Chris Chatfield. *The analysis of time series: an introduction*. Chapman and Hall/CRC, 2003.

[23] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[24] Rob J Hyndman and George Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018.

[25] Hing Kai Chan, Shuojiang Xu, and Xiaoguang Qi. A comparison of time series methods for forecasting container throughput. *International Journal of Logistics Research and Applications*, 22(3):294–303, 2019.

[26] Daniel T Larose and Chantal D Larose. *Discovering knowledge in data: an introduction to data mining*, volume 4. John Wiley & Sons, 2014.

[27] Douglas C Montgomery, Cheryl L Jennings, and Murat Kulahci. *Introduction to time series analysis and forecasting*. John Wiley & Sons, 2015.

[28] Henrik Madsen. *Time series analysis*. Chapman and Hall/CRC, 2007.

[29] G Peter Zhang. Time series forecasting using a hybrid arima and neural network model. *Neurocomputing*, 50:159–175, 2003.

[30] Foster Provost and Tom Fawcett. *Data Science for Business: What you need to know about data mining and data-analytic thinking*. " O'Reilly Media, Inc.", 2013.

[31] J Kihoro, RO Otieno, and C Wafula. Seasonal time series forecasting: A comparative study of arima and ann models. 2004.

[32] Javier Contreras, Rosario Espinola, Francisco J Nogales, and Antonio J Conejo. Arima models to predict next-day electricity prices. *IEEE transactions on power systems*, 18(3):1014–1020, 2003.

[33] ByoungSeon Choi. *ARMA model identification*. Springer Science & Business Media, 2012.

[34] Moufida Bouzerdoum, Adel Mellit, and A Massi Pavan. A hybrid model (sarima–svm) for short-term power forecasting of a small-scale grid-connected photovoltaic plant. *Solar Energy*, 98:226–235, 2013.

[35] Agostino Tarsitano and Ilaria L Amerise. Short-term load forecasting using a two-stage sarimax model. *Energy*, 133:108–114, 2017.

[36] Nari Sivanandam Arunraj, Diane Ahrens, and Michael Fernandes. Application of sarimax model to forecast daily sales in food retail industry. *International Journal of Operations Research and Information Systems (IJORIS)*, 7(2):1–21, 2016.

[37] Sean J Taylor and Benjamin Letham. Forecasting at scale. *The American Statistician*, 72(1):37–45, 2018.

[38] JC Park, BP Chang, and N Mok. 144 time series analysis and forecasting daily emergency department visits utilizing facebook's prophet method. *Annals of Emergency Medicine*, 74(4):S57, 2019.

[39] Hristos Tyralis and Georgia A Papacharalampous. Large-scale assessment of prophet for multi-step ahead forecasting of monthly streamflow. *Advances in Geosciences*, 45:147–153, 2018.

[40] Zheng Chen, Yin-Liang Zhao, Xiao-Yu Pan, Zhao-Yu Dong, Bing Gao, and Zhi-Wen Zhong. An overview of prophet. In *International Conference on Algorithms and Architectures for Parallel Processing*, pages 396–407. Springer, 2009.

[41] Yuxi Hayden Liu and Pablo Maldonado. *R Deep Learning Projects: Master the techniques to design and develop neural network models in R*. Packt Publishing Ltd, 2018.

[42] Sougata Chakraborty and Debabrata Sarddar. Optimized routing approach using mcculloch-pitts neuron model.

[43] Chris Nicholson. A beginner's guide to lstms and recurrent neural networks. *Skymind. Saatavissa: https://skymind. ai/wiki/lstm. Hakupäivä*, 6:2019, 2019.

[44] https://medium.com/introducing-gradient-descent-minimizing-the-cost-function-65fff3de8ac6.

[45] MHMR Shyamali Dilhani and Chawalit Jeenanunta. Effect of neural network structure for daily electricity load forecasting. In *2017 Moratuwa Engineering Research Conference (MERCon)*, pages 419–424. IEEE, 2017.

[46] Joel Grus. *Data science from scratch: first principles with python*. O'Reilly Media, 2019.

[47] D Gupta. Fundamentals of deep learning–introduction to recurrent neural networks. *Analytics Vidhya (2107)*, 2018.

[48] Christopher Olah. Understanding lstm networks. 2015.

[49] David Palchak, Siddharth Suryanarayanan, and Daniel Zimmerle. An artificial neural network in short-term electrical load forecasting of a university campus: a case study. *Journal of Energy Resources Technology*, 135(3), 2013.

[50] MM Tripathi, KG Upadhyay, and SN Singh. Short-term load forecasting using generalized regression and probabilistic neural networks in the electricity market. *The Electricity Journal*, 21(9):24–34, 2008.

[51] Nigel Da Costa Lewis. *Deep Time Series Forecasting with Python: An Intuitive Introduction to Deep Learning for Applied Time Series Modeling*. ND Lewis, 2016.

[52] Chollet François. Deep learning with python, 2017.

[53] Adele Cutler, D Richard Cutler, and John R Stevens. Random forests. In *Ensemble machine learning*, pages 157–175. Springer, 2012.

[54] Tavish Srivastava. Introduction to random forest–simplified, 2014.

[55] Afroz Chakure. Random forest regression along with its implementation in python. *Towards Data Science*.

[56] Niklas Donges. A complete guide to random foerst algorithm.

[57] Tavish Srivastava. Tuning the parameters of your random forest model. *Analytics Vidhya*, 9, 2015.

[58] G Casella, S Fienberg, and I Olkin. Springer texts in statistics. 2013.

[59] Andy Liaw, Matthew Wiener, et al. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

[60] Mark R Segal. Machine learning benchmarks and random forest regression. 2004.

[61] Ziyao Xu, Jijian Lian, Lingling Bin, Kaixun Hua, Kui Xu, and Hoi Yi Chan. Water price prediction for increasing market efficiency using random forest regression: A case study in the western united states. *Water*, 11(2):228, 2019.

[62] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[63] Shiju Sathyadevan and Remya R Nair. Comparative analysis of decision tree algorithms: Id3, c4. 5 and random forest. In *Computational intelligence in data mining-volume 1*, pages 549–562. Springer, 2015.

[64] Cheng Li. A gentle introduction to gradient boosting.

[65] Zhang Mei, Fei Xiang, and Liu Zhen-hui. Short-term traffic flow prediction based on combination model of xgboost-lightgbm. In *2018 International Conference on Sensor Networks and Signal Processing (SNSP)*, pages 322–327. IEEE, 2018.

[66] Darren Cook. *Practical machine learning with H2O: powerful, scalable techniques for deep learning and AI.* " O'Reilly Media, Inc.", 2016.

[67] Brad Boehmke and Brandon M Greenwell. *Hands-On Machine Learning with R.* CRC Press, 2019.

[68] PRANJAL KHANDELWAL. Which algorithm takes the crown: Light gbm vs xgboost? *Analytics Vidhya*.

[69] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3146–3154. Curran Associates, Inc., 2017.

[70] Aurélien Géron. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems.* O'Reilly Media, 2019.

[71] J Scott Armstrong and Robert Fildes. Correspondence on the selection of error measures for comparisons among forecasting methods. *Journal of Forecasting*, 14(1):67–71, 1995.

[72] J Scott Armstrong and Fred Collopy. Error measures for generalizing about forecasting methods: Empirical comparisons. *International journal of forecasting*, 8(1):69–80, 1992.

[73] Rob J Hyndman and Anne B Koehler. Another look at measures of forecast accuracy. *International journal of forecasting*, 22(4):679–688, 2006.

[74] http://alkaline-ml.com/pmdarima/.

[75] https://facebook.github.io/prophet/docs/quick$_s$*tart.html*.

[76] Wes McKinney. *Python for data analysis: Data wrangling with Pandas, NumPy, and IPython.* " O'Reilly Media, Inc.", 2012.

[77] https://keras.io/.

[78] Weicong Kong, Zhao Yang Dong, Youwei Jia, David J Hill, Yan Xu, and Yuan Zhang. Short-term residential load forecasting based on lstm recurrent neural network. *IEEE Transactions on Smart Grid*, 10(1):841–851, 2017.

[79] https://www.tensorflow.org/tutorials/structured$_d$*ata/time$_s$eries*.

[80] Jake VanderPlas. *Python data science handbook: Essential tools for working with data.* " O'Reilly Media, Inc.", 2016.

[81] https://lightgbm.readthedocs.io/en/latest/.