ARISTOTLE UNIVERSITY OF THESSALONIKI
FACULTY OF SCIENCES
SCHOOL OF INFORMATICS
DEPARTMENT OF COMPUTER SCIENCE
«KNOWLEDGE, DATA AND SOFTWARE TECHNOLOGIES»

Master Thesis:

Machine Learning Techniques for the Short-Term Electric Load Forecasting

Praxitelis-Nikolaos Kouroupetroglou
UID:629

Supervisor:
Grigorios Tsoumakas, Assistant Professor

September 2017

ΑΡΙΣΤΟΤΕΛΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ ΣΤΗΝ ΠΛΗΡΟΦΟΡΙΚΗ
«ΤΕΧΝΟΛΟΓΙΕΣ ΓΝΩΣΗΣ, ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΛΟΓΙΣΜΙΚΟΥ»
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΑΡΙΣΤΟΤΕΛΕΙΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΟΝΙΚΗΣ

Διπλωματική Εργασία:

Machine Learning Techniques for the Short-Term Electric Load Forecasting

Πραξιτέλης-Νικόλαος Κουρουπέτρογλου
ΑΕΜ:629

Επιβλέπων Καθηγητής:
Γρηγόριος Τσουμάκας, Επίκουρος Καθηγητής

Σεπτέμβριος 2017

Abstract

Economic growth in the modern world, depends directly on the availability of electric energy, especially because most societies, industries, economies etc depend almost entirely on its use. The availability of a source of continuous, cheap, and reliable energy is of foremost economic importance. Electrical load forecasting is an important tool used to ensure that the energy supplied by utilities meets the load plus the energy lost in the system. To this end, a staff of trained personnel is needed to carry out this specialized function. Load forecasting is always defined as basically the science or art of predicting the future load on a given system, for a specified period of time ahead. These predictions may be just for a fraction of an hour ahead for operation purposes, or as much as 20 years into the future for planning purposes

The purpose of this master thesis is to predict the short-term electrical load demand forecasting in short-term horizon (for 1 day ahead) for the Greek Electric Network Grid

The load data that the Thesis uses originates from IPTO which stands for Independent Power Transmission Operator. Meteorological Features taken from Dark Sky API and to compare the Thesis predictions, load predictions from OoEM which stands of Operator of Electricity Market in Greece was used.

The datasets were downloaded from its origins, preprocessed, cleaned and six machine learning algorithms was used. After four different types of experiments, the combination of SVM, XGBoost and Model Trees models was used in order to get 2.4% prediction error and on the other hand OoEM gives 5.25% prediction error. Hence the Thesis models improve the prediction performance by 54.9%

The code of this master thesis can be found at this github repo and results - visualizations can be found here: IPTO ML

Praxitelis-Nikolaos Kouroupetroglou
September - 2017

Keywords: Machine Learning, Supervised Learning,Forecasting, Regression

Περίληψη

Η οικονομική ανάπτυξη του σύγχρονου κόσμου εξαρτάται άμεσα από τη διαθεσιμότητα ηλεκτρικής ενέργειας, ιδιαίτερα επειδή οι περισσότερες κοινωνίες, οι βιομηχανίες, οι οικονομίες κλπ. Εξαρτώνται σχεδόν εξ ολοκλήρου από τη χρήση της. Η διαθεσιμότητα μιας πηγής συνεχούς, φτηνής και αξιόπιστης ενέργειας έχει πρωταρχική οικονομική σημασία. Η πρόβλεψη του ηλεκτρικού φορτίου είναι ένα σημαντικό εργαλείο που χρησιμοποιείται για να εξασφαλίσει ότι η ενέργεια που παρέχεται από επιχειρήσεις κοινής ωφελείας ικανοποιεί το φορτίο συν την απώλεια ενέργειας στο σύστημα. Για το σκοπό αυτό, απαιτείται προσωπικό εξειδικευμένο προσωπικό για να ασκεί αυτή την εξειδικευμένη λειτουργία. Η πρόβλεψη φορτίου ορίζεται πάντα ως βασικά η επιστήμη ή η τέχνη της πρόβλεψης του μελλοντικού φορτίου σε ένα δεδομένο σύστημα για μια καθορισμένη χρονική περίοδο. Αυτές οι προβλέψεις μπορεί να είναι μόνο για ένα κλάσμα μιας ώρας μπροστά για επιχειρησιακούς σκοπούς, ή μέχρι 20 χρόνια στο μέλλον για σκοπούς προγραμματισμού

Σκοπός αυτής της διπλωματικής εργασίας είναι η πρόβλεψη της βραχυπρόθεσμης πρόβλεψης της ζήτησης ηλεκτρικού φορτίου σε βραχυπρόθεσμο ορίζοντα (έως για 1 μέρα) για το Ελληνικό Ηλεκτρικό Δίκτυο

Τα δεδομένα ηλεκτρικού φορτίου προέρχονται από την ΑΔΜΙΕ που συμαίνουν τα αρχικά της Ανεξάρτητος Διαχειριστής Ηλεκτρικής Ενέργειας. Μετεωρολογικά δεδομένα αντλήθηκαν από Dark Sky API και για να συγκρίνω τις προβλέψεις της διπλωματικής εργασίας, προβλέψεις από, ΛΑΓΙΕ που σημαίνει Λειτουργός Αγορά Ηλεκτρικής Ενέργειας που είναι υπεύθυνη για .

Τα σύνολα δεδομένων αντλήθηκαν από την προέλευσή τους, προεπεξεργάστηκαν, και χρησιμοποιήθηκαν έξι αλγόριθμοι μηχανικής μάθησης. Μετά από τέσσερις διαφορετικούς τύπους πειραμάτων χρησιμοποιήθηκε ο συνδυασμός μοντέλων SVM, XGBoost και Model Trees προκειμένου να ληφθεί σφάλμα πρόβλεψης 2.4% ενώ η ΛΑΓΙΕ δίνει 5,25% σφάλμα πρόβλεψης. Έτσι, τα μοντέλα της συγκεκριμένης διπλωματικής εργασίας βελτιώνουν την απόδοση πρόβλεψης κατά 54,9%

Ο κώδικα της διπλωματικής εργασίας μπορεί να βρεθεί στο github repo και τα αποτελέσματα - οπτικωποιήσεις των αναλύσεων βρίσκονται στο: IPTO ML

Πραξιτέλης - Νικόλαος Κουρουπέτρογλου
Σεπτέμβριος 2017

Keywords: Μηχανική Μάθηση, Μάθηση με επίβλεψη, Πρόβλεψη, Παλινδρόμηση

Acknowledgments

I would like to thank my parents for their support, and love all these years. I want to thank my supervisor Dr. Grigorios Tsoumakas, which guided throw the master thesis and aided me when all hope was lost. Moreover I want to thank other professors that I met during my postgraduate studies such as Dr. Ioannis Vlahavas, Dr Eleutherios Aggelis, Dr Athina Vakali, Dr Vaseiliadis Nikolaos, Dr Anastasios Gounaris, Dr Apostolos Papadopoulos, Dr Konstantinos Tsichlas. Their postgraduate courses helped me to explore and understand the world of ”Data Science” from many aspects.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter a brief introduction will be given about the subject of this master thesis and a summary of the following contents

1.1 Reasoning behind the thesis

This master thesis trying to solve the problem of electrical load demand forecasting with short-term horizon. An issue very important for the electrical companies who have to meet the needs of its customers. Knowing how much load will be demanded in the near future is beneficial in terms of administration, economic, marketing, and transmission of power of an electrical company.

1.2 Summary of the following contents

The Thesis is composed by the following chapters:

- Chapter 2: Electricity Load Demand Forecasting, trying to describe the issue, why it is important, commercial software on the Internet that deal with it
- Chapter 3: This chapter presents a variety of techniques, models, approaches that trying to predict the electrical load demand.
- chapter 4: This chapter presents a variety of statistical and data analysis software platforms and languages to implement the forecasting
- chapter 5: This chapter presents the implementation and predictions that developed in R, their accuracy performance and finally the visualizations in R-Shiny, R-Markdown
- Chapter 6: This chapter presents a summary for the whole master Thesis
- chapter 7: This chapter suggests future work and improvements in the current development.

Chapter 2

Electrical Load Demand Forecasting

2.1 What is the Problem and why is important

The economic progress in every society depends on the abundance of electric power. The abundance of sources for electrical production and the continuous existence of electric power is one of the at most factor of importance.

Nowadays, the electric load prediction is a vital process with applications that are used is various fields such as in an electric company producing electrical power. Many offices like the marketing department or the trade market inside such a company benefits from electric load prediction. The business reasons using electric load prediction are in summary the following:

1. For purchasing and producing electric power.
2. For transmitting, transferring and distributing electric power.
3. For managing and maintaining the electric power sources.
4. For managing the daily electric load demand.
5. For economic and marketing planning. [1]

We observe that the reasons for predicting the electric load demand are important and vital for an electric production company. The following table presents According to the lead time range of each business need described above, the minimum updating cycle and maximum horizon of the forecasts.

Table 2.1: Needs of forecasts in utilities/business needs

	Minimum updating cycle	Max horizon
purchasing and producing electric power	1 hour	10 years and above
transmitting, transferring and distributing electric power.	1 day	30 years
managing and maintaining the electric power sources.	15 minutes	2 weeks
managing the daily electric load demand	15 minutes	10 years and above
economic and marketing planning	1 month	10 years and above

As it is shown from the table above, the electric load prediction is an useful tool for an electric company that produce electric power because in this modern current world electric companies must provide adequate amount of electrical power for the needs of consumers. In general the electric load prediction is defined as the methodology for the load forecasting for a specific duration/horizon. The maximum horizon can be at most 20 years. [2]

There is no formal or typical procedure for electric load forecasting for every type of electric company and for every duration of time. The forecasting depends on various factors such as the means and sources of electrical production by each electrical company, the demand for electric load, climate factors, economical reasons and the human activity. [1]

Climate factors are those who are based on meteorological features like temperature, humidity, wind speed, precipitation etc. Based on the scientific literature temperature plays an important role for electric load consumption and many electric load forecasting uses temperature as a feature. However temperature can be used for small horizon for example for 1-2 days of forecast and it is an unreliable factor for more than that. [2]

The impact of human activity on the electric load consumption can be analyzed in various features. One of the features is the calendar features such as the day of the week or the month of the year. Another similar feature is the economical factors like the economic activities in urban areas or the economic transactions. For a long-term forecasting, e.g. forecasting for a year economic factors play a vital role for the future prediction. Moreover this kind of forecasting can use the human activities from rural areas such as the agricultural activities and the changes in rural land. [1]

The different types of electric load forecasting based on the data/features available that affect the forecasting and in conjunction with the duration of time for the prediction, can be categorized in the following categories:

1. very short term load forecasting (VSTLF), Can be used for small time-window, some hours at most and it utilizes the amount of previous hourly loads on that current day. It can not use information from economic factors or land because they do not alter ini this small amount of time.

2. short term load forecasting (STLF), This type of forecasting can be used for 24-hour forecasting until 2 weeks. They difference from the previous forecasting method is that here temperature and in general climate/meteorological factors play a vital role for the forecasting. STLF can be used for decision-taking like how much energy should an electric company purchase from other electric companies in the near future.

3. medium term load forecasting (MTLF), This forecasting method can be used for 1 month to 3 years horizon. Temperature and other climate factors do not be used for forecasting due to their unavailability of their own forecasting for this horizon. So other ways are used such as simulating their behavior and are being forecasting with this technique in order to be used of MTLF loaf forecasting. MTLF uses economical factors because their impact plays a role in daily life and for the needs of consumers. The same thing applies to the change of land in rural areas.

4. long term load forecasting (LTLF), This forecasting is used for forecasting for time-window 3-10 years. Simulation techniques are used to calculate climate factors and the economic transactions/activities.

In the previous 4 categories of load forecasting, the model that is used for predictions, it describes the relationship between the electric load and the features that affect it such as the climate factors, economical factors, land use etc.

However, for the load forecasting for long periods such as MTLF and LTLF, climate factors and economical features can not precisely determined for this long period of time.
The following table summarizes the importance of climate features, economical factors and land use for each type of load forecasting.

Table 2.2: The use of factors/features for load forecasting

factors/forecasting accuracy	Accurate	Inaccurate	Unreliable
Temperature	1 day	2 weeks	> 2 week
Economics	1 month	3 years	> 3 years
Land Use	1 year	5 years	> 5 years

In addition the following table categorizes the different types of forecasting based on the horizon for the prediction and the different kind of business needs for prediction:

Table 2.3: Classification of different types of electric load forecasting based on different business needs

	Temperature and climate factors	Economics	Land Use	Updating Cycle	Horizon
VSTLF	Optional	Optional	Optional	<= 1 Hour	1 day
STLF	Required	Optional	Optional	1 Day	2 weeks
MTLF	Simulated	Required	Optional	1 month	3 years
LTLF	Simulated	Simulated	Required	1 year	30 years

Based on the previous table we can relate every type of forecasting with the suitable features/factors needed to be applied. Furthermore, the different types of forecasting can be related with the different business need as mentioned above. The following table presents this relationship. Due to the fact that many business needs are being applied in various time periods/horizons, thus various of the previous forecasting techniques can be used for the same business need.

Table 2.4: Applications of the forecasts in Business Needs

	STLF	STLF	MTLF	LTLF
purchasing and producing electric power.	X	X	X	X
transmitting, transferring and distributing electric power.		X	X	X
managing and maintaining the electric power sources.	X	X		
managing the daily electric load demand.	X	X	X	X
financial and marketing planning.			X	X

2.2 Commercial Software

In this chapter commercial software will be presented used for electric load forecasting.

Load Forecasting Software

Company: Etap - Powering Success

Commercial Software: Etap - Load Forecasting Software

It is an ideal commercial software providing tools for short-term load forecasting.

Characteristics

- Adaptive Bus Load Forecasting
- Real-Time Trending
- Load Profile Library
- Forecasting Scenario Archiving
- Predict loading up to seven days ahead
- Forecast multiple load areas per individual meters
- User-adjustable weather variables and load profiles
- Revise forecasts based on loading and weather conditions
- Pattern and load profile libraries
- Import and export historical forecast data
- Unlimited forecast views

Commercial Software: Aiolos Forecasting Studio

Company: aiolosforecaststudio

Software: aiolosforecaststudio

The commercial software Aiolos Forecast Studio is a cloud based software, user-friendly and is used for forecasting. Aiolos Forecast Studio is developed by the Scandinavian company Vitec to manage forecasts such as electricity, gas hydro power. Electricity demand is forecasted with the highly accurate Aiolos model that handles +10K forecast series at the same time. Gas forecasting, Gas demand is forecasted with the highly accurate Cilia model that handles +10K forecast series at the same time. For

district heating/cooling plants the model Nestor is developed. A mixture of parameters from weather forecasts and the production plants makes this model essential for any production plant. For Hydro Power demand forecasting. The Aiolos software contains the model Achelous, which is adaptive and gives you precise forecasts on the water flow to hydro plants. For Wind power demand, Aiolos software can be connected to wind turbines and its model;h the Zephyros model provides production forecasts for entire coutries if needed.

Dealing the electric load demand forecast can be made with high accuracy, used historical past data, combined with adaptive models in conjunction with the input data, providing daily forecasts.

Commercial Software: Electric Load Forecasting Using Artificial Neural Networks

Company: gmdhshells

Software: Electric Load Forecasting Using Artificial Neural Networks

The commercial software Electric Load Forecasting Using Artificial Neural Networks by GMDH Shelluses neural networks for reliable and accurate forecasts for electric load power demand. Is uses past historical data provided by the user. The way that it works is through minimizing the error of prediction vs the real values. When the error criterion is satisfied then the final model is given to the user.

This particular software is user-friendly and works only in Windows systems. It provides descriptive statistics tools. It uses calendar features (Weekends, Holidays, National Days etc). It can utilize ODBC Data Base connections in order to retrieve information from Data Bases. The horizon of prediction is from short-term to long-term forecasting. Moreover is has the ability to decompose time series data in its components (trend, seasonality, cyclic).

Commercial Software: Escoware demand-forecasting

Company: escoware

Software: Demand Forecasting Solution

The Software ESCOWare® DFS solution uses advanced and modern machine learning techniques for accurate electric load forecast. The ESCOWare® DFS solution utilizes proprietary, state of the art weather normalized algorithms that help to mitigate risk by accurately forecasting customer usage, allowing for predictable margins and profitability. ESCOWare® DFS is a cloud-based software solution designed specifically for the complexities of retail energy suppliers, and utilities.

Commercial Software: SAS® Energy Forecasting

Company: SAS

Commercial Software: SAS® Energy Forecasting

The commercial software SAS Energy Forecasting provides reliable electric load forecasting predictions. It has the utilities to predict from short-term to long-term horizon. It is designed in order to predict the electric load forecasting providing reliable and accurate results regardless of the size of the input meteorological data that are given to be used as independent variables for prediction.

Commercial Software: Statgraphics

Commercial Software: statgraphics

Statgraphics is a commercial statistical software for statistical analysis. It is developed in 1980 from Dr. Neil Polhemus, professor in statistics from Princeton University. The current edition is called Statgraphics Centurion XVII is realised in 2014.

In terms of data analysis for electricity consumption, their structure is in the form of time series. Statgraphics has a plethora of time series visualization tools. It has controls for finding random time series, analyzing their oscillations, and finding any voltage, seasonality, periodicity, etc. In addition, it has mechanisms for descriptive data statistics, and more specifically for time series consumption data, it has autocorrelation functions to interpret how past data affect future ones. Also other tools such as the periodogram are useful for analyzing time series oscillations and whether they maintain specific frequencies at regular intervals. It also has linear and non-linear smoothing techniques for time series and especially those that hold enough noise to reliably find the time series trend. It also has time series degradation techniques in their basic components to make it possible to find seasonal patterns and produce data adapted to the seasonality-periodicity component. Finally, Statgraphics has tools for predicting time series, exploiting the past time series data to be predicted by various processes such as random walk, mobile media, trend models, simple, linear, polynomial models, exponential smoothing. ARIMA parametric models. Statgraphics also has the ability to suggest that the user selects the model itself, which minimizes the prediction error of the time series.

Chapter 3

Electricity Load Demand Forecasting Techniques

In this chapter various techniques from statistics to machine learning will be presented for electricity load forecasting.

3.1 Statistical Approaches

- Regression Analysis:
  The short-term electricity load demand forecasting with regression analysis was analyzed a lot from the scientific literature. For example the use of regression analysis for the Pacific Gas and Electric Company for the 24-hour electricity load demand was used. [3]. The fundamental techniques that was used in this approach was the least square method and the method of weighted least squares to minimize the consequences from the extreme values. The use of temperature, calendar variables such as holidays, weekends etc. was also used as independent variables. Subsequent research into the energy consumption forecast for the same company as an extension of the original survey was made with the purpose of using meteorological data such as humidity [4]

- Time Series Approach:
  Time series and ARIMA models for short-term STLF forecasts methods are commonly found in the scientific literature. [5] ARIMA models are used with normalized data of electricity demand and temperature before the prediction is produced. ARIMA models along with the Box N 'Jenkins time series were used for STLF in research published in IEEE [6]. In this study, the non-linear relationship of temperature to the demand for electrical energy has emerged, and the third degree of polynomial function has been used for multiple linear regression. Also in a different survey for a short-term forecast of electricity consumption published in the IEEE, the use of multiple linear regression with ARIMA was used, normalization of data in combination with the use of human behavior in calendar taking into account holidays, weekends etc. [7]

3.2 Artificial Intelligence - Machine Learning Approaches

- Neural Networks:
  Historically, since 1990, there have been published surveys that use artificial neural networks in short-term electricity consumption forecasts originally published in IEEE. An initial survey is the following: [8]. In this study, an algorithm of artificial neural networks was proposed in conjunction with time series and regression analysis to address the non-linear relationship of demand for electricity consumption with various meteorological data and was applied to the data of Puget Sound Power and Light Company's from November 1988 to January 1989. Artificial neural networks were also used in the above-mentioned survey of Pacific Gas and Electric Company [3] and

compared the model of neural networks with that developed by regression analysis in the same set of training data from 1986 to 1990. It was shown that artificial neural networks exhibit greater prediction accuracy than regression analysis. The neural network model used was the BPN (Back Propagation model). An extension of the previous research was based on the use of the radial basis function neural network (RBFN) [9], used for the same company and the same set of data. The new algorithm showed better performance and accuracy than the BPN model of neural networks and the use of calendar variables for holidays, holidays, etc. For Greece, PPC, there have been remarkable surveys using clusters of neural networks for short-term forecasting of electricity consumption from the research by Adamakos Apostolos [10], and other similar investigations using artificial intelligence techniques [11–13]

- Fuzzy Logic - Fuzzy Neural Networks:
  Since the late 1980s and early 1990s, many researchers have designed intelligent systems to predict power consumption using fuzzy logic, such as in 1988 the publication of the study of the prediction of electricity consumption for the Old Dominion Electric Cooperative Virginia [14, 15]. Another similar research was developed in 1990 for the Taiwan power system [16,17]. And especially after 1990, many studies have begun to design fuzzy logic based systems (automatic fuzzy inference systems) [18–20]. Finally, since 1990 there has been a boom in the use of neural networks of fuzzy logic.

- Deep Learning:
  Recent scientific / research publications have been made on the prediction of electricity consumption using modern methods of machine learning and artificial intelligence, the techniques of deep learning. For example, in the research by Xueheng Qiu, Le Zhang, Ye Ren and Suganthan of the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, in collaboration with researcher Gehan Amaratunga, Department of Engineering, University of Cambridge, UK, DBN) to predict electricity consumption compared to time series regression and analysis techniques, also presenting the benefits of the neural learning networks in depth compared to traditional statistical methods and methods time analysis using reference methods. [24]. In another recently published 2016 research by Seunghyoung Ryu, Jaekoo Noh and Hongseok Kim from the Electronic Engineering Department at Sogang University in Seoul, South Korea, used the latest technology of artificial intelligence in deep learning and implemented a deep model neural network (DNN) - based for short-term power consumption. They used to train the model pre-trained restricted Boltzmann machine. Meteorological features were also used within the prediction model and performance comparisons with other prediction models, the results of the shallow neural network (SNN) algorithm, a double seasonal Holt-Winters (DSHW) model, and an autoregressive integrated moving average ARIMA) model. The metric was compared with MAPE, and DNN showed the fewest and lowest predictive errors compared to the rest of the models. [25].

- SVM:
  Support Vector Machines are widely used as models for predicting electricity consumption. For example, in the published IEEE article in 2001, the European Intelligent Technology Organization (EUNITE) organized a competition for medium-term forecasting of electricity consumption with the best model being SVM-based [21]. SVM for MTLF was used to predict power consumption in Istanbul using average temperature data per day, calendar factors and daytime electricity demand records [22]. In addition to SVM models, Artificial Neural Networks models based on Wavelet-based Networks (EWNs) [23] as well as hybrid techniques.

- Random Forests:
  Browsing the scientific literature various papers can be found for random Forest approaches. A Random Forest combined with A feature selection method based on the generalized minimum redundancy and maximum relevance is proposed to improve the accuracy of short-term load forecasting (STLF) by the School of Electrical Engineering, Northeast Dianli University from China [26].

Another paper using Random Forests is the following study that proposes using Random Forest models from Czestochowa University of Technology, Poland, for short-term electricity load forecasting. This study uses an ensemble learning method that generates many regression trees and aggregates their results. The model operates on patterns of the time series seasonal cycles which simplify the forecasting problem especially when a time series exhibits nonstationarity, heteroscedasticity, trend and multiple seasonal cycles. The main advantages of the model are its ability to generalization, built-in cross-validation and low sensitivity to parameter values. As an illustration, the proposed forecasting model is applied to historical load data in Poland and its performance is compared with some alternative models such as CART, ARIMA, exponential smoothing and neural networks. Application examples confirm good properties of the model and its high accuracy [27].

- Rule-Based Model Trees:
  Rule-Based Model Trees are also being applied to the field of Short-term load forecasting and the scientific literature verifies it. For example from the University of Economics Prague & University of Finance and Administration Prague, Czech Republic, a study was made to investigate the possibility to apply symbolic data mining methods to the problem of prediction. By employing Model Trees that is used for extraction of classification or prediction rules from data. When new data is coming, the active rules (rules with a fulfilled left-hand side) from the rule base are applied to the data and their weights are composed by the inference mechanism to the resulting weight of a given prediction. The presented approach is applied to the problem of short-term electric load forecasting [28]. Another Rule based Model Trees approach is discussed from the next survey from IEEE [29]. In that survey the formulation of rules for the rule base and the application of such rules was discussed. The classification of the load forecast parameters into weather-sensitive and nonweather-sensitive categories was described. The rationale underlying the development of rules for both the one-day and seven-day forecast is presented. This exercise leads to the identification and estimation of parameters relating load, weather variables, day types, and seasons. Sample rules that are the product of identifiable statistical relationships and expert knowledge are examined. A self-learning process is described which shows how rules governing the electric utility load can be updated [29].

- K-Nearest Neighbors:
  K-Nearest Neighbors models are also can be found in the scientific literature. For example a published paper from the school of mathematics and statistics, Lanzhou University in China they used KNN algorithms to construct a hybrid model, Wavelet Denoising-Extreme Learning Machine and k-Nearest Neighbor Regression (EWKM), which combines k-Nearest Neighbor (KNN) and Extreme Learning Machine (ELM) based on a wavelet denoising technique was proposed for short-term load forecasting. The proposed hybrid model decomposed the time series into a low frequency-associated main signal and some detailed signals associated with high frequencies at first, then used KNN to determine the independent and dependent variables from the low-frequency signal. The model ELM was used to get the non-linear relationship between these variables to get the final prediction result for the electric load. Compared with three other models, Extreme Learning Machine optimized by k-Nearest Neighbor Regression (EKM), Wavelet Denoising-Extreme Learning Machine (WKM) and Wavelet Denoising-Back Propagation Neural Network optimized by k-Nearest Neighbor Regression (WNNM), the model proposed in that paper can improve the accuracy efficiently [30]. In another paper the knn algorithms was used in order paper proposes a k-nearest neighbor (kNN)-based model for predicting the next-day's load with only limited temperature forecasts, namely minimum and maximum temperature of a day, as the forecasting input. The proposed model consists of three individual kNN models which have different neighboring rules. The three are combined together by tuned weighting factors for a final forecasting output. The proposed model is tested on the Australian National Electricity Market (NEM) data, showing reasonably high accuracy. It can be used as an alternative tool for day-ahead load forecasting when

only limited temperature information is available [31].

- XGBoost Models:
  Furthermore XGBoost models can be found in the scientific literature to be used in electricity load forecasting. For example in the following study, the use of XGBoost models was used in order to be built a short-term power load extreme gradient boosting (XGBoost) model with multi-information fusion by using the gradient boosting algorithm. [32]. In another study XGBoost models was used in order to compete in Kaggle competition in 2012. The competition involved a hierarchical load forecasting problem for a US utility with 20 geographical zones. The available data consisted of the hourly loads for the 20 zones and hourly temperatures from 11 weather stations, for four and a half years. For each zone, the hourly electricity load for nine different weeks needed to be predicted without having the location of zones or stations. They used separate models for each hourly period, with component-wise gradient boosting to estimate each model using univariate penalized regression splines as base learners. The models allow for the electricity demand to change with time-of-year, day-of-week, time-of-day, and on public holidays, with the main predictors being current and past temperatures as well as past demand. The team ranked 5th among 105 teams [33].

3.3 Theoretical Background for Various Models

For this project, the theoretical background for six from the previous models - approaches will be described. These was used for the analysis of the master thesis for STLF in the Greek Network. These are: svm, neural networks, K-Nearest Neighbors, Random Forest, Rule - Based and XGBoost.

3.3.1 SVM Theoretical Background

Support Vector Machines are very specific class of algorithms, characterized by usage of kernels, absence of local minima, sparseness of the solution and capacity control obtained by acting on the margin, or on number of support vectors, etc. They were invented by Vladimir Vapnik and his co-workers, and first introduced at the Computational Learning Theory (COLT) 1992 conference with the paper. All these nice features however were already present in machine learning since 1960's: large margin hyper planes usage of kernels, geometrical interpretation of kernels as inner products in a feature space. Similar optimization techniques were used in pattern recognition and sparsness techniques were widely discussed. Usage of slack variables to overcome noise in the data and non - separability was also introduced in 1960s. However it was not until 1992 that all these features were put together to form the maximal margin classifier, the basic Support Vector Machine, and not until 1995 that the soft margin version was introduced.

Support Vector Machine can be applied not only to classification problems but also to the case of regression. Still it contains all the main features that characterize maximum margin algorithm: a non-linear function is leaned by linear learning machine mapping into high dimensional kernel induced feature space. The capacity of the system is controlled by parameters that do not depend on the dimensionality of feature space.

In the same way as with classification approach there is motivation to seek and optimize the generalization bounds given for regression. They relied on defining the loss function that ignores errors, which are situated within the certain distance of the true value. This type of function is often called – epsilon intensive – loss function. The figure below shows an example of one-dimensional linear regression function with – epsilon intensive – band. The variables measure the cost of the errors on the training points. These are zero for all points that are inside the band.

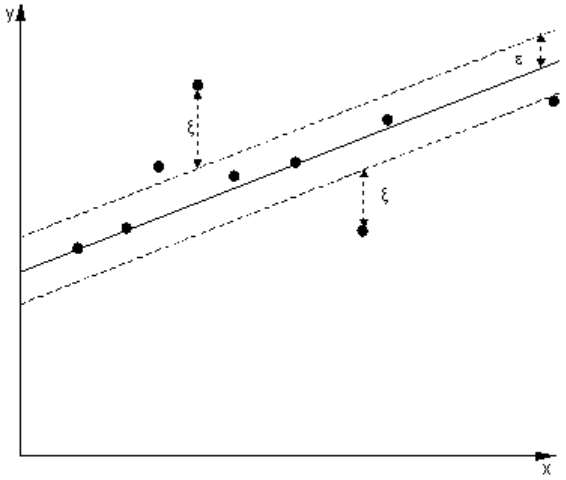

Figure 3.1: One-dimensional linear regression with epsilon intensive band.

Another picture shows a similar situation but for non-linear regression case.

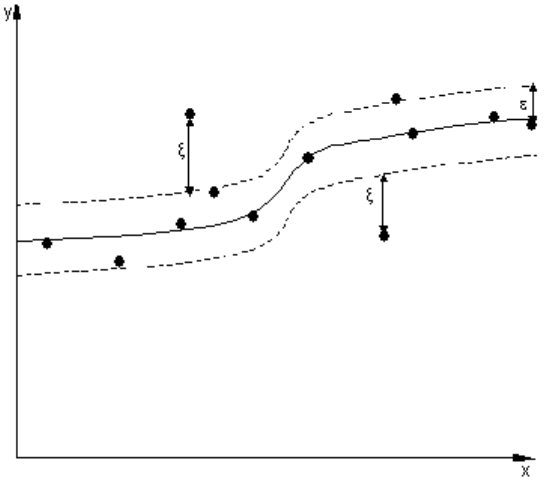

Figure 3.2: Non-linear regression function

One of the most important ideas in Support Vector Classification and Regression cases is that presenting the solution by means of small subset of training points gives enormous computational advantages. Using the epsilon intensive loss function we ensure existence of the global minimum and at the same time optimization of reliable generalization bound. In SVM regression, the input is first mapped onto a m-dimensional feature space using some fixed (nonlinear) mapping, and then a linear model is constructed in this feature space. Using mathematical notation, the linear model (in the feature space) $f(x, w)$ is given by: $f(x, w) = \sum_{j=1}^{m} w_i g_i(x) + b$ where $g_j(x), j = 1, ..., m$ denotes a set of nonlinear transformations, and b is the "bias" term. Often the data are assumed to be zero mean (this can be achieved by preprocessing), so the bias term is dropped.

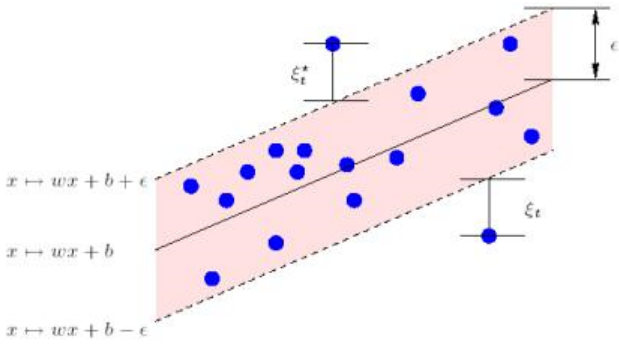

Figure 3.3: Non-linear mapping of input examples into high dimensional feature space. (Classification case, however the same stands for regression as well).

The quality of estimation is measured by the loss function $L(y, f(x, w))$. SVM regression uses a new type of loss function called -insensitive loss function proposed by Vapnik:

$$L(y, f(x, w)) = \begin{cases} 0 & if|y - f(x, w)| \leq \epsilon \\ |y - f(x, w)| - \epsilon & otherwise \end{cases}$$

The empirical risk is:

$$R_{(emp)}(w) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i, w))$$

SVM regression performs linear regression in the high-dimension feature space using ε-insensitive loss and, at the same time, tries to reduce model complexity by minimizing $||w||^2$. This can be described by introducing (non-negative) slack variables , to measure the deviation of training samples outside -insensitive zone. Thus SVM regression is formulated as minimization of the following functional:

$$min\frac{1}{2}||w^2|| + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \begin{cases} y_i - f(x_i, w) \leq \epsilon + \xi_i^* \\ f(x_i, w) - y_i \leq \epsilon + \xi_i \\ \xi_i, \xi_i^* \leq 0, i = 1, ..., n \end{cases} , s.t.$$

This optimization problem can transformed into the dual problem and its solution is given by:

$$f(x) = \sum_{i=1}^{n_{SV}} (a_i - a_i^*) K(x_i, x), s.t. 0 \leq a_i^* \leq C, 0 \leq a_i \leq C,$$

where n_{SV} is the number of Support Vectors (SVs) and the kernel function:

$$K(x, x_i) = \sum_{j=1}^{m} g_j(x) g_j(x_i)$$

It is well known that SVM generalization performance (estimation accuracy) depends on a good setting of meta-parameters parameters C, and the kernel parameters. The problem of optimal parameter selection is further complicated by the fact that SVM model complexity (and hence its generalization performance) depends on all three parameters. Existing software implementations of SVM regression usually treat SVM meta-parameters as user-defined inputs. Selecting a particular kernel type and kernel function parameters is usually based on application-domain knowledge and also should reflect distribution of input (x) values of the training data.

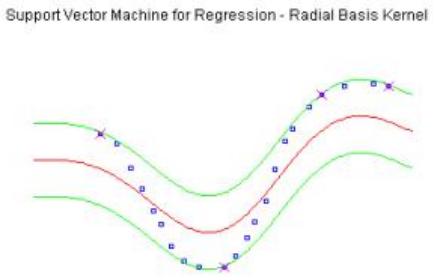

Figure 3.4: Performance of Support Vector Machine in regression case. The epsilon boundaries are given with the green lines. Blue points represent data instances.

Parameter C determines the trade off between the model complexity (flatness) and the degree to which deviations larger than are tolerated in optimization formulation for example, if C is too large (infinity), then the objective is to minimize the empirical risk only, without regard to model complexity part in the optimization formulation.

Parameter controls the width of the -insensitive zone, used to fit the training data. The value of can affect the number of support vectors used to construct the regression function. The bigger , the fewer support vectors are selected. On the other hand, bigger -values results in more 'flat' estimates. Hence, both C and -values affect model complexity (but in a different way).

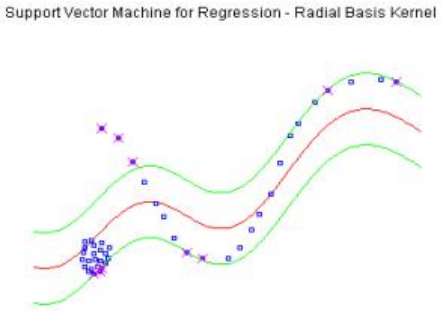

Figure 3.5: Performance of Support Vector Machine in regression case. The epsilon boundaries are given with the green lines. Blue points represent data instances.

(Additional instances are introduced in this case and after being supplied to the model are inside epsilon band. The regression function has changed. However this makes some points that were initially inside the interval to be misclassified.)

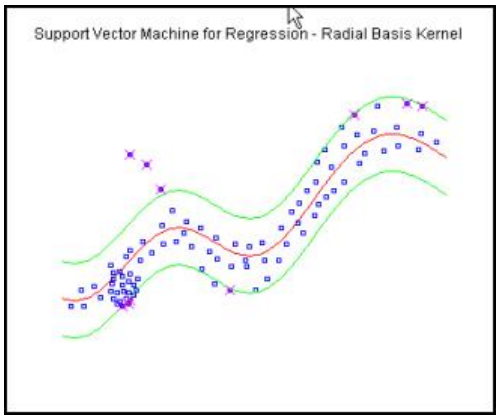

Figure 3.6: Performance of Support Vector Machine in regression case. The epsilon boundaries are given with the green lines. Blue points represent data instances.

One of the advantages of Support Vector Machine, and Support Vector Regression as the part of it, is that it can be used to avoid difficulties of using linear functions in the high dimensional feature space and optimization problem is transformed into dual convex quadratic programmes. In regression case the loss function is used to penalize errors that are grater than threshold - . Such loss functions usually lead to the sparse representation of the decision rule, giving significant algorithmic and representational advantages.

3.3.2 Random Forest Theoretical Background

Random forests combine the predictions of multiple decision trees. The datasets are repeatedly divided into subtrees, guided by the best combination of variables. However, finding the right combination of variables can be difficult. For instance, a decision tree constructed based on a small sample might be not be generalizable to future, large samples. To overcome this, multiple decision trees could be constructed, by randomizing the combination and order of variables used. The aggregated result from these forest of trees would form an ensemble, known as a random forest.

Random forest predictions are often better than that from individual decision trees. The chart below compares the accuracy of a random forest to that of its 1000 constituent decision trees. Only 12 out of 1000 individual trees yielded an accuracy better than the random forest. In other words, there is a 99% certainty that predictions from a random forest would be better than that from an individual decision tree.

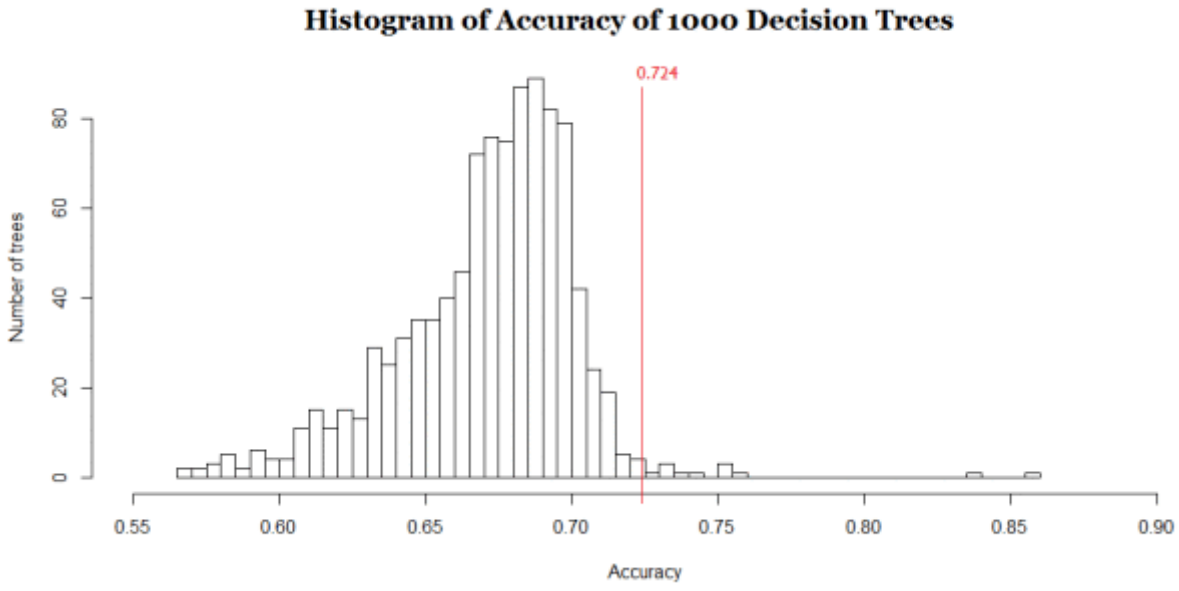

Figure 3.7: Histogram showing the accuracy of 1000 decision trees. While the average accuracy of decision trees is 67.1%, the random forest model has an accuracy of 72.4%, which is better than 99% of the decision trees.

Random forests are widely used because they are easy to implement and fast to compute. Unlike most other models, a random forest can be made more complex (by increasing the number of trees) to improve prediction accuracy without the risk of over-fitting.

A random forest is an example of an ensemble, which is a combination of predictions from different models. In an ensemble, predictions could be combined either by majority-voting or by taking averages. Below is an illustration of how an ensemble formed by majority-voting yields more accurate predictions than the individual models it is based on:

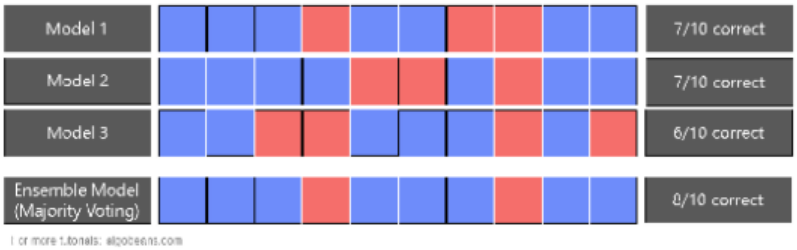

Figure 3.8: Example of three individual models attempting to predict 10 outputs of either Blue or Red. The correct predictions are Blue for all 10 outputs. An ensemble formed by majority voting based on the three individual models yields the highest prediction accuracy.

As a random forest is an ensemble of multiple decision trees, it leverages "wisdom of the crowd", and is often more accurate than any individual decision tree. This is because each individual model has its own strengths and weakness in predicting certain outputs. As there is only one correct prediction but many possible wrong predictions, individual models that yield correct predictions tend to reinforce each other, while wrong predictions cancel each other out. For this effect to work however, models included in the ensemble must not make the same kind of mistakes. In other words, the models must be uncorrelated. This is achieved via a technique called bootstrap aggregating (bagging).

In random forest, bagging is used to create thousands of decision trees with minimal correlation. In bagging, a random subset of the training data is selected to train each tree. Furthermore, the model randomly restricts the variables which may be used at the splits of each tree. Hence, the trees grown are dissimilar, but they still retain certain predictive power.

Random forests are considered "black-boxes", because they comprise randomly generated decision trees, and are not guided by explicitly guidelines in predictions. We do not know how exactly the model came to the conclusion that a violent crime would occur at a specific location, instead we only know that a majority of the 1000 decision trees thought so. This may bring about ethical concerns when used in areas like medical diagnosis.

Random forests are also unable to extrapolate predictions for cases that have not been previously encountered. For example, given that a pen costs $2, 2 pens cost $4, and 3 pens cost $ 6, how much would 10 pens cost? A random forest would not know the answer if it had not encountered a situation with 10 pens, but a linear regression model would be able to extrapolate a trend and deduce the answer of $20.

3.3.3 K-Nearest Neighbors Theoretical Background

K nearest neighbors is a simple algorithm that stores all available cases and predict the numerical target based on a similarity measure (e.g., distance functions). KNN has been used in statistical estimation and pattern recognition already in the beginning of 1970's as a non-parametric technique.

A simple implementation of KNN regression is to calculate the average of the numerical target of the K nearest neighbors. Another approach uses an inverse distance weighted average of the K nearest neighbors. KNN regression uses the same distance functions as KNN classification.

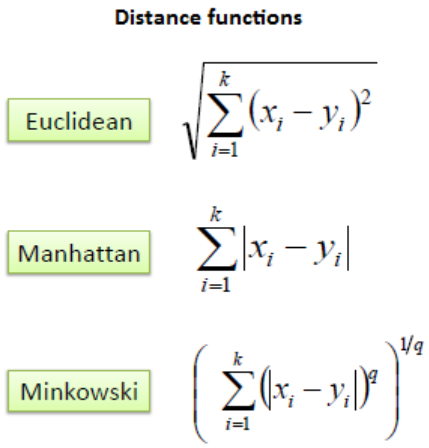

Figure 3.9: KNN distance functions

Choosing the optimal value for K is best done by first inspecting the data. In general, a large K value is more precise as it reduces the overall noise; however, the compromise is that the distinct boundaries within the feature space are blurred. Cross-validation is another way to retrospectively determine a good K value by using an independent data set to validate your K value. The optimal K for most datasets is 10 or more. That produces much better results than 1-NN.

3.3.4 Neural Networks Theoretical Background

Artificial neural networks (ANNs) were originally devised in the mid-20th century as a computational model of the human brain. Their used waned because of the limited computational power available at the time, and some theoretical issues that weren't solved for several decades (which I will detail at the end of this post). However, they have experienced a resurgence with the recent interest and hype surrounding Deep Learning. One of the more famous examples of Deep Learning.

It is theorized that because of their biological inspiration, ANN-based learners will be able to emulate how a human learns to recognize concepts or objects without the time-consuming feature engineering step. Whether or not this is true (or even provides an advantage in terms of development time) remains to be seen, but currently it's important that we machine learning researchers and enthusiasts have a familiarity with the basic concepts of neural networks.

Neural network terminology is inspired by the biological operations of specialized cells called neurons. A neuron is a cell that has several inputs that can be activated by some outside process. Depending on the amount of activation, the neuron produces its own activity and sends this along its outputs. In addition, specific input or output paths may be ”strengthened” or weighted higher than other paths. The hypothesis is that since the human brain is nothing but a network of neurons, we can emulate the brain by modeling a neuron and connecting them via a weighted graph.

The artificial equivalent of a neuron is a node (also sometimes called neurons, but I will refer to them as nodes to avoid ambiguity) that receives a set of weighted inputs, processes their sum with its activation function □□, and passes the result of the activation function to nodes further down the graph. Note that it is simpler to represent the input to our activation function as a dot product:

$$\phi\left(\sum_i w_i a_i\right) = \phi(\mathbf{w}^T\mathbf{a})$$

Figure 3.10: activation function as a dot product

Visually this looks like the following:

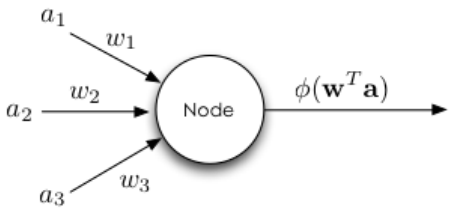

Figure 3.11: 1-node neural network

There are several canonical activation functions. For instance, the sigmoid activation function:

$$\phi(\mathbf{w}^T\mathbf{a}) = \frac{1}{1+\exp(-\mathbf{w}^T\mathbf{a})}$$

Figure 3.12: sigmoid activation function

We can then form a network by chaining these nodes together. Usually this is done in layers - one node layer's outputs are connected to the next layer's inputs (we must take care not to introduce cycles in our network, for reasons that will become clear in the section on backpropagation).

Our goal is to train a network using labelled data so that we can then feed it a set of inputs and it produces the appropriate outputs for unlabeled data. We can do this because we have both the input x_i and the desired target output y_i in the form of data pairs. Training in this case involves learning the correct edge weights to produce the target output given the input. The network and its trained weights form a function (denoted hh) that operates on input data. With the trained network, we can make predictions given any unlabeled test input.

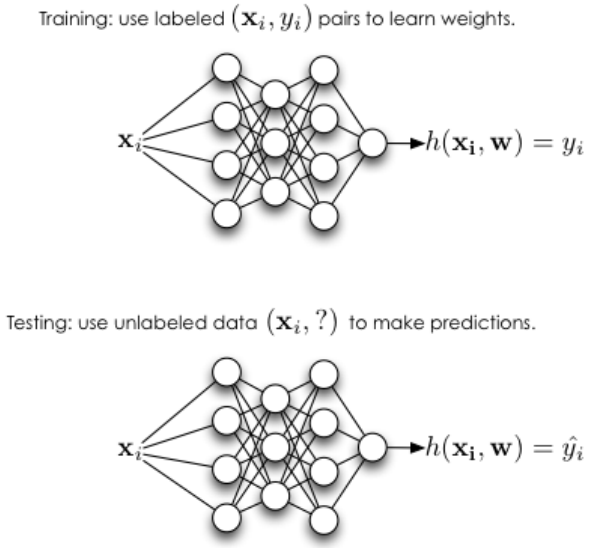

Figure 3.13: KNN distance functions

3.3.5 XGBoost Theoretical Background

XGBoost is short for "Extreme Gradient Boosting", where the term "Gradient Boosting" is proposed in the paper Greedy Function Approximation: A Gradient Boosting Machine, by Friedman. XGBoost is based on this original model.

The GBM (boosted trees) has been around for really a while, and there are a lot of materials on them. It is an implementation of gradient boosting machines created by Tianqi Chen, now with contributions from many developers. It belongs to a broader collection of tools under the umbrella of the Distributed Machine Learning Community or DMLC who are also the creators of the popular mxnet deep learning library.

XGBoost is focused on computational speed and model performance. The implementation of the model supports the features of the scikit-learn and R implementations, with new additions like regularization. Three main forms of gradient boosting are supported:

- Gradient Boosting algorithm also called gradient boosting machine including the learning rate.
- Stochastic Gradient Boosting with sub-sampling at the row, column and column per split levels.
- Regularized Gradient Boosting with both L1 and L2 regularization.

XGBoost algorithm was engineered for efficiency of compute time and memory resources. A design goal was to make the best use of available resources to train the model. Some key algorithm implementation features include:

- Sparse Aware implementation with automatic handling of missing data values.
- Block Structure to support the parallelization of tree construction.
- Continued Training so that you can further boost an already fitted model on new data.

XGBoost dominates structured or tabular datasets on classification and regression predictive modeling problems. The evidence is that it is the go-to algorithm for competition winners on the Kaggle competitive data science platform.

The XGBoost library implements the gradient boosting decision tree algorithm. This algorithm goes by lots of different names such as gradient boosting, multiple additive regression trees, stochastic gradient boosting or gradient boosting machines.

Boosting is an ensemble technique where new models are added to correct the errors made by existing models. Models are added sequentially until no further improvements can be made. A popular example is the AdaBoost algorithm that weights data points that are hard to predict.

Gradient boosting is an approach where new models are created that predict the residuals or errors of prior models and then added together to make the final prediction. It is called gradient boosting because it uses a gradient descent algorithm to minimize the loss when adding new models. This approach supports both regression and classification predictive modeling problems.

3.3.6 Rule-Based Model Trees Theoretical Background

The model tree approach described in Quinlan (1992) called M5, which is similar to regression trees except:

- the splitting criterion is di□erent
- the terminal nodes predict the outcome using a linear model (as opposed to the simple average)
- when a sample is predicted, it is often a combination of the predictions from di□erent models along the same path through the tree.

The main implementation of this technique is a “rational reconstruction” of this model called M5’, which is described by Wang and Witten (1997) and is included in the Weka software package.

When model trees make a split of the data, they fit a linear model to the current subset using all the predictors involved in the splits along the path. This process proceeds until there are not enough samples to split and/or fit the model. A pruning stage is later used to simplify the model.

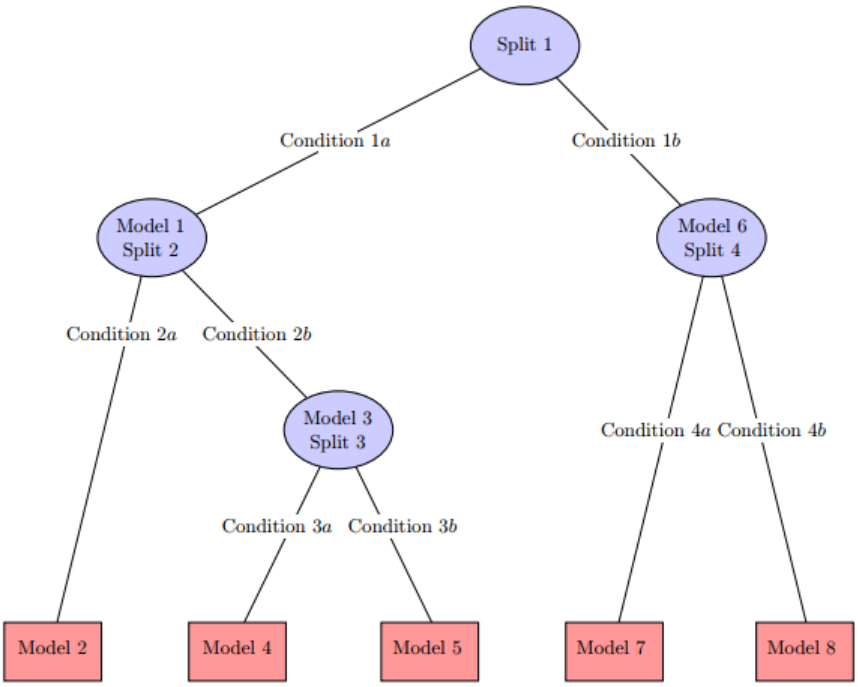

Figure 3.14: Model Trees

Tree–based models consist of one or more nested if-then statements for the predictors that partition the data. Within these partitions, a model is used to predict the outcome. For example, a very simple tree could be defined as:

```
if >= 1.7 then
|   if X2 >= 202.1 then Y = 1.3
|   else Y = 5.6
else Y = 2.5
```

Figure 3.15: Regression Trees rules

Notice that the if-then statements generated by a tree define a unique route to one terminal node for any sample. A rule is a set of if-then conditions (possibly created by a tree) that have been collapsed into independent conditions. For the example above, there would be three rules:

```
if X1 >= 1.7 & X2 >= 202.1 then Y = 1.3
if X1 >= 1.7 & X2 <  202.1 then Y = 5.6
if X1 <  1.7               then Y = 2.5
```

Figure 3.16: From Regression Trees to Rules

The next step here is to substitute the results from the leaves with linear regression models, creating thus Rule - Based Model Trees where each terminal node has rules that drive us to use a specific linear regression formula This is very useful because we have predictive models with high accuracy, stability and ease of interpretation that they map non-linear relationships quite well. They are adaptable at solving any kind of problem at hand. For example the following figure shows the result from the library Cubist, it creates 2 rules and each rule ends with a linear regression function:

```
Rule 1: [58 cases, mean 4.980517, range 4.477121 to 5.523746, est err 0.152796]

if zip in {z95621, z95626, z95660, z95673, z95683, z9581, z95817, z95820,
           z95822, z95823, z95824, z95826, z95827, z95828, z95832, z95838
           z95841, z95842, z95843} and
   beds <= 2 then
outcome = 7.944631 + 0.323 beds + 4e-05 sqft + 0.03 longitude

Rule 2: [126 cases, mean 5.200466, range 4.788875 to 5.662758, est err 0.090147]

if zip in {z95626, z95660, z95683, z95815, z95823, z95824, z95827, z95832,
           z95838, z95841} and
   beds > 2 then
outcome = 8.524561 - 0.056 beds + 0.000342 sqft + 0.03 longitude + 0.003 baths
```

Figure 3.17: Rule-Based Regression Model Tree

3.4 Summary of Meteorological Forecasting Factors

It is well known that electricity load demand is directly affected by the weather. In weather changes, heating and air conditioning appliances are used, resulting in increased demand for electricity. Some forecasting techniques for short-time STLF electricity consumption do not require the use of meteorological data, but as a matter of fact, most use them. From published articles to well-known scientific organizations, they use data from temperature, humidity, wind speed, cloudiness, dew point, wind direction, temperature-humidity index (THI) wind chill index (WCI). From all of the aforementioned data, what you use is the temperature. The way you use it, you can use it on an hourly basis, the previous hour, the difference with the previous hour and the current, the minimum and the maximum, or even the average value of it over a few hours etc.

3.5 Summary of Calendar Forecasting Factors

There are many calendar factors that affect the short-term load forecasting. Months are one of them. Months can be categorized from 4 seasons, to multiple groups of seasons, for example into 7 types to distinguish the transitions between two adjacent seasons: winter (Dec 1 – Feb 15), late winter – early spring (Feb 16 – 15 March), Spring (16 march – May 31), late spring – early summer (Jun 1 – Jun 30), Summer (Jul 1 – Sep 15), late summer – early fall (Sep 16 – 28 Oct), and Fall, (29 Oct – Nov 30) [35].

The energy consumption behavior in different days of a week may be different, which is due to many reasons. For instance, office buildings may be closed during weekend, which causes fewer loads than those of work days. People may get up late in the morning during weekend, which shifts the morning peak one or two hours later than a normal work day so seven groups of days of a week can be created. In daily basis, hours can be grouped from rush hours and non-rush hours.

Forecasting the loads of special days, e.g., holidays (national and religious), has been a challenging issue in STLF, not only because the load profiles may vary from different holidays and the same holiday of different years, but also due to the limited data history.

3.6 Evaluating forecast accuracy

this section will describe some metrics to evaluate prediction accuracy as a benchmark for the performance of each model being developed. We assume that Y_i indicates the i_{th} observation by the variable to predict and $\widehat{Y_i}$ the prediction of the variable Y_i.

The forecast error is simply $e_i = Y_i - \widehat{Y}_i$, which is on the same scale as the data. Accuracy measures that are based on e_i are therefore scale-dependent and cannot be used to make comparisons between series that are on different scales.

The two most commonly used scale-dependent measures are based on the absolute errors or squared errors:

- Mean absolute error: MAE = $mean(|e_i|)$
- Root mean squared error: RMSE = $\sqrt{mean(e_i^2)}$

When comparing forecast methods on a single data set, the MAE is popular as it is easy to understand and compute.

The percentage error is given by $p_i = 100e_i/y_i$. Percentage errors have the advantage of being scale-independent, and so are frequently used to compare forecast performance between different data sets. The most commonly used measure is:

- Mean absolute percentage error: MAPE = $mean(|p_i|)$

Chapter 4

Data Analysis and Machine Learning Platforms

4.1 Programming Languages and Libraries

1. R

Figure 4.1: R Data science software

R is an open source high-level programming language and a computing statistics and graphics development environment supported by the R Foundation for Statistical Computing. R language is widely used by statisticians, data miners, data analysts and its popularity has increased over the years. R is a GNU project, its source code is mainly written in C, Fortran. H R is available for free under the GNU General Public License. It has a command line environment as well as a variety of graphical environments (R Studio). R's predecessor was S, with lexical scoping semantics, the second being implemented by Bell Labs. Initially, R was created by Ross Ihaka and Robert Gentleman of the University of Auckland, New Zealand, and is currently its main development team. Named by the original letter of the names of its creators. Its development began in 1992 with an initial release in 1995 and the stable version appeared in 2000. R and its libraries implement a wide range of statistical and graphical functions, including linear and nonlinear models, classical statistical controls, time series analysis, engineering classroom classification models, clustering, etc. It can easily be expanded because it is an open source project and her community is particularly active in contributing to improving her libraries. [36]

2. Python
   Python is a widely known high-level language, created by Guido van Rossum and released in 1991. The concept is to simplify and legibility of the code by using only the blank character/s to avoid code track block using brackets as implemented with other programming languages. This language has also adopted a special way of writing to allow developers to write code with fewer lines than C ++ and Java. It makes these features a specific language suitable for easy-to-read code suitable for small or large scale and scalable. Python holds several libraries for data analysis, some of which are::

   - Scikit-learn:

Figure 4.2: Scikit-learn Data mining software

Scikit-learn is a free python library for mechanical learning and its main features are classification, regression, clustering, gradient boosting, k-means, DBSCAN and is designed to work with other Python libraries (NumPy and SciPy) for scientific and numerical calculations mainly.

- Pandas:

Figure 4.3: Rapidminer Data mining software

Pandas is a Python library for data management and analysis. Specifically, it provides suitable data structures and methods for managing multidimensional arrays and time series. It is compatible with the BSD license and is a free library software to use.

- Statsmodel:

Figure 4.4: Statsmodel Data mining software

Statsmodel is a Python library that allows users to explore data, evaluate different statistical models, apply statistical controls, is a suitable library for descriptive statistics, for graphs. It is a subset of the superset of SciPy library-based libraries.

- Anaconda:

Figure 4.5: Statsmodel Data mining software

Anaconda was created and promoted by continium, a suite of open source libraries from both Python and R, and is suitable for large-scale data analysis and processing, forecasting and computing computing. Its purpose is to collect these appropriate libraries for the above functions for the simplified management of large data. Anaconda is in line with the Open Data Science innovation, based on open source communities for better and more quality data analysis, using over 720 data analysis libraries to visualize them, for mechanical learning, deep learning and resource management large data. Includes the appropriate libraries for Apache Hadoop and Spark to handle large python data. It also includes suitable libraries for deep learning mechanical learning and image processing such as theano, tensorflow, neon, h2o, keras etc.

3. Rapidminer

Figure 4.6: Rapidminer Data mining software

Rapidminer is a software from Rapidminer and is a complete platform for engineering learning, deep learning, text analysis, and predictive analytics. Use for commercial, business, research and educational use. It includes all the mechanical learning processes that are the preparation of data, visualization of results, verification of results and optimization of analyzes. It is available in a free version but with limited capabilities of 1 logical kernel computing resources and 10000 training examples. The commercial version provides all the software features at an initial price of $2500. The original YALE (Yet Another Learning Environment) was developed in 2001 by Ralf Klinkenberg, Ingo Mierswa, and Simon Fischer in the Artificial Intelligence Lab at the Dortmund Polytechnic University. In 2006 its development continued with Rapid-l, with founders Ingo Mierswa and Ralf Klinkenberg. 2007 was renamed by YALE to Rapidminer as Rapidminer was also renamed to Rapidminer in 2013. Rapidminer has received up to now several downloads of its free software version and over 250,000 commercial users including BMW, Cisco, GE and Samsung. You are the leader in data analysis software and compete with SAS and IBM.

4. Orange

Figure 4.7: Orange Data mining software

Orange is an open source software for data visualization, mechanical learning and knowledge mining from the data. Its main feature is visual programming, front end explorative data analysis and data visualization. Its development began at the University of Ljubljana in 1997. It is currently in version 3.4. It is based on Python, Cython, C ++, C. It is cross platform software and follows the GNU General Public License.

5. KNIME

Figure 4.8: Knime Data mining software

Knime (The Konstanz Information Miner) is an open source software for data analysis, data visualization, data mining and mechanical learning. Its graphical environment is developed so that a user can choose the tools - functions - transformations he wants to implement and with which order he wishes to do in the data without writing code fragments, and this provides user - friendly usability for analyzing the data. Knime is mostly used for pharmaceutical research, customer data analysis (CRM), business intelligence for forecasting, data analysis and visualization models.

6. Weka

Figure 4.9: Weka Data mining software

Weka is a free program for mechanical learning and data analysis based on and written in Java code. It was developed by the University of Waikato in New Zealand and is in line with the GNU General Public License. It includes a set of visualization tools for data analysis, prediction models, pre-processing data, and a user-friendly interface environment. Historian started developing this software from the University of Waikato in New Zealand in 1993 with original programming language Tcl / Tk and Makefiles. In 1997, the development of the Java program began again. In 2005, the software was awarded the SIGKDD Data Mining and Knowledge Discovery Service Award. And in 2006, Pentaho Corporation has attributed Weka certification as a software for business intelligence, data mining and predictive analytics. By 2011, it has received 2,487,213 downloads from sourceforge.net.

7. SPSS

Figure 4.10: IBM SPSS, Data Analysis Software

SPSS, is an IBM software for statistical analysis. Originally developed by SPSS Inc. and in 2009 it was transferred to IBM. You mainly use in areas such as marketing, health sciences, social sciences, text and emotion analysis, knowledge mining, data analysis from gallop. The main functions of SPSS for data analysis are statistical statistics with variable frequencies, cross tabulation, ratio statistics, and Bivariate Statistics. Correlations, ANOVA, Non-parametric controls. Predictions with multiple linear regrowth. Analysis of factors, clustering analysis, discreet analysis as well as visualization of data - examples.

8. Excel

Figure 4.11: Excel SpreadSheet Application

Microsoft Excel is a spreadsheet software and is a cross platform software for Windows, macOS, Android and iOS and one of the software packages of the Microsoft Office software suite. Its features are various computations of spreadsheet data, graphical depictions - their representations, script programming with Visual Basic for Applications on spreadsheet data. It's a well-known spreadsheet software. Microsoft Excel has a key feature in its spreadsheets. This is the use of cells for data management and the use of spreadsheets cells and columns for data management, computational operations and graphical representations. Microsoft Excel includes a large set of statistical and econometric functions for displaying graphs, histograms, etc. Its first version was in 1987, and the latest version is called Excel 2016 where the stable and final version of the software was released in April 2016. An easy-to-use Microsoft Excel tool from 2010, 2013, 2016 is Power Pivot, with the main goal of developing linear regression models for spreadsheet data, and mainly uses DAX (Data Analysis Expressions) as the primary language money within the tool to create the models.

Chapter 5

Application - ML Models for STLF in Greek Electrical Network

5.1 Goals - Development Platform - Software Environment

For this project R Language was selected to be used as development software language. The R 3.4.1 version with R-Studio as IDE platform and R-Markdown, R-Shiny for visualizations. The main goal for the development of an application in R, is to develop Machine Learning Algorithms in order to analyze the load demand in the Greek Electricity Network and make predictions with less error than the existed one OoEM (ΛΑΓΙΕ it will be described in the next sections) has.

5.2 Datasets

The Datasets come from 3 origins.

1. The IPTO's Loads
2. The OoEM's predictions
3. The Meteorological Data

The Datasets span from 2010-10-01 to 2017-04-30

5.2.1 IPTO Loads

The Electricity Load Demand for Greece come from IPTO (ΑΔΜΙΕ in Greece).IPTO stands for Independent Power Transmission Operator. Is the Operator of the Hellenic Electricity Transmission System, IPTO's mission is to ensure Greece's electricity supply in a safe, efficient and reliable manner while promoting the development of competition in the Greek electricity market and guaranteeing the non-discriminatory treatment of System users. Their main goal is To be one of the most efficient electricity transmission system operators in Europe and build value for all stakeholders within a framework of sustainable development while respecting man, the environment and benefiting System Users and society as a whole. IPTO's current organizational structure is the following:

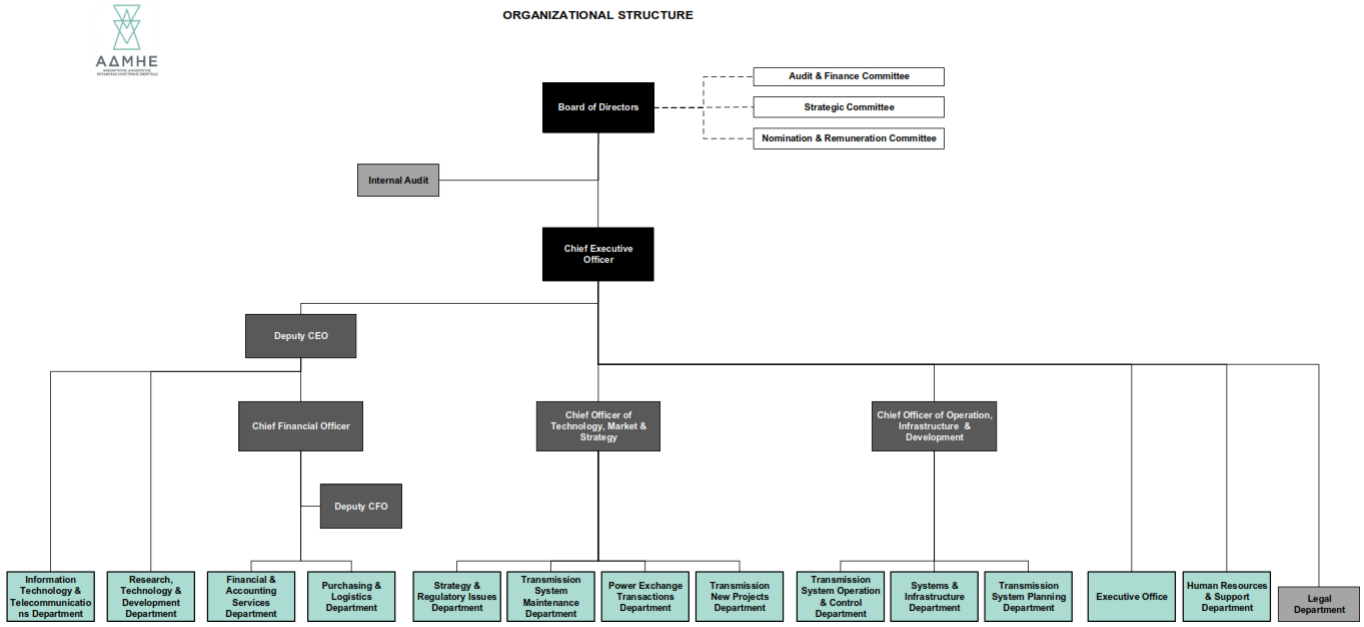

Figure 5.1: IPTO organizational Structure

The IPTO's Loads Demand Datasets are in hourly format and span from 2010-10-01 to 2017-04-30 roughly 6.5 years of data.

5.2.2 OoEM Predictions

OoEM (ΛΑΓΙΕ in Greek) applies the rules for the operation of the Electricity Market in accordance with the provisions of Law 4001/2011 and its delegated acts and in particular the Daily Energy Planning. It predicts the electricity load demand for the next day for the whole Greek electricity network. This task is very important because OoEM needs the know how many loads in MW will be needed in the network per hour because the electric power units in the Greek region must be prepared and be ready to produce the right amount of energy for the needs of consumers and if energy purchase from the neighboring countries is needed.

5.2.3 Meteorological Data

Meteorological Data as features are needed in order to make predictions. Meteorological Data was fetched from darksky.net. The Dark Sky Company specializes in weather forecasting and visualization. It provides an API with REST call which gives meteorological data for a geographical location in JSON format. Meteorological Data are taken from the two biggest cities of Greece, Athens and Thessaloníki. The concept behind these two cities is from a paper which dealt the same issue to predict the short-term electric load forecasting from A. Adamakos [10], but the difference here is the use of more meteorological data than the paper used. The date period of the meteorological data are the same of the IPTO's loads and OoEM predictions.

5.3 Electricity Load Demand - Summary Statistics

A typical electical load demand plot from a weekday is the following:

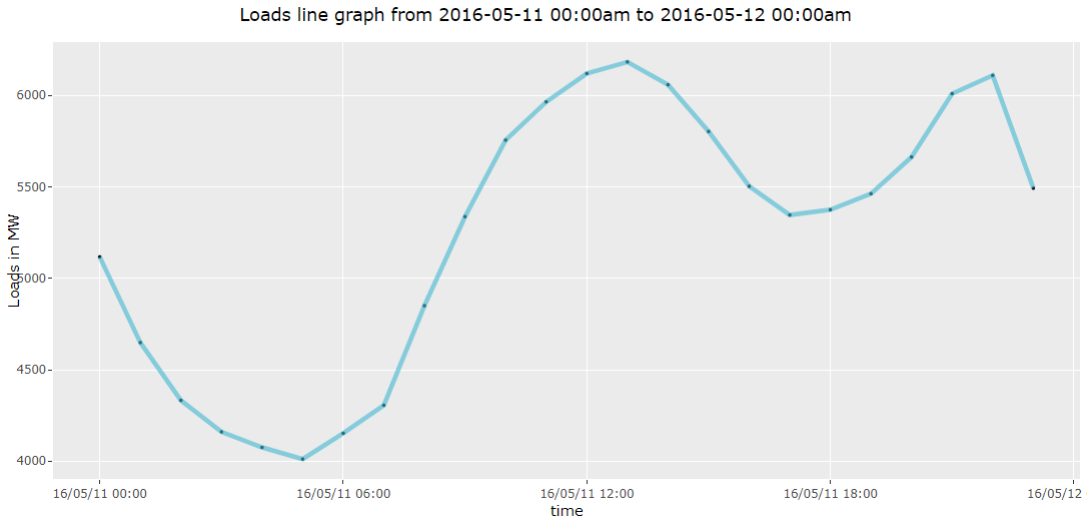

Figure 5.2: Load Demand Line Plot

We can see that from 00:00 - 7:00 (non-rush hours) load demand is decreasing, from 7:00 - 14:00 (rush hours) load are increasing, from 14:00 - 18:00 (non-rush hours) load are decreasing, from 18:00 - 21:00 again loads are increasing and finally from 21:00 - 00:00 loads are decreasing.

5.4 Meteorological Features - Summary Statistics

The meteorological Features taken from Dark Sky API for Athens and Thessaloníki in hourly basis are the following:

- summary
  Categorical variable, summary of current weather conditions.
- icon
  Categorical variable, more detailed edition of current weather conditions.
- temperature
  Continuous variable, the temperature in the selected location measured in Celsius degrees.
- dewpoint
  Continuous variable, the dew point in the selected location measured in Celsius degrees.
- humidity
  Continuous variable, the humidity in the air in the selected location in measured percentage.
- windSpeed
  Continuous variable, the wind speed the air flows in the selected location measured in meters per second.
- windBearing
  Continuous variable, the bearing of the wind flows in the selected location measured in degrees.
- visibility
  Continuous variable, the visibility in the selected location in kilometers.
- CloudCover
  Continuous variable, measuring how much the sky is covered by clouds in the selected location measured in percentage.

- uvIndex
  Continuous variable, measuring sun's radiation during the day in the selected location.

All meteorological Features follows the S.I. units.

5.5 Preprocessing - Cleaning the Data

After fetching all the datasets, one vital step is the preprocessing. Downloading the meteorological data from DarkSky API comes with many missing values. A missing value can be a problem if we leave it untouched because we can have a problematic case and may be omitted during predictions. Thus all the missing values was filled by:

1. fill the missing values by the adjacent by hour values.
2. if the previous are missing values too, fill the missing values by the adjacent by day values.
3. but if and the previous are missing are missing values too, copy one of the adjacent values to the missing value.

5.6 Calendar Features

Calendar features were constructed from the dates of the meteorological dataset. The reason behind is that specific hours and specific days affect the human behavior beyond meteorological factors. Hence the following features were constructed:

- isRushHour, categorical variable, if a hour is rush hour or not
- isHoliday, categorical variable, if that day is holiday (Greek National and Religious Holidays).
- isWeekend, categorical variable, if that day is weekend.

5.7 Joining the Datasets and Constructing the Cases

IPTO's Loads and Dark Sky meteorological data are two different datasets that needs to be joined together in order to construct the cases. The common field was the time. So an inner join was performed in order to join the two datasets. The final case again based on A. Adamakos paper but with more meteorological and Calendar features is the following:

Table 5.1: My caption

	Case and its Features
1	**Weather forecast in Athens for the forecasted day(d).**
2	**Weather forecast in Thessaloniki for the forecasted day(d).**
3	**Weather data in Athens for preceding day(d-1) of the forecasted day(d).**
4	**Weather data in Thessaloniki for preceding day(d-1) of the forecasted day(d).**
5	**hourly load measurements of the second preceding day (d-2) of the forecasted day(d).**
6	**hourly load measurements of the second preceding day (d-2) of the forecasted day(d).**
7	**Current Loads for the forecasted day (d)**

5.8 Adding Noise

Due to the fact that we had historical past observed data from the meteorological data, we wanted to add a bit of noise to the data in the part of the cases where weather forecast is used (columns 1 and 2), because sometimes weather forecast is not accurate enough. So we added a small noise (+/- 0.5 from uniform distribution).

5.9 Feature Selection

In order to make predictions several experiments were made in order to minimize the error of prediction. One technique was the feature selection. The feature selection algorithm that was used is named ”Boruta”. Boruta is a feature selection algorithm. Precisely, it works as a wrapper algorithm around Random Forest. This package derive its name from a demon in Slavic mythology who dwelled in pine forests. We know that feature selection is a crucial step in predictive modeling. This technique achieves supreme importance when a data set comprised of several variables is given for model building. Boruta can be your algorithm of choice to deal with such data sets. Particularly when one is interested in understanding the mechanisms related to the variable of interest, rather than just building a black box predictive model with good prediction accuracy.

5.10 Machine Learning - Experiments

During Machine Learning experimentation 6 machine Learning algorithms were used:

1. SVM: from package e1071
2. K-Nearest Neighbors: from package: FNN
3. Random Forest: from package: randomForest
4. Neural Networks: from package: RSNNS
5. XGBoost: from package: XGBoost
6. Model Trees: from package: Cubist

Four Experiments were performed in order to minimize the error of prediction and increase the prediction performance. The type of experiments are the following:

1. Experiment #1:
   Run the machine learning algorithms with their default parameters, and cases as is without feature selection but with full features. The dataset partition was the following: train-set: the first 5.5 years and test-set: the last 1 year
2. Experiment #2:
   Run the machine learning algorithms with their default parameters, and cases with feature selection. The dataset partition was the following: train-Set: the first 5.5 years and test-set: the last 1 year
3. Experiment #3:
   Run the machine learning algorithms with tuned parameters from grid search, and cases with as is with full features. The dataset partition was the following: train-set: the first 4.5 years, validation-set: the following 1 year and test-set: the last 1 year.

4. Experiment #4:
   Run the machine learning algorithms with tuned parameters from grid search, and cases with feature selection. The dataset partition was the following: train-set: the first 4.5 years, validation-set: the following 1 year and test-set: the last 1 year.

5.11 Forecasting Experiment with Full Features and Default Parameters

This is the first experiment, in this experiment we ran the machine learning algorithms with their default parameters, and cases as is without feature selection but with full features. The dataset partition was the following: train-set: the first 5.5 years and test-set: the last 1 year.

24 Models where trained per Response Variable and per machine learning algorithm as many are the hours in the day (24 hours of the day). The performance results are presented in the following table:

Table 5.2: mape evaluation performance from various models with default model parameters for the 24-hour based response variables

mape evaluation performance from various models with default model parameters for the 24-hour based response variables						
response variable / models	**svm**	**knn**	**random Forest**	**nn**	**xgboost**	**Model Trees**
load.0	2.76	3.57	2.91	4.67	2.23	2.03
load.1	2.63	3.48	2.7	4.61	2.43	1.92
load.2	2.57	3.44	2.58	4.28	2.22	2.1
load.3	2.7	3.51	2.62	4.22	2.53	2.15
load.4	2.73	3.74	2.72	4.04	2.6	2.36
load.5	2.71	3.71	2.71	3.87	2.56	2.54
load.6	3.01	3.86	2.92	4.16	2.94	2.87
load.7	3.78	4.97	3.27	3.67	3.3	2.73
load.8	5.58	7.17	3.74	11.5	3.83	3
load.9	6.08	7.86	4.22	6.83	3.65	3.47
load.10	5.72	7.52	3.77	5.94	3.3	3.17
load.11	5.08	6.58	3.68	4.35	3.04	2.85
load.12	4.53	5.87	3.37	4.35	3.07	2.54
load.13	4.47	5.81	3.21	4.43	2.72	2.45
load.14	4.68	6.18	3.33	5.01	3.12	2.62
load.15	5.23	6.9	3.77	5.43	2.98	2.85
load.16	5.63	7.11	4.07	5.73	4.07	3.06
load.17	5.46	7.05	4.14	5.86	3.93	3.46
load.18	5.67	6.85	4.29	6.29	3.83	3.48
load.19	5.74	6.36	3.99	12.7	3.95	2.95
load.20	6.03	5.81	3.59	14.1	3.35	2.39
load.21	4.29	5.19	3.06	11.1	2.63	2.26
load.22	3.57	4.71	3.15	3.73	2.74	2.26
load.23	3.01	4.61	3.42	4.94	2.87	2.27

The following table shows the mean mape for all the response variables per machine learning algorithm:

Table 5.3: mean mape for the 24 models per machine learning algorithm with default parameters and full features

mean mape for the 24 models per machine learning algorithm with default parameters and full features						
	svm	**knn**	**random forest**	**nn**	**xgboost**	**Model Trees**
mean mape	4.31	5.49	3.38	6.07	3.07	2.65

Some Remarks

- Model Trees with Cubist library, performed almost perfectly, with 2.65% error. Only Model Trees and no other ML algorithm has the best performance per response variable.
- On the second and third place takes the XGBoost with 3.07% and RandomForest with 3.38% further experimentation will increase their performance.
- Having as minimum mean mape prediction error 2.65% and the load prediction error from OoEM is 5.35%, thus this experiment increase the predictive accuracy performance by **50.46%**.

5.12 Forecasting Experiment with Feature Selection and Default Parameters

This is the second experiment, in this experiment we ran the machine learning algorithms with their default parameters, and cases with feature selection. The dataset partition was the following: train-set: the first 5.5 years and test-set: the last 1 year.

24 Models where trained per Response Variable and per machine learning algorithm as many are the hours in the day (24 hours of the day). The performance results are presented in the following table:

Table 5.4: mape evaluation performance from various models with tuned parameters for the 24-hour based response variables

mape evaluation performance from various models with tuned parameters for the 24-hour based response variables						
response variable / models	**svm**	**knn**	**random Forest**	**nn**	**xgboost**	**Model Trees**
load.0	2.17	3.58	2.64	3.4	2.11	2.05
load.1	2.17	3.49	2.49	3.8	2.34	2.23
load.2	1.96	3.43	2.3	3.68	2.29	1.99
load.3	1.93	3.48	2.28	3.72	2.32	2.12
load.4	2.16	3.73	2.45	3.42	2.52	2.45
load.5	2.13	3.7	2.49	3.39	2.7	2.4
load.6	2.29	3.87	2.64	3.12	2.68	2.67
load.7	2.8	4.98	2.95	3.25	2.87	2.76
load.8	3.82	7.19	3.34	4.73	3.59	2.75
load.9	4.33	7.89	3.7	4.29	3.61	3.37
load.10	3.96	7.53	3.5	5.95	3.23	2.85
load.11	3.44	6.5	3.27	5.22	3.04	2.99
load.12	2.96	5.91	3.05	4.8	2.85	2.37
load.13	2.94	5.81	2.95	4.71	2.68	2.5
load.14	3.07	6.14	2.99	5.63	2.95	2.56

Table 5.4 continued from previous page

mape evaluation performance from various models with tuned parameters for the 24-hour based response variables						
load.15	3.59	6.93	3.33	6.45	3.43	2.96
load.16	3.69	7.12	3.69	5.81	3.78	3.08
load.17	3.98	6.98	3.76	5.94	3.77	3.35
load.18	4.19	6.81	3.87	6.38	3.67	3.22
load.19	4.45	6.42	3.5	6.4	3.41	2.64
load.20	4.18	5.75	3.36	8.33	3.64	2.52
load.21	2.87	5.19	2.83	7.56	2.57	2.15
load.22	2.38	4.71	2.86	4.6	2.64	2.19
load.23	2.39	4.58	3.02	4.89	2.49	2.31

The following table shows the mean mape for all the response variables per machine learning algorithm:

Table 5.5: mean mape for the 24 models per machine learning algorithms with default parameters and feature selection

mean mape for the 24 models per machine learning algorithms with default parameters and feature selection						
	svm	**knn**	**random forest**	**nn**	**xgboost**	**Model Trees**
mean mape	3.07	5.48	3.05	4.97	2.96	2.6

Some Remarks

- Model Trees with Cubist library, performed again almost perfectly, it went from 2.65% prediction error to 2.6% prediction error.
- However now with feature selection SVM started to perform better than the previous experiment, its prediction error decreased from 4.31% to 3.07% and in some response variables performs better than Model Trees.
- XGBoost although is increase its performance than the previous experiment, it comes third in mean performance but it overruns SVM in most of response variables.
- one significant tactic here is to combine different models from repsonse variable to response variable. Here from 00:00 to 6:00 we can use SVM and the rest Model Trees. If we do this we get even better performance with prediction error **2.55%**. The load prediction error from OoEM is 5.35%, thus this experiment increase the predictive performance by **52.33%**.

5.13 Forecasting Experiment with Full Features and Grid Search

This is the third experiment, in this experiment we ran the machine learning algorithms with tuned parameters from grid search, and cases as is with full features. The dataset partition was the following: train-set: the first 4.5 years, validation-set: the following 1 year and test-set: the last 1 year.

The grid search tuning parameters per machine learning algorithm are being presenting in the following table:

Table 5.6: Grid Search tuning Parameters

Grid Search - Tuning Parameters				
Models	***List of parameters per model***			
SVM	Radial Kernel	Gamma paremmeter	Cost parameter	
K-Nearest Neighbors	KNN algorithm (kd-tree, brute, cover-tree)	Number of Neighbors		
Random Forest	number of trees	mtry, (Number of variables randomly sampled as candidates at each split)		
Neural Networks	Resilient Backpropagation	Learning Rate	Number of Neurons at the hidden Layer	maxit, (maximum of iterations to learn)
XGBoost	Tree Booster	eta (control the learning rate)	max_depth (maximum depth of a tree)	nrounds, (the max number of iterations)
Model Trees	unbiased, (should unbiased rules be used?)	committes = 1, how many committee models (boosting iterations) should be used? For fairness to other models, commitee = 1		

24 Models where trained per Response Variable and per machine learning algorithm as many are the hours in the day (24 hours of the day). The performance results are presented in the following table:

Table 5.7: mape evaluation performance from various models with tuned parameters and full features for the 24-hour based response variables

mape evaluation performance from various models with tuned parameters and full features for the 24-hour based response variables						
response variable/ models	**svm**	**knn**	**random Forest**	**nn**	**xgboost**	**Model Trees**
load.0	2.53	3.55	2.93	2.89	2.12	2.06
load.1	2.48	3.39	2.61	3.64	2.05	1.92
load.2	2.41	3.23	2.61	2.89	1.97	2.12
load.3	2.51	3.32	2.52	2.9	2.14	2.16
load.4	2.6	3.55	2.84	3.24	2.3	2.32
load.5	2.53	3.62	2.79	3.16	2.16	2.5
load.6	2.69	3.85	3.03	2.91	2.31	2.87
load.7	3.07	4.99	3.27	3.47	2.81	2.73
load.8	4.19	7.23	4.01	4.17	3.48	3
load.9	4.56	8.01	4.28	4.06	3.68	3.47
load.10	4.74	7.5	3.85	5.15	3.37	3.17
load.11	4.38	6.36	3.55	4.22	2.84	2.85
load.12	3.95	5.53	3.52	4.42	2.68	2.54
load.13	3.99	5.45	3.34	3.9	2.63	2.45
load.14	4.15	5.89	3.37	4.61	2.66	2.62

Table 5.7 continued from previous page

mape evaluation performance from various models with tuned parameters and full features for the 24-hour based response variables						
load.15	4.59	6.66	3.9	4.83	2.78	2.85
load.16	4.97	7.03	4.16	5.7	3.22	3.06
load.17	4.91	6.86	4.12	4.97	3.11	3.46
load.18	4.92	6.75	4.45	5.42	3.21	3.48
load.19	5.03	6.24	3.84	6.37	3.26	2.98
load.20	4.93	5.63	3.64	5.06	3.26	2.45
load.21	3.73	5.03	3.21	4.31	2.8	2.26
load.22	3.15	4.52	3.35	3.85	2.52	2.37
load.23	2.96	4.39	3.28	3.25	2.41	2.27

The following table shows the mean mape for all the response variables per machine learning algorithm:

Table 5.8: mean mape for the 24 models per machine learning algorithms with tuned parameters and full features

mean mape for the 24 models per machine learning algorithms with tuned parameters and full features						
	svm	**knn**	**random forest**	**nn**	**xgboost**	**Model Trees**
mean mape	3.74	5.35	3.43	4.14	2.73	2.66

Some Remarks

- Again Model Trees has overall the best mean performance with 2.66% prediction error.
- XGBoost got even better better performance than before with prediction error 2.73%. On the other hand SVM performance became even worse than before. KNN can not match the in performance the other ML algorithms.
- Even though Model Trees has the best mean performance from the othr five machine learning algorithms, there are some specific response variables which XGBoost shows better performance. If we combine models per response variable and mix Model Trees and XGBoost per response variable by mape performance we get even small prediction error **2.58%**, which is worst than before. This implies that combining the two tactics feature selection and grid search may lead us to even better results. The load prediction error from OoEM is 5.35%, thus this experiment increase the predictive performance by **51.77%**.

5.14 Forecasting Experiment with Feature Selection and Grid Search

This is the forth and last experiment, in this experiment we ran the machine learning algorithms with tuned parameters from grid search, and cases as is with feature selection. The dataset partition was the following: train-set: the first 4.5 years, validation-set: the following 1 year and test-set: the last 1 year.

The grid search tuning parameters per machine learning algorithm are being presenting in the following table:

Table 5.9: Grid Search tuning Parameters

Grid Search - Tuning Parameters				
Models	***List of parameters per model***			
SVM	Radial Kernel	Gamma paremmeter	Cost parameter	
K-Nearest Neighbors	KNN algorithm (kd-tree, brute, cover-tree)	Number of Neighbors		
Random Forest	number of trees	mtry, (Number of variables randomly sampled as candidates at each split)		
Neural Networks	Resilient Backpropagation	Learning Rate	Number of Neurons at the hidden Layer	maxit, (maximum of iterations to learn)
XGBoost	Tree Booster	eta (control the learning rate)	max_depth (maximum depth of a tree)	nrounds, (the max number of iterations)
Model Trees	unbiased, (should unbiased rules be used?)	committes = 1, how many committee models (boosting iterations) should be used? For fairness to other models, commitee = 1		

24 Models where trained per Response Variable and per machine learning algorithm as many are the hours in the day (24 hours of the day). The performance results are presented in the following table:

Table 5.10: mape evaluation performance from various models with tuned model parameters and feature selection for the 24-hour based response variables

mape evaluation performance from various models with tuned model parameters and feature selection for the 24-hour based response variables						
response variable/ models	**svm**	**knn**	**random Forest**	**nn**	**xgboost**	**Model Trees**
load.0	1.78	3.57	2.73	2.46	2.01	2.05
load.1	1.86	3.4	2.53	2.59	1.94	2.23
load.2	1.77	3.22	2.33	2.57	2.05	1.99
load.3	1.79	3.27	2.29	2.55	2.04	2.13
load.4	2.05	3.52	2.49	2.58	2.23	2.43
load.5	1.94	3.61	2.47	2.99	2.1	2.37
load.6	2.08	3.83	2.64	2.67	2.22	2.67
load.7	2.3	4.97	2.99	2.59	2.63	2.76
load.8	3.45	7.21	3.49	4.69	3	2.75
load.9	3.45	8	3.8	4.47	3.49	3.37
load.10	2.93	7.49	3.64	11.3	3.08	2.85
load.11	2.69	6.43	3.28	5.32	2.74	2.99
load.12	2.16	5.61	3.04	4.12	2.59	2.37
load.13	2.43	5.44	3.02	3.83	2.5	2.5
load.14	2.56	5.84	2.99	6.28	2.54	2.56

Table 5.10 continued from previous page

mape evaluation performance from various models with tuned model parameters and feature selection for the 24-hour based response variables						
load.15	3.04	6.62	3.4	5.12	2.78	2.96
load.16	3.19	7.02	3.73	4.77	3.05	3.08
load.17	3.91	6.76	3.75	4.84	3.46	3.35
load.18	3.62	6.75	3.84	4.47	3.26	3.22
load.19	3.86	6.24	3.55	4.51	3.3	2.64
load.20	3.71	5.68	3.3	3.97	3.25	2.55
load.21	2.67	5.06	2.91	4.14	2.79	2.15
load.22	1.95	4.52	3.01	3.86	2.38	2.19
load.23	1.94	4.36	3.17	3.21	2.13	2.31

The following table shows the mean mape for all the response variables per machine learning algorithm:

Table 5.11: mean mape for the 24 models per machine learning algorithms with tuned parameters and feature selection

mean mape for the 24 models per machine learning algorithms with tuned parameters and feature selection						
	svm	**knn**	**random forest**	**nn**	**xgboost**	**Model Trees**
mean mape	2.62	5.35	3.09	4.16	2.64	2.60

Some Remarks

- Again Model Trees has overall the best mean performance with 2.60% prediction error.
- XGBoost got the best performance than every other experiment with prediction error 2.64%.
- SVM also got the best performance than every other experiment with prediction error 2.62%.
- Again if we combine and mix models from response variable to response variable by minimum mape we get the best combined (ensemble) performance ever which is **2.41%**. The load prediction error from OoEM is 5.35%, thus this experiment increase the predictive performance by **54.9%**.

5.15 R-Shiny, R-Markdown Visualizations

To visualize the loads, the meteorological data, out predictions, the OoEM predictions, and display some descriptive statistics a R-Shiny / R-Markdown application was created. The reason behind this is that visualizations give a different perspective by visualizing the results and sometimes can find issues that can not be found from R-Studio command line.

The application is called IPTO ML, is interactive and the user can see for different dates and durations the real load values, the OoEM prediction, this project predictions, descriptive statistics and the performance of predictions in mape evaluation. The application is deployed at shinyapps, and its page is named: IPTO ML.

The initial page is the following:

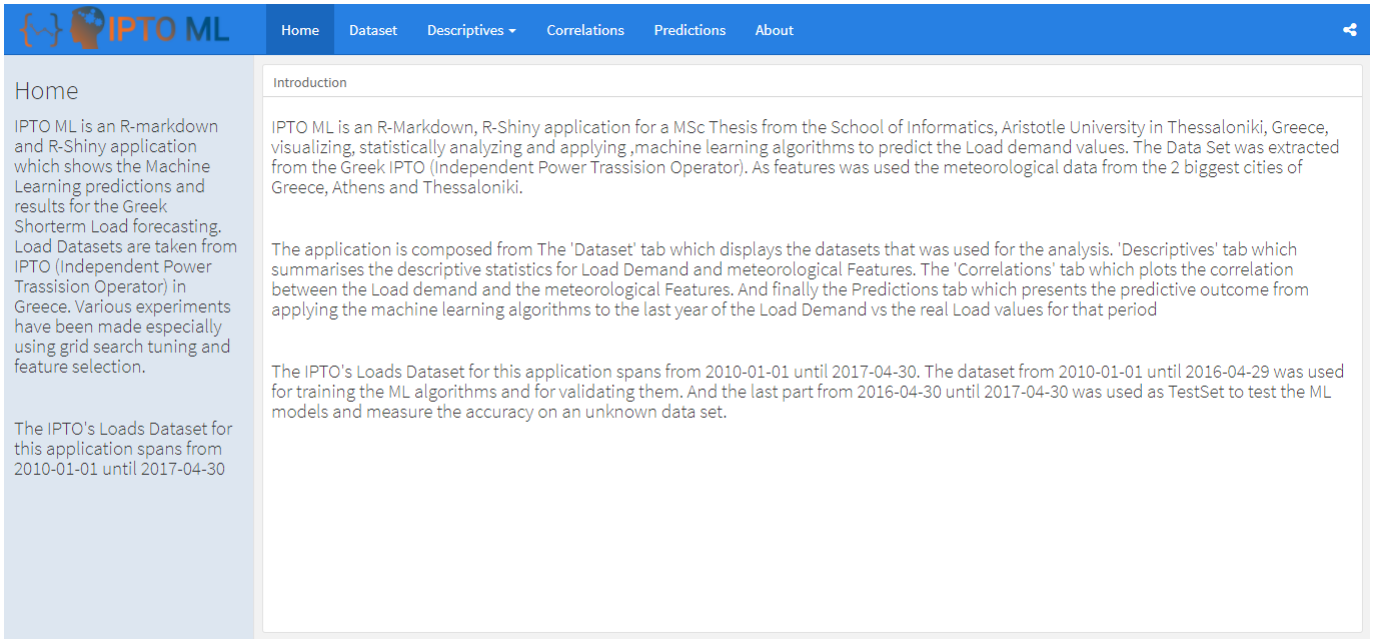

Figure 5.3: IPTO ML first page

The second page presents both loads and meteorological features in table format

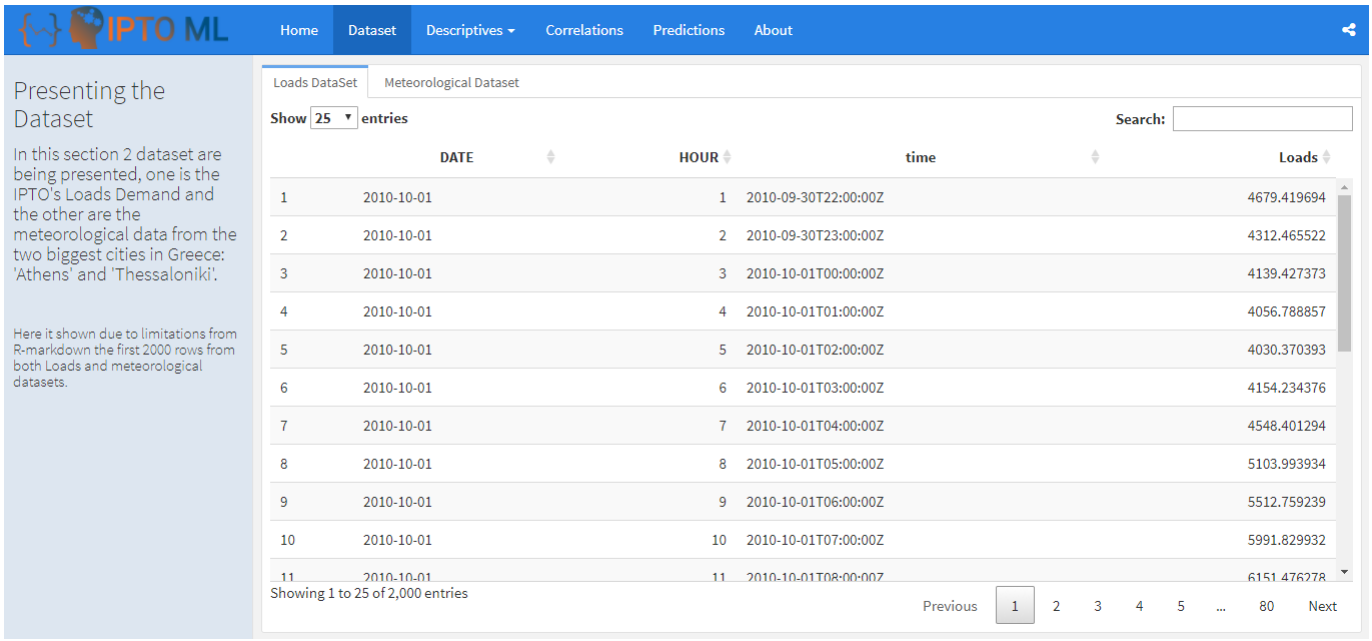

Figure 5.4: IPTO ML second page

The third page shows descriptive statistics for both loads and the meteorological features (histograms and boxplot)

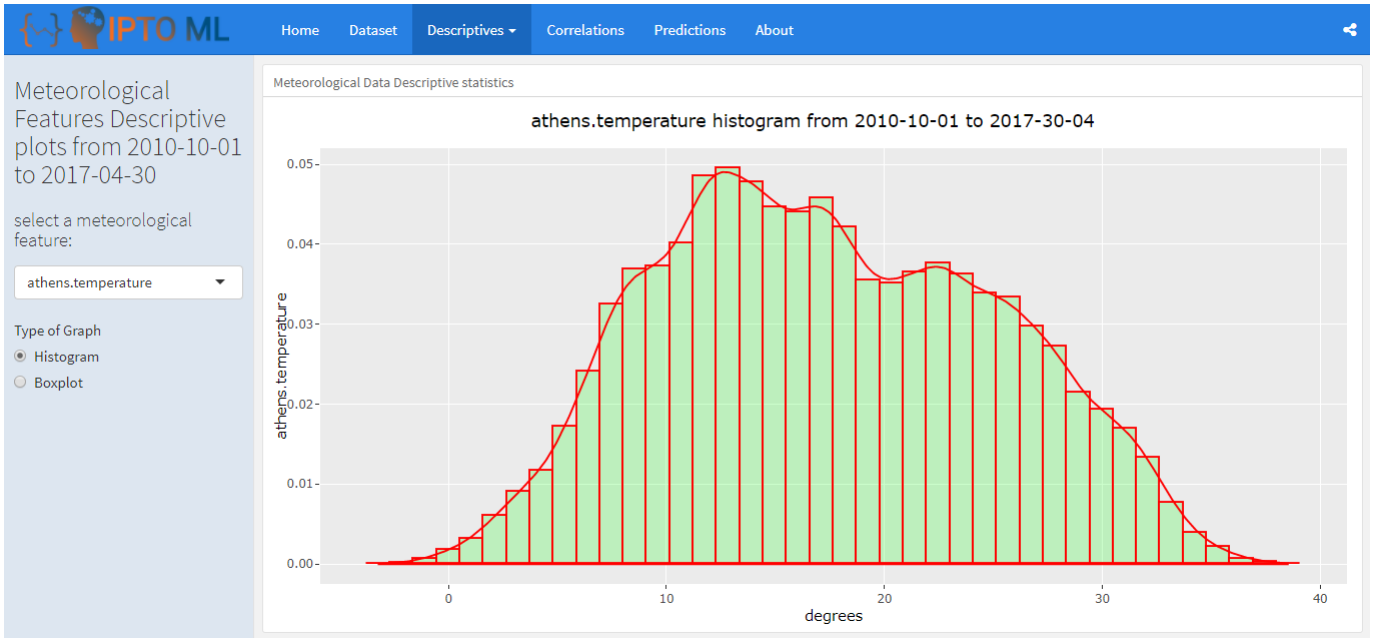

Figure 5.5: IPTO ML third page

The forth page shows the correlation between electricity load demand and meteorological features.

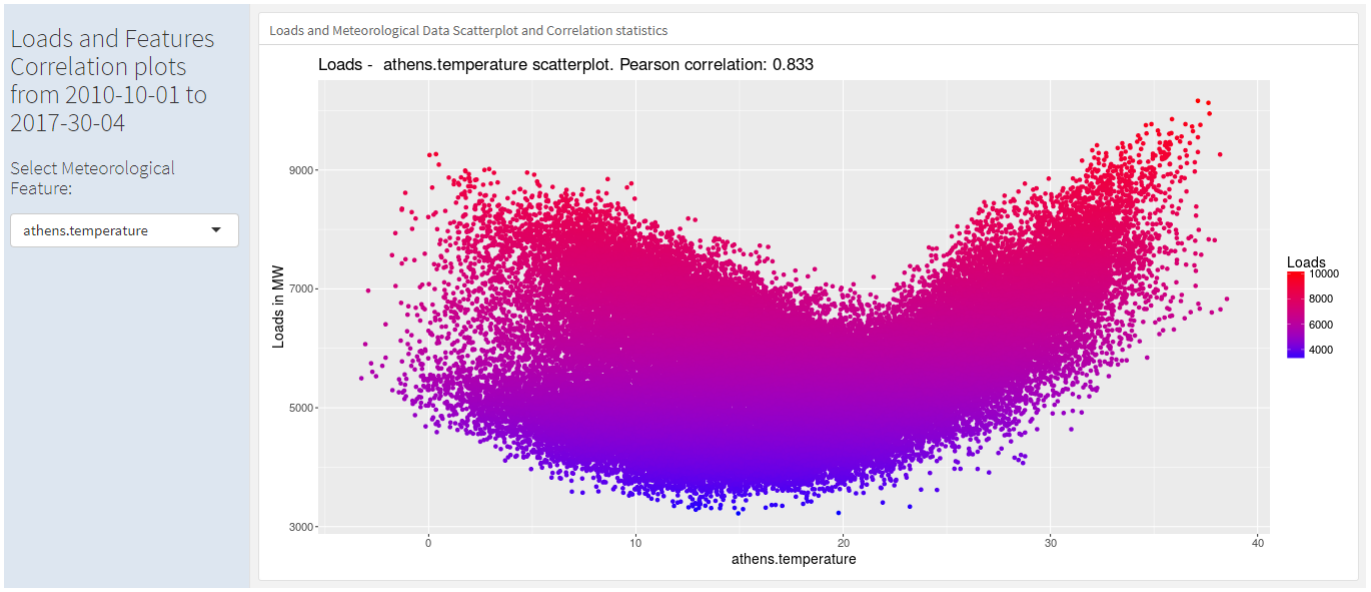

Figure 5.6: IPTO ML forth page

And the fifth page shows the correlation between electricity load demand and meteorological features

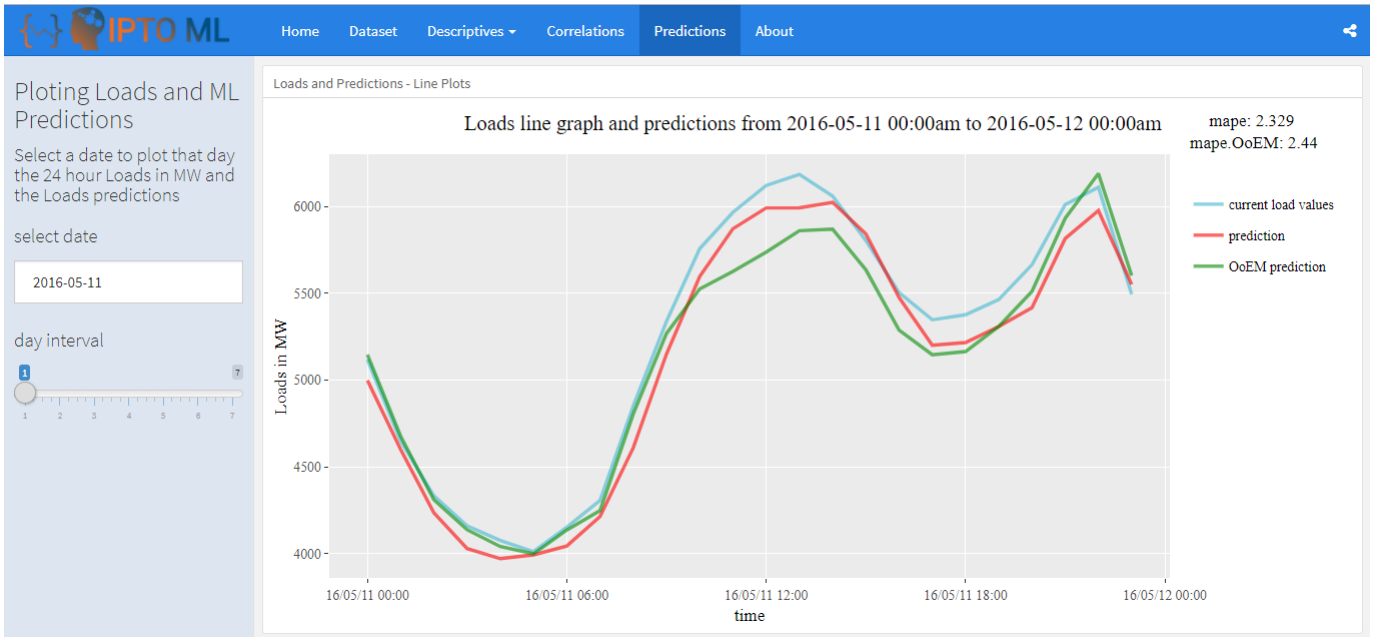

Figure 5.7: IPTO ML fifth page

Chapter 6

Conclusions - Final Remarks

As presented in the initial chapters, electrical load demand forecasting is a vital processing nowadays, due to the needs from electrical power from consumers, our daily life and routine. Electrical companies have to take into account this process and constantly improve their models for more accurate forecasts

As seen in the scientific literature before many machine learning and statistical approaches exist to predict the short-term load forecasting. Applying six of them (SVM, Random Forest, K-Nearest Neighbors, Neural Networks, XGBoost, Model Trees) increase the accuracy of OoEM at least 50% with the best prediction error 2.41% One interesting part is that combining models by their evaluation we can increase even more the accuracy of error models in total.

Chapter 7

Future Work

Many techniques have not been developed as seen from the literature. More importantly if they are developed the combination of them should give interesting results. Furthermore techniques of multi-target regression can be applied. Moreover to increase the performance because grid search takes a lot of time, spark library for distributed computing or parallel for loops should be introduced.

Bibliography

[1] Electrical load forecasting : modeling and model construction / Soliman Abdel-hady Soliman (S.A. Soliman), Ahmad M. Al-Kandari.

[2] Short Term Electric Load Forecasting by Tao Hong, A dissertation submitted to the Graduate Faculty of North Carolina State University in partial fulfillment of the requirements for the degree of Doctor of Philosophy, Operations Research and Electrical Engineering, Raleigh, North Carolina

[3] A. D. Papalexopoulos and T. C. Hesterberg, ”A regression-based approach to short-term system load forecasting,” IEEE Transactions on Power Systems.

[4] T. Haida and S. Muto, ”Regression based peak load forecasting using a transformation technique,” IEEE Transactions on Power Systems.

[5] B. Krogh, E. S. de Llinas, and D. Lesser, ”Design and Implementation of An on-Line Load Forecasting Algorithm,” IEEE Transactions on Power Apparatus and Systems.

[6] M. T. Hagan and S. M. Behr, ”The Time Series Approach to Short Term Load Forecasting,” IEEE Transactions on Power Systems.

[7] N. Amjady, ”Short-term hourly load forecasting using time-series modeling with peak load estimation capability,” IEEE Transactions on Power Systems

[8] D. C. Park, M. A. El-Sharkawi, R. J. Marks, II, L. E. Atlas, and M. J. Damborg, ”Electric load forecasting using an artificial neural network,” IEEE Transactions on Power Systems, vol. 6, pp. 442-449, 1991.

[9] D. K. Ranaweera, N. F. Hubele, and A. D. Papalexopoulos, ”Application of radial basis function neural network model for short-term load forecasting,” IEEE Proceedings - Generation, Transmission and Distribution

[10] A. Apostolos, ”Short-Term Load Forecasting using a Cluster of Neural Networks for the Greek Energy Market”, Electrical & Computer Engineer Public Power Corporation Greece.

[11] Sp.Kiartzis, Artificial Intelligence Applications in Short-term Load Forecasting, PhD Dissertation, AUTH, 1998

[12] A.G.Bakirtzis, V.Petridis, S.J.Kiartzis, M.C.Alexiadis, A.H.Maissis, A neural network short term load forecasting model for the Greek power system, IEEE Transactions on Power Systems, Vol. 11, No.2, May 1996, pp. 858-863

[13] D. Papamilios, Load Declaration Software, Dissertation, 2012, AUTH

[14] S. Rahman and R. Bhatnagar, ”An expert system based algorithm for short term load forecast,” IEEE Transactions on Power Systems, vol. 3, pp. 392- 399, 1988.

[15] S. Rahman, ”Formulation and analysis of a rule-based short-term load forecasting algorithm,” Proceedings of the IEEE, vol. 78, pp. 805-816, 1990

[16] K.-L. Ho, Y.-Y. Hsu, C.-F. Chen, T.-E. Lee, C.-C. Liang, T.-S. Lai, and K.-K. Chen, ”Short term load forecasting of Taiwan power system using a knowledge-based expert system,” IEEE Transactions on Power Systems, vol. 5, pp. 1214-1221, 1990.

[17] Y. Y. Hsu and K. L. Ho, ”Fuzzy expert systems: an application to short-term load forecasting,” IEE Proceedings - Generation, Transmission and Distribution, vol. 139, pp. 471-477, 1992.

[18] P. A. Mastorocostas, J. B. Theocharis, and A. G. Bakirtzis, ”Fuzzy modeling for short term load forecasting using the orthogonal least squares method,” IEEE Transactions on Power Systems, vol. 14, pp. 29-36, 1999.

[19] H. Mori and H. Kobayashi, ”Optimal fuzzy inference for short-term load forecasting,” IEEE Transactions on Power Systems, vol. 11, pp. 390-396, 1996.

[20] S. Rahman, ”Formulation and analysis of a rule-based short-term load forecasting algorithm,” Proceedings of the IEEE, vol. 78, pp. 805-816, 1990

[21] N. Sapankevych and R. Sankar, ”Time Series Prediction Using Support Vector Machines: A Survey,” IEEE Computational Intelligence Magazine, vol. 4, pp. 24-38, 2009

[22] Electrical Load Forecasting using Support Vector Machines, Belgin Emre, Dilara Demren, Instanbul Technical University, Turkey.

[23] C.-M. Huang and H.-T. Yang, ”Evolving wavelet-based networks for shortterm load forecasting,” IEE Proceedings - Generation, Transmission and Distribution, vol. 148, pp. 222-228, 2001.

[24] Ensemble Deep Learning for Regression and Time Series Forecasting, Xueheng Qiu, Le Zhang, Ye Ren and P. N. Suganthan, School of Electrical and Electronic Engineering, Nanyang Technological Univeristy, Singapore, Gehan Amaratunga, Department of Engineering, University of Cambridge, UK

[25] Deep Neural Network Based Demand Side Short Term Load Forecasting, Seunghyoung Ryu, Department of Electronic Engineering, Sogang University, 35 Baekbeom-ro, Mapo-gu, Seoul 121-742, Korea; shryu@sogang.ac.kr, Jaekoo Noh, Software Center, Korea Electric Power Corporation (KEPCO), 105 Munji Road, Yuseong-Gu, Daejeon 305-760, Korea; jknoh@kepco.co.kr, Hongseok Kim, Correspondence: hongseok@sogang.ac.kr; Tel.: +82-2-705-7989

[26] Short Term Electrical Load Forecasting Using Mutual Information Based Feature Selection with Generalized Minimum-Redundancy and Maximum-Relevance Criteria, Nantian Huang, Zhiqiang Hu, Guowei Cai and Dongfeng Yang, School of Electrical Engineering, Northeast Dianli University, Jilin 132012 China.

[27] Short-Term Load Forecasting using Random Forests, Grzegorz Dudek, Department of Electrical Engineering, Czestochowa University of Technology, Czestochowa, Poland

[28] Rule-Based Prediction of Short-term electric Load, Petr Berka, Prof, PhD, University of Economics Prague & University of Finance and Administration Prague, Czech Republic

[29] Formulation and analysis of a rule-based short-term load forecasting algorithm, S. Rahman, Dept of Electr. Eng., Virginia Polytech. Inst. & State Univ. Blacksburg, VA, USA, ieeexplore.ieee.org/document/53400

[30] A Novel Hybrid Model Based on Extreme Learning Machine, k-Nearest Neighbor Regression and Wavelet Denoising Applied to Short-Term Electric Load Forecasting, Weide Li, Demeng Kong and Jinran Wu, School of Mathematics and Statistics, Lanzhou University, Gansu, China

[31] A composite k-nearest neighbor model for day-ahead load forecasting with limited temperature forecasts, Rui Zhang, Yan Xu, Zhao Yang Dong, Weicong Kong, Kit Po Wong, School of Electrical and Information Engineering, University of Sydney, NSW 2006, Australia ieeexplore.ieee.org/document/7741097

[32] Short-Term Electricity Load Forecasting Based on the XGBoost Algorithm, Guangye Li, Wei Li, Xiaolei Tian, Yifeng Che, State Grid Liaoning Electric Power Co., Ltd., Shenyang Liaoning

[33] A gradient boosting approach to the Kaggle load forecasting competition, Souhaib Ben Taieb, Machine Learning Group, Department of Computer Science, Faculty of Sciences, Universit´e Libre de Bruxelles, Rob J Hyndman, Department of Econometrics and Business Statistics, Monash University, Clayton, VIC 3800, Australia

[34] O. Chapelle and V. Vapnik, Model Selection for Support Vector Machines. In Advances in Neural Information Processing Systems, Vol 12, (1999)

[35] M. T. Hagan and S. M. Behr, ”The Time Series Approach to Short Term Load Forecasting,” IEEE Transactions on Power Systems, vol. 2, pp. 785-791, 1987.

[36] R language wiki