Energy Market Analysis Using Kernel Methods

Master Thesis

L. Pernigo

June To be defined, 2024

Advisor: Prof. Dr. M. Multerer

Co-Advisor: Dr. D. Baroli

Faculty of Informatics, USI Lugano

Abstract

The theory of kernel methods will be applied to the problem of point and probabilistic forecasting for the energy sector. Such choice is motivated by its interesting implications.

Contents

Contents iii

List of Figures 1

List of Tables 2

1 Problem Description 5
- 1.1 Motivation . . . 5
- 1.2 Point versus probabilistic forecast . . . 6
- 1.3 Aims and objectives . . . 6
- 1.4 Outline . . . 6

2 Literature Review 9
- 2.1 Electricity forecasting classification . . . 9
- 2.2 Bibliographic analysis . . . 10
- 2.3 Electricity forecasting literature review . . . 13
- 2.4 Kernel methods literature review . . . 16

3 Kernel Theory 19
- 3.1 Kernel mean embedding of distributions: A review and beyond 19
  - 3.1.1 RKHS . . . 20
  - 3.1.2 Kernel families . . . 20
- 3.2 Recovering distributions from Gaussian RKHS embeddings . 21
- 3.3 Super samples from kernel herding . . . 22

4 Evaluation metrics 27
- 4.1 Mean absolute error . . . 27
- 4.2 Root mean squared error . . . 27
- 4.3 Mean absolute percentage error . . . 28
- 4.4 Root mean squared percentage error . . . 28
- 4.5 Mean absolute scaled error . . . 28

4.6 Root mean squared scaled error . . . 28
4.7 Pinball . . . 28
4.8 Winkler . . . 29
4.9 Continous ranked probability score . . . 29
4.10 Probability integral transform . . . 31

5 The Energy Market **33**
5.1 Electricity distribution network . . . 33
5.2 Electricity markets . . . 34
5.2.1 The marketplace . . . 34
5.2.2 How auctions work . . . 35
5.2.3 Day ahead . . . 35
5.2.4 Intraday . . . 35
5.2.5 Price peaks . . . 35
5.2.6 Negative prices . . . 36
5.2.7 Nodal and zonal pricing . . . 36
5.3 Smart grids . . . 36
5.4 Renewables . . . 36

6 Point forecasting **39**
6.1 Multiple linear regression . . . 39
6.2 Autoregressive models . . . 39
6.2.1 Autoregressive integrated moving average . . . 40
6.2.2 Autoregressive integrated moving average with exogenous variables . . . 40
6.2.3 Seasonal autoregressive integrated moving average and Seasonal autoregressive integrated moving average with exogenous variables . . . 40
6.3 Generalized additive model . . . 41
6.4 K-nearest neighbours regression . . . 41
6.5 Support vector regression . . . 41
6.6 Artificial neural networks . . . 43
6.6.1 Multilayer perceptrons . . . 43
6.6.2 Long short term memory . . . 44
6.7 Kernel methods . . . 45
6.7.1 Kernel ridge regression . . . 45
6.7.2 Kernel support vector regression . . . 47

7 Probabilistic forecasting **51**
7.1 Quantile regression . . . 51
7.2 Quantile forest . . . 52
7.2.1 Decision trees . . . 52
7.2.2 Bootstrap . . . 54
7.2.3 Bagging . . . 55

7.2.4 Random forest . . . 55
7.2.5 Quantile forest . . . 55
7.3 Quantile gradient boosting machine . . . 57
7.3.1 Boosting . . . 57
7.3.2 Boosted trees . . . 57
7.3.3 Gradient boosting . . . 57
7.4 Kernel methods . . . 58
7.4.1 Kernel quantile regression . . . 58
7.4.2 Weather quantiles . . . 63

8 Exploratory Analysis and Data ETL **67**
8.1 Dataset . . . 67
8.1.1 Price track . . . 68
8.1.2 Load track . . . 68
8.2 ETL . . . 69
8.3 EDA . . . 69

9 Implementation **71**
9.1 Point forecasting . . . 71
9.1.1 Multiple linear regression . . . 71
9.1.2 Tbats . . . 71
9.1.3 Prophet . . . 71
9.1.4 K-nearest neighbours . . . 72
9.1.5 Support vector regression . . . 72
9.1.6 LSTM . . . 72
9.1.7 Kernel ridge regression . . . 72
9.1.8 Kernel support vector regression . . . 72
9.2 Probabilistic forecasting . . . 72
9.2.1 Linear quantile regression . . . 72
9.2.2 Quantile gradient boosting machine . . . 72
9.2.3 Quantile forest . . . 72
9.2.4 Kernel quantile regression . . . 72

10 Experiments Analysis **75**
10.1 Point forecasting . . . 75
10.1.1 Multiple linear regression . . . 75
10.1.2 Trigonometric seasonality Box-Cox transformation ARMA errors trend and seasonal components . . . 76
10.1.3 Prophet . . . 76
10.1.4 K-nearest neighbours . . . 77
10.1.5 Support vector regression . . . 78
10.1.6 Long short term memory . . . 78
10.1.7 Kernel ridge regression . . . 79
10.1.8 Kernel support vector regression . . . 80

10.1.9 Results 81
10.2 Probabilistic forecasting 83

List of Symbols **85**

A Appendix **89**
A.1 Feature map normalization 89
A.2 Quantile regressor extensive comparison 89
A.2.1 Boston housing dataset 89
A.2.2 Abalone dataset 91
A.2.3 Vehicle dataset 92
A.3 Cross validation 93
A.3.1 K-fold cross validation 93
A.3.2 Gridsearch 93
A.3.3 Randomized search 93
A.3.4 Halving gridsearch 94
A.3.5 Cross validation for time series data 94
A.4 Kernel methods best practices 94
A.4.1 Data normalization 95
A.4.2 Data compression 96
A.5 Source code 98

Bibliography **99**

List of Figures

2.1 Time classsification [46] . . . 10
2.2 Electricity forecasting publications over the past years . . . 11
2.3 Point versus probabilistic pubblications over the past years . . . 11
2.4 Pubblications by method over the past years . . . 12
2.5 Pubblications by subject area . . . 12
2.6 Most popolar sources/outlets . . . 13

4.1 Pinball loss with $q = 0.2$ [38] . . . 29
4.2 CRPS integral [38] . . . 30
4.3 PIT types [38] . . . 31

5.1 Electricity grid [35] . . . 33
5.2 Pool market versus power echange [97] . . . 35
5.3 Zonal pricing EU [1] . . . 37
5.4 Local pricing US [4] . . . 38
5.5 Electricity production by source . . . 38

6.1 Support vector regression,[83] . . . 42
6.2 support vector regression . . . 44
6.3 Long short term memory cell . . . 44
6.4 polynomial support vector regression . . . 48
6.5 rbf support vector regression . . . 49

7.1 two-dimensional feature space partitioned by recursive binary splitting . . . 53
7.2 partition tree and regression model . . . 53
7.3 decision tree regression . . . 54
7.4 random forest regression . . . 56
7.5 gradient boosting regression . . . 58
7.6 Melbourne temperatures dataset . . . 64
7.7 Quantile regressors . . . 65

8.1 Price track 68

10.1 Multiple linear regression prediction 76
10.2 Tbats prediction 77
10.3 Prophet prediction 77
10.4 Prophet v2 prediction 78
10.5 Seasonalities breakdown 78
10.6 Prices trend 79
10.7 K-nearest neighbour regression 80
10.8 Support vector regression prediction 80
10.9 Long short term memory prediction 81
10.10 Kernel ridge prediction 81
10.11 Kernel support vector prediction 82

A.1 Cross validation for one step ahead timeseries data [51] 95
A.2 Cross validation for m step ahead timeseries data [51] 95
A.3 importance of feature scaling 96

List of Tables

3.1 Kernel types 21

7.1 Pinball loss Melbourne data 63
7.2 Pinball loss quantile-wise Melbourne data 64
7.3 Mean absolute error Melbourne data 64

10.1 Root mean squared errors 82
10.2 Mean absolute errors 82
10.3 Mean absolute percentage errors 83

A.1 Pinball loss Boston housing data 90
A.2 Pinball loss quantile-wise Boston data 90
A.3 Mean absolute error Boston data 90
A.4 Pinball loss Abalone data 91
A.5 Pinball loss quantile-wise Abalone data 91
A.6 Mean absolute error Abalone data 91

A.7 Pinball loss Vehicle data 92
A.8 Pinball loss quantile-wise Vehicle data 92
A.9 Mean absolute error Vehicle data 92

Chapter 1

Problem Description

Individuals and organizations constantly face situations of uncertainty thus the need of robust forecasting methods. Such methods are crucial in the process of taking informed decision and for strategic planning.
The basic idea of forecasting is that we can extract knowledge from the past in order to make educated guesses about the future. Consequently, the range of fields where forecasting can be applied is very wide. In this thesis, our focus lies on applying forecasting techniques to the energy sector.
Our decision to focus on the energy market is mainly motivated by the rapid changes it has undergone. Over the last decades, electricity markets have gone through an unprecedented transformation. This shift was driven by the liberalization of such markets, the development and integration of renewable energy sources, the increase of low carbon technologies and the adoption of smart meters. Events like the California electricity crisis are further motivating the choice of the electricity sector as subject of our studies, see [13]. Moreover, the process of deregulation lead to an increasing interest in the field of electricity forecasting (EF) within the academic community, see 2.2. In addtion, the United Nations have identified the right to access affordable, reliable, sustainable and modern energy as one of their 17 SDGs [6]. Finally, the electricity market has a set of features that make it unique: electricity cannot be stored in an efficient way and supply and demand have to be matched instantly.

1.1 Motivation

There are mutliple reasons why the energy sector needs robust forecasting techniques. For power market companies, being able to predict prices with a low mean absolute percentage error (MAPE) 4.3 results in increased savings [87]. Furthermore, the adoption of smart meters provides power market

companies with a huge amount of consumer data. This can enable them to better model consumer preferences.
Transmission system operators' (TSO) main goal is to match supply and demand, generally TSO do so by increasing or decreasing the generation. Thus, from their point of view forecasting is critical for balancing the electricity netwrok. Probabilistic forecasting may be useful to power producers, traders and consumers in order to improve their decision making process and managing risk. This holds in particular for traders, because probabilistic forecasts enables them to simulate scenarios and carry out stress tests. Other possible applications include: control of storage, demand side response, anomaly detection, network design and planning, simulating inputs and handling missing data.

1.2 Point versus probabilistic forecast

A distinction has to be made between two types of forecasting approaches: point forecasts and probabilistic forecasts. Point prediction, also called deterministic forecasting in the literature [98], is all about predicting a particular value in time. On the other hand, with probabilistic forecasting we aim at predicting either a prediction interval, quantiles or a probability distribution for each point in time [73]. For this reason, probabilistic forecasts are more informative than point forecasts. This is why the interest of the research community is shifting towards them. A probabilistic forecast can be turned into a point forecast by simply taking its expecatation. Alternatively, a probabilistic forecast can be derived from a point one by modeling the residuals of the point prediction.

1.3 Aims and objectives

The scope of the thesis is analyzing state of the art forecasting methods in the energy market and to compare and to integrate them with ideas coming from the theory of kernel methods.

1.4 Outline

We start with a literature review and a bibliometric analysis in Section 2. Then, the theory underlying kernel methods is covered in Section 3. Evaluation metrics necessary to rank the forecasting techniques are presented in Section 4. Section 5 explains the core features and terminology of the energy market and of the electricity newtork. Following, Sections 6 and 7 introduce the state of the art methods in the context of point and probabilistic forecasting respectively. Section 8 goes on with the etract-load-transform (ETL) pipeline

and the exploratory analysis (EDA). Implementation details are included in 9. Finally, section 10 presents the experiments, the results and discusses models' strenghts, weaknesses and possible improvements.

Chapter 2

Literature Review

During the past 25 years a wide range of new ideas have been proposed for electricity point forecasting and for probabilistic forecasting.
The field benefitted greatly from the increse of computing power, the greater availability of data and the interest in data science. As a consequence, the forecaster's toolbox has grown in size and complexity.
Before delving into the literature review, we stress that at this point in time there is no superior method. Different solutions may outperform or underperform compared to other techniques depending on the problem settings. Thus, understanding the complexity, strenghts and weaknesses of each method is crucial for fitting the right model to the right setting.
Within this research community, it emerged the need for more homogeneity in the choice of the error valuation metrics (Section 4), data quality and in the way of comparing model performances [98]. As a solution, [61] proposes a checklist to aid evaluating the meaningfulness of new research. Throughout this thesis work, we will stick to the proposed principles and best practices peculiar of the EF field.

2.1 Electricity forecasting classification

Electricity forecasting is a vague term and it is used in the literature to refer to the whole field. Thus, in order to introduce some clarity it is useful to classify the range of EF articles in terms of their core attributes.
In the context of energy forecasting, the quantities of most interest are electricity prices (EPF), electricity loads (ELF) and renewables generation (mostly wind and solar).
In terms of forecasting horizons, we can group EF into four major categories: very short term forcasting (VSTF), short term load forecasting (STF), medium term forecasting (MTF) and long term forecasting (LTF). Consensus in the literature is to use one day, two weeks and three years respectilvely [45] as

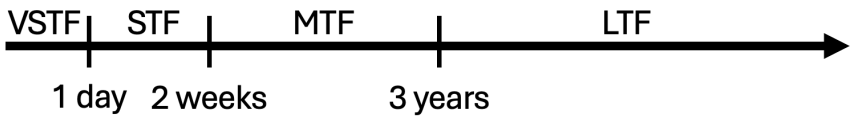


Figure 2.1: Time classsification [46]

cut off horizons; see 2.1 for a visualisation.
Forecasts can either be for the whole target electricity network (system) or for a subset of it (zonal).
EF literature distinguishes between point and probabilistic forecasts. Each of the two has its advantages and disadvantages. Point forecasts are easier to generate and less computationally intensive while probabilistic forecast are more informative. Industry and research efforts have focused primarily on point forcasting. Nevertheless, interest in probabilistic forecasting has risen considerably over the last years due to renewable integration requirements, introduction of smart grids and increased market competitivness.

2.2 Bibliographic analysis

This section presents the results of the bibliometric analysis we performed on March 6th 2024. This survey has been carried out by using the Scopus citation database. For details on the specific queries entered in Scopus, refer to Subsection 5 of the Appendix A.5.
To get started, let us consider the evolution of the EF field over the years. This is visualized in Figure 2.2, with results grouped by category. Articles prior to the 2000 have been aggregated together due to their small number. Figure 2.2 shows the trend of an increasing interest in EF.
The next question is to compare the state of point versus probabilistic forecasting, this is visualized in Figure 2.3. What can be concluded is that probabilistic is less developed than point forecasting. To our mind this is due to the complexity of probabilistic forecasts. Nevertheless, we can see a trend that suggests researchers are making an effort to fill this gap.
The EF literature is dominated by statistical and computational intelligence (CI) methods as can be seen from Figure 2.4, with CI methods being slightly preferred.
EF is a heterogenous field of research, its researchers come from a wide array of backgrounds, with electrical engineers and statisticians making up the top contributors; their different educational training may explain why the split between statistical and computational intelligence methods is so marked. Figure 2.5 depitcs the EF publications by subject area. What can be concluded, is that the bulk of pubblication come from engineering, computer

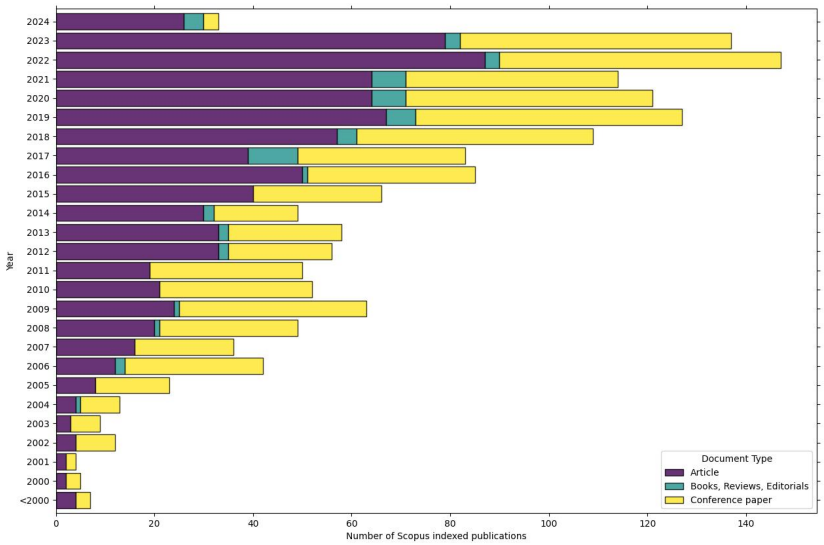


Figure 2.2: Electricity forecasting publications over the past years

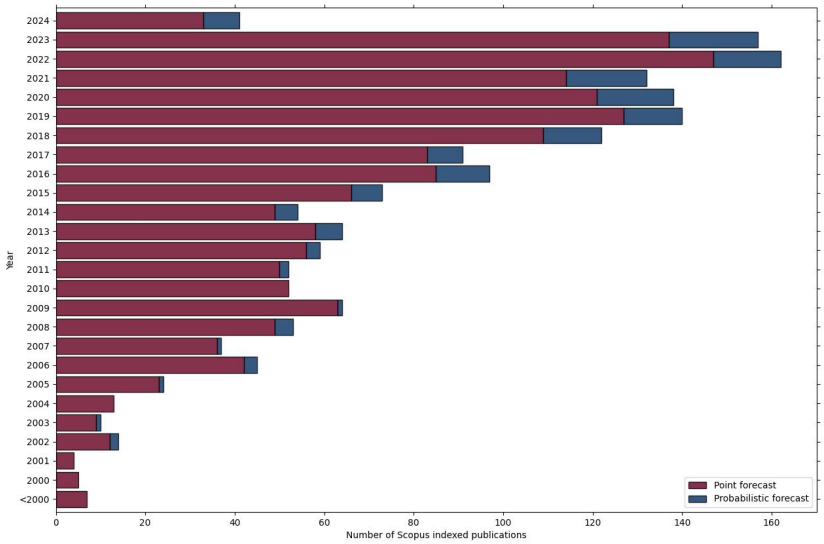


Figure 2.3: Point versus probabilistic pubblications over the past years

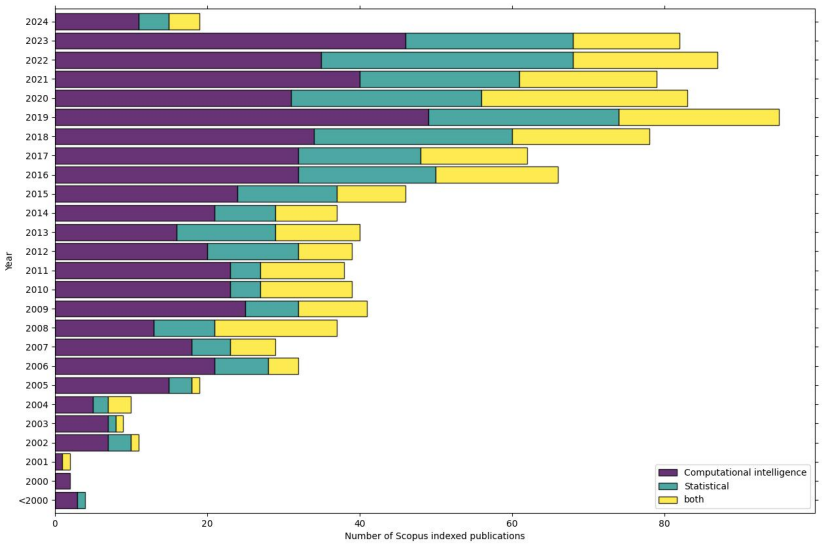


Figure 2.4: Pubblications by method over the past years

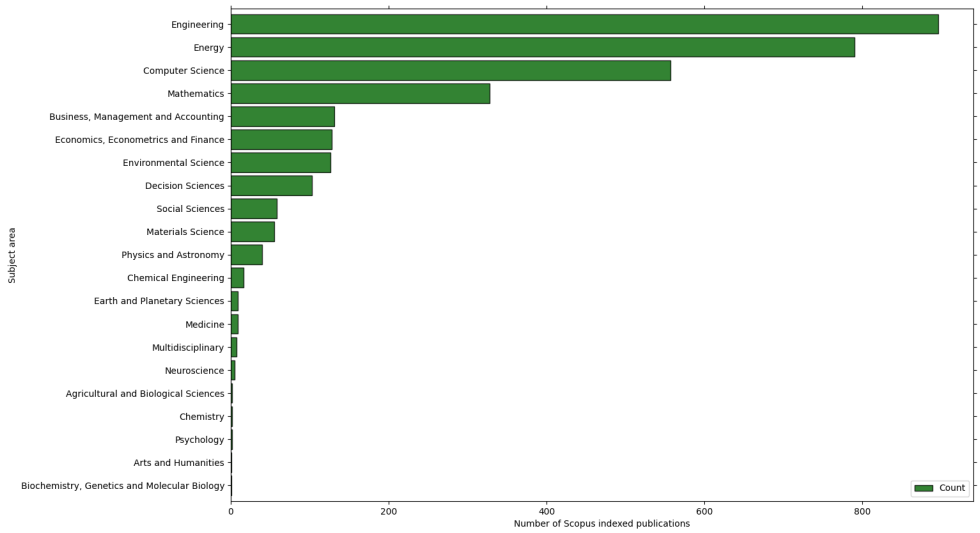


Figure 2.5: Pubblications by subject area

science, mathematics and econometrics.
In order to refer to the most relevant source in the field, EF outlets have been ranked by popularity and plotted in Figure 2.6.

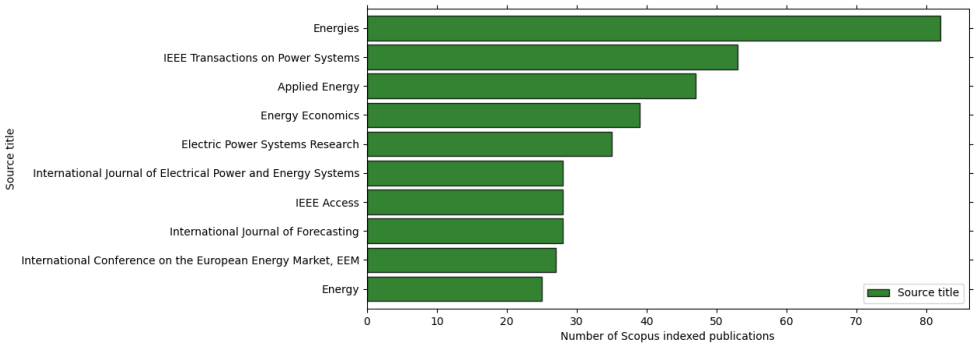


Figure 2.6: Most popolar sources/outlets

2.3 Electricity forecasting literature review

To get started a few review articles were collected in order to understand conventions, best practicies and terminology of the EF community. Weron [98] reviews the state of the art for electricity price forecasting. Besides analysing complexity of available solutions, strenghts and weaknesses it also stresses the need for objective comparative EPF studies. Specifically, it advocates for studies using similar datasets, using the same error evaluation metric and statistical testing model's outperformance. Hong et al. [46] discusses the state of the art in probabilistic electric load forecasting. It differentiates between techniques and methodologies. With techniques they refer to a family of models, like multiple linear regression or artificial neural networks. On the other hand, methodologies consist of general frameworks that can be incorporated into any method, for example variable selection mechanisms. Also this paper stresses the need for some guidelines to standardize research in the field. Nowotarski et al. [73] carried out a review of probabilistic forecasting. Weron et al. [61] offer a set of best practices when forecasting electricity prices in order to have a common framework to evaluate and compare future research. Zhang et al. [104] considers state of the art methods in wind power probabilistic forecasting and describes current challenges and possible future developments. Ziel et al. [107] provides detailed tables, groups research papers by the time dimension and objective and reports dataset, model and accuracy measures adopted. David et al. [23] adopt a combination of autoregressive moving average (ARMA) and generalized autoregressive conditional heteroskedasticity (GARCH) in probabilistic forecasts of solar irradiance. Furthermore, they propose a recursive framework for parameter estimation. De Gooijer [24] reviews 25 years of time series forecasting for the period 1990-2005, highlighting the most influential works. He et al. [41] models multistep wind speed probabilistic forecasting by mixing complementary ensemble empirical mode decomposition(CEEDMAN), least absolute shrinkage and selection operator (LASSO) and quantile regression (QR). In this

work, CEEMDAN is used to decompose the wind speed time series, LASSO compresses high dimensional features, QR is used for obtaining quantile forecasts, finally kernel density estimation (KDE) converts quantile forecasts into density estimates. Wan et al. [95] proposes a combination of QR and extreme learning machine (ELM) to generate non parametric probabilistic forecasts of wind generation. Zhang et al. [105] forecasts wind speed by adopting QRMGM, which is a combination of QR with a minimal gated memory network. Hyndman et al. [52] explains kernel estimator of conditional density, analyses its asymptotic behaviour and covers an application for the daily temperatures of Melbourne. Kaur et al. [57] carries out a comparative study of techniques spanning ideas from statistic and artificial intelligence. Van der Meer et al. [90] provides another thorough anlysis of the probabilistic forecasting realm by covering recent advances and identifying research gaps. The IEEE Power and Energy Society provides also insightful lecture notes on probabilistic energy forecasting methodologies, implementations, and applications [2]. Marcjasz et al. [64] uses distributional neural networks to create probabilistic forecasts for the day ahead electricity prices in the German market. Nowotarski et al. [72] introduce a method for constructing prediction intervals (PI) and call it quantile regression averaging. Their idea is weighting a set of model predictions such that the pinball loss of the weighted model is minimized. The observed result is that QRA performed better compared to twelve individual models. Arora et al. [11] focus on modelling electricity smart meter data by proposing a non parametric probabilistic technique based on kernel density estimation and conditional kernel density estimation [78, 52]. Their conclusion is that kernel density methods are competitive against exponential smoothing when forecasting residential data. Conversely, exponential smoothing has still an edge in predicting small to medium-sized enterprises data (SMEs). Zhang et al. [103] introduce a framework based on quantile regression and kernel density estimation in the context of short term wind forecasting. The proposed methods behave well compared to an autoregressive ARMA(1,1) model. Haben et al. [37] analyses a variety of techniques in terms of both probabilistic and point forecasting. Within this study, they focus on load forecasting at the low voltage level. Koochali et al. [60] reviews various existing methods for assessing probabilistic forecast models and discusses their advantages and disadvantages. Matheson et al. [65] develops classes of scoring rules for continuous probability distributions. Gneiting et al. [33] provides a thorough overview of the theory of scoring rules for interval and density forecasts. Gneiting et al. [31] covers theory and state of the art techniques in probabilistic forecasting. Zhang et al [102] proposes a two stage bootstrap sampling framework for probabilistic load forecasting. They test it for different regression models such as random forest (RF), gradient boosting regression tree (GBRT), linear regression, and least squares support vector regression (LSSVM). Jónsson et al. [55] introduces a density model for the day ahead market extending the adaptive

QR framework of [68] by modelling the tails of the predicted density with an exponential distribution. Fatema et al. [27] considers Gaussian process regression for point forecasting and prediction intervals prediction. Then, it inputs prediction intervals to kernel density estimation in order to estimate a probability distribution. Dudek [26] proposes a probabilistic forecasting model based on the Nadaraya Watson estimator [71, 96]. Huurman et al. [50] surveys the predictive power of weather variables for electricity prices in the Danish market. Their empirical results suggest that weather is central for point forecasting day ahead prices. The opposite conclusion are drawn for density forecasting.

Lately, the idea of combining forecasts has gained popularity in the forecasting community [75]. In the literature, combined forecasts are called ensemble [32]. Experimental results have shown ensemble methods to outperform their component forecasts. Note that, the more the errors of the combined models are not correlated, the more we can benefit from ensembles. It is also worth noting that older and simpler methods are still valuable (in combination with other models or on their own). These being less subject to overfitting than complex models.

A major step forward in EF was the creation of the global energy forecasting competition (GEFCom) in 2012. Until then, no formal benchmarking process or data pool was established and new publications rarely reproduced the results from work done by others. Addressing these issues was the motivation behind the creation of GEFCom by the IEEE working group on energy forecasting. The EF field was positively affected by this competition. A number of ideas were tested on the same setting with only the best ones being published and it also contributed bridging the gap between industry practice and academic research. GEFCom 2012 had two tracks; the former about hierarchical load forecasting, the latter about wind power forecasting, see [47] for a comprehensive review.

The focus of GEFCom 2014 was on probabilistic forecasting, Hong et al. [48] discusses the problem tracks, the data and the winning methods. In this paragraph some of the winning entries of the 2014 GEFCom edition are discussed. Xie et al. [101] propose a two stage approach; in the first stage they use multiple linear regression (MLR) to build a point forecast, then in the second stage they try different approaches for modelling the MLR residuals, among other they tried exponential smoothing (ESM), artificial neural networks (ANN) and autoregressive integrated moving average (ARIMA). Maciejowska et al. [62] proposes a new probabilistic model extending the idea of quantile regression averaging (QRA) [72]. Haben et al. [36] mixes conditional kernel density (CKD) and quantile regression in their competition entry. Gaillard et al. [29, 30] combines quantile regression with generalized additive models [40]. Ziel et al. [106] estimates an AR model through the LASSO [89] instead of the standard OLS.

The last GEFCom was held in 2017, its focus was providing probabilistic

load forecasts, see [49]. The GEFCom competition has also inspired the organization of other competitions such as the RWEnpower competition in the UK, the RTE competition in France, the Tokyo electric power company competition in Japan and the BigDEAL forecasting competitions.

A couple of considerations can be drawn from the above literature review. The EF field is characterized by heterogeneity in its forecasting techniques; methods come from statisitics, mathematics, econometrics, electrical engineering and the artificial intelligence communities.
Every paper uses different datasets. Therefore, it is not possible to compare directly results from one paper to another without implementing the paper specific algorithms and then applying them to the respective dataset. An additional hurdle is that some datasets are not freely accessible.
Thus, a good understanding of state of the art methods in EF is required to carry out a rigorous comparison between methods. That is why the following chapters are devoted to summarising the mathematical theory underlying such techniques.

2.4 Kernel methods literature review

Kernel methods are a class of algorithms for pattern analysis. With kernel methods we are able to apply linear methods with predictors in a high dimensional space, without having to explicitly evaluate the involved dot products of the features. Throughout this thesis work, we will address the performance of kernel methods in the context of EF.
Their name comes from the German word kern, which translates to core in english. Such term was first used by David Hilbert in his paper on integral equations [42] where he introduced the term "definite kernels". Following, Hilbert's and Schmidt's work [79] lead to the introduction of a new space, the Hilbert space. In 1909, James Mercer improved Hilbert's work by proposing his theorem [67]. This theorem underlies the power of kernel methods, that is the kernel trick. In 1938, Schoenberg [80] developed the mathematical results that allow us to find the kernel associated to a specific feature space metric. In 1941, Kolmogorov [59] carried out stuides on representing kernel in linear spaces. In 1950, Aronszajn [10] published the first work on reproducing kernel hilbert spaces (RKHS); developing the general method for representing kernels in linear spaces. In 1964, Aizerman [9] further improved the theory of RKHS. It was in the nineties that theory of kernel methods got popular, particularly in the field of machine learning. Kernels have been used in various different tasks such as SVM [92] [91], Gaussian process classifiers [99], spline methods [94], neural networks [76] and principal component analysis [81]. Nevertheless kernel methods received very little attention in the specific setting of EF literature.

The kernel theory needed for this thesis work is covered in Section 3. For an introduction to kernel methods, we refer to [83, 43].
Kanagawa et Fukumizu introduces to the concept of kernel mean embedding [56]. Muandet et al. [69] surveys established results and new advances in the theory of Hilbert space distribution embeddings. It has to be said that, computing and storing such embeddings becomes prohibitive for large scale settings. Rudi et al. [20] proposes an efficient approximation procedure based on the Nyström method [74], providing also an upper bound for the approximation error. Smola et al. [21] presents kernel herding; basically, the authores used the kernel trick to extend the herding algorithm to continous spaces. The result is an infinite memory deterministic process that takes in a collection of samples and learns to approximate a probability density function (PDF).

Chapter 3

Kernel Theory

The following section convers the theory of kernel methods. To get started, the building blocks of kernel theory will be introduced. After that, a couple of relevant articles in the kernel community will be reviwed and summarized.

3.1 Kernel mean embedding of distributions: A review and beyond

From this first paper [69], the notation and terms used in the theory of Reproducing Kernel Hilbert Spaces are summarized.
Many algorithms use the inner product as similarity measure between data instances $x, x' \in \mathcal{X}$. However, this inner product spans only the class of linear similarity measures.
The idea behind kernel methods is to apply a non-linear transformation φ to the data x in order to get a more powerful non linear similarity measure.

$$\begin{aligned} \varphi(x) : \mathcal{X} &\longrightarrow \mathcal{F} \\ x &\mapsto \varphi(x) \end{aligned}$$

Then we take the inner product in the high dimensional space $\mathcal{F}$ mapped by $\varphi(x)$.

$$k(x, x') := \langle \varphi(x), \varphi(x') \rangle_{\mathcal{F}}$$

$\varphi(x)$ is referred as feature map while k as kernel function.
Therefore, we can kernelize any algorithm involving a dot product by substituting $\langle x, x' \rangle_{\mathcal{X}}$ with $\langle \varphi(x), \varphi(x') \rangle_{\mathcal{F}}$
One would expect constructing the feature maps explicitly and then evaluate

their inner product in $\mathcal{F}$ to be computationally expensive, and indeed it is. However, we do not have to explicitly perform such calculations. This is because of the existence of the kernel trick. To illustrate the idea behind the kernel trick consider the following example.
Suppose $x \in \mathbb{R}^2$ and assume to select $\varphi(x) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)$, then the inner product in the feature space is $x_1^2x_1'^2, x_2^2x_2'^2 + 2x_1x_2x_1'x_2'$. Notice that this is the same of $\langle \varphi(x), \varphi(x') \rangle$; thus the kernel trick consists of just using $k(x, x') =: (x^T x')^2$.

3.1.1 RKHS

Following are the definitions that make up the basis for the theory of kernel methods.

Definition 3.1 *A sequence $\{v_n\}_{n=1}^{\infty}$ of elements of a normed space $\mathcal{V}$ is a Cauchy sequence if for every $\varepsilon > 0$, there exist $N = N(\varepsilon) \in \mathbf{N}$ such that $\|v_n - v_m\|_{\mathcal{V}} < \varepsilon \ \forall m, n \geq N$*

Definition 3.2 *A complete metric space is a metric space in which every Cauchy sequence is convergent.*

Definition 3.3 *A Hilbert space is a vector space $\mathcal{H}$ with an inner product $\langle f, g \rangle$ such that the norm defined by $\|f\| = \sqrt{\langle f, f \rangle}$ turns $\mathcal{H}$ into a complete metric space.*

Definition 3.4 *RKHS. A Reproducing Kernel Hilbert Space is an Hilbert space with the evaluation functionals $\mathcal{F}_x(f) := f(x)$ bounded, i.e. $\forall x \in \mathcal{X}$ there exists some $C > 0$ such that $\|\mathcal{F}_x(f)\| = \|f(x)\| \leq C\|f\|_{\mathcal{H}} \ \forall f \in \mathcal{H}$*

Theorem 3.5 *Riesz Representation. If $A : \mathcal{H} \to \mathbf{R}$ is a bounded linear operator in a Hilbert space $\mathcal{H}$, there exists some $g_A \in \mathcal{H}$ such that $A(f) = \langle f, g_A \rangle_{\mathcal{H}}$, $\forall f \in \mathcal{H}$.*

The Riesz representation theorem results in the following proposition for RKHS.

Proposition 3.6 *For each $x \in \mathcal{X}$ there exists a function $k_x \in \mathcal{H}$ such that $\mathcal{F}_x(f) = \langle k_x, f \rangle_{\mathcal{H}} = f(x)$*

The function k_x is the reproducing kernel for the point x. Furthermore, note that k_x is itself a function lying on $\mathcal{X}$

$$k_x(y) = \mathcal{F}_y(k_x) = \langle k_x, k_y \rangle_{\mathcal{H} = \langle \varphi(x), \varphi(x) \rangle_{\mathcal{H}}}$$

3.1.2 Kernel families

Table 3.1 contains popular kernel families in literature and applications.

Table 3.1: Kernel types

Kernel function	Equation	Hyperparameters
Linear	$k(x_1, x_2) = x_1 x_2$	
Polynomial	$k(x_1, x_2) = (x_1^\intercal x_2 + c)^d$	c, d
Gaussian RBF	$k(x_1, x_2) = e^{-\frac{\|x_1 - x_2\|^2}{2\sigma^2}}$	σ
Exponential RBF/Laplacian	$k(x_1, x_2) = e^{-\frac{\|x_1 - x_2\|}{\gamma}}$	γ
Hyperbolic/Sigmoid Kernel	$k(x_1, x_2) = \tanh(\gamma x_1^\intercal x_2 + r)$	γ, r
Periodic	$k(x_1, x_2) = e^{\frac{-2sin^2\left(\frac{\pi}{p}\|x_1 - x_2\|\right)}{l^2}}$	p, l

3.2 Recovering distributions from Gaussian RKHS embeddings

This paper covers the RKHS embedding approach to nonparametric statistical inference [56]. The idea here is computing an estimate of the kernel mean in order to obtain an approximation of the underlying distribution of the observed random variable. The kernel mean embedding $\mu_{\mathbb{P}}$ of a probability $\mathbb{P}$ corresponds to the feauture map $\varphi(x)$ integrated with respect to the $\mathbb{P}$ measure.
That is $\mu_{\mathbb{P}} := \int_{\mathcal{X}} k(x, \cdot) d\mathbb{P}(x)$.
Kernel mean embedding serves as a unique representation of $\mathbb{P}$ in the RKHS $\mathcal{H}$. This holds provided that $\mathcal{H}$ is characteristic.

Definition 3.7 *The RKHS $\mathcal{H}$ and the associated kernel k are said characteristic, when the mapping $\mu : \mathbb{P} \to \mathcal{H}$ is injective.*

When the mapping is injective, we have that $\mu_{\mathbb{P}}$ is uniquely associated with $\mathbb{P}$; thus, $\mu_{\mathbb{P}}$ is a unique representation of $\mathbb{P}$ in $\mathcal{H}$.
Note that, by the reproducing property of RKHS $\langle f, k(x, \cdot)\rangle = f(x)$ we have:

$$E_{\mathbb{P}}[f(x)] = \langle f, \mu_{\mathbb{P}}\rangle_{\mathcal{H}}, \ \forall f \in \mathcal{H}$$

Proof

$$\begin{aligned} E_{\mathbb{P}}[f(x)] &= \int_{\mathcal{X}} f(x) d\mathbb{P}(x) \\ &= \int_{\mathcal{X}} \langle f, k(x, \cdot) \rangle_{\mathcal{H}} d\mathbb{P}(x) \\ &= \sum_{i=1}^{\infty} \langle f, k(x_i, \cdot) \rangle_{\mathcal{H}} \mathbb{P}(\mathcal{X}_i) \\ &= \langle f, \sum_{i=1}^{\infty} k(x_i, \cdot) \mathbb{P}(\mathcal{X}_i) \rangle_{\mathcal{H}} \\ &= \langle f, \mu_{\mathbb{P}} \rangle_{\mathcal{H}} \end{aligned}$$ □

The kernel mean embedding can be employed to estimate the density p at any fixed point x_0. Letting δ_{x_0} to be the dirac delta function we have:

$$p(x_0) = \int \delta_{x_0} p(x) dx = E_{\mathbb{P}}[\delta_{x_0}]$$

Therefore, the idea is to define an estimator for the expectation of δ_{x_0} through $\mu_{\mathbb{P}}$; this would result in an estimator of $p(x_0)$.
A kernel $k(x_0, \cdot)$ is used to approximate the delta function, furthermore applying theorem 1 of [56] we have that a consistent estimator of $E_{\mathbb{P}}[k(x_0, \cdot)]$ is given by $\sum_{i=1}^{n} w_i k(x_0, x_i)$
When the weigths are all $1/n$ we end up with the standard kernel density estimation.
Alternatively, the optimal weights can be found by minimizing the following problem $\|\hat{\mu} - \Phi w\|^2$ where $\Phi : \mathbb{R}^n \to \mathcal{H}$.

3.3 Super samples from kernel herding

Kernel herding is a deterministic sampling algorithm designed to draw "Super Samples" from probability distributions [21]. The idea of herding, is to generate pseudo-samples that greedily minimize the error betweeen the mean operator and the empirical mean operator resulting from the selected herding points.

Letting $p(x)$ be a probability distribution, kernel herding is recursively defined as follows:

$$\begin{aligned} x_{t+1} &= \arg\max_{x \forall \mathcal{X}} \langle w_t, \varphi(x) \rangle \\ w_{t+1} &= w_t + E_{\mathbb{P}}[\varphi(x)] - \varphi(x_{t+1}) \end{aligned}$$

w denotes a weight vector that lies in $\mathcal{H}$.
Here, by assuming that the inner product between weights and the mean operator is equal to a general functional f evaluated at x, that is $\langle w, \varphi(x)\rangle_{\mathcal{H}} = f(x)$. We have:

$$\begin{aligned}
\langle w, \mu_{\mathbb{P}}\rangle_{\mathcal{H}} &= \langle w, \int k(x,\cdot) d\mathbb{P}(x)\rangle_{\mathcal{H}} \\
&= \langle w, \sum_{i=1}^{\infty} k(x_i,\cdot)\mathbb{P}(\mathcal{X}_i)\rangle_{\mathcal{H}} \\
&= \sum_{i=1}^{\infty} \langle w, k(x_i,\cdot)\rangle_{\mathcal{H}} \mathbb{P}(\mathcal{X}_i) \\
&= \sum_{i=1}^{\infty} f(x_i)\mathbb{P}(\mathcal{X}_i) \\
&= \int f(x) d\mathbb{P}(x) \\
&= \mathbb{E}_{\mathbb{P}}[f(x)]
\end{aligned}$$

Moreover, second assumption of the model is that $\|\varphi(x)\|_{\mathcal{H}} = R \quad \forall x \in X$. That is the Hilbert space norm of the feature vector is equal to a constant R for all states in the set $\mathcal{X}$.

This can be achieved by taking the new feature vector as $\varphi^{new}(x) = \frac{\varphi(x)}{\|\varphi(x)\|_{\mathcal{H}}}$. See A.1 for details.
By rewriting the formula for the weights we end up with

$$\begin{aligned}
w_{t+1} &= w_t + E_{\mathbb{P}}[\varphi(x)] - \varphi(x_{t+1}) \\
&= w_{t-1} + E_{\mathbb{P}}[\varphi(x)] + E_{\mathbb{P}}[\varphi(x)] - \varphi(x_{t+1}) - \varphi(x_t) \\
&= w_{t-2} + E_{\mathbb{P}}[\varphi(x)] + E_{\mathbb{P}}[\varphi(x)] + E_{\mathbb{P}}[\varphi(x)] - \varphi(x_{t+1}) - \varphi(x_t) - \varphi(x_{t-1})
\end{aligned}$$

Considering w_T, we have

$$w_T = w_0 + TE_{\mathbb{P}}[\varphi(x)] - \sum_{t=1}^{T} \varphi(x_t)$$

Note that $E_{\mathbb{P}}[\varphi(x)] = \int_{\mathcal{X}} \varphi(x) d\mathbb{P}(x)$ which corresponds to the definition of $\mu_{\mathbb{P}}$. That is μ is the mean operator associated with the distribution $\mathbb{P}$; it lies in $\mathcal{H}$.
Thus,

$$w_T = w_0 + T\mu_{\mathbb{P}} - \sum_{t=1}^{T} \varphi(x_t)$$

Notice we do not have to compute $\mu_{\mathbb{P}}$ explicitly, the terms involving $\mu_{\mathbb{P}}$ will

be computed by applying the kernel trick.

Now we have everything we need in order to reformulate the original problem in a way such that it depends just on the states x. Plug the formula for the weights in the formula for the x_t and use the kernel trick; we end up with

$$\begin{aligned} x_{T+1} &= \arg\max_{x \forall \mathcal{X}} \langle w_0 + T\mu_{\mathbb{P}} - \sum_{t=1}^{T} \varphi(x_t), \varphi(x) \rangle_{\mathcal{H}} \\ &= \arg\max_{x \forall \mathcal{X}} \langle w_0, \varphi(x) \rangle_{\mathcal{H}} + \langle T\mu_{\mathbb{P}}, \varphi(x) \rangle_{\mathcal{H}} - \langle \sum_{t=1}^{T} \varphi(x_t), \varphi(x) \rangle_{\mathcal{H}} \\ &= \arg\max_{x \forall \mathcal{X}} \langle w_0, \varphi(x) \rangle_{\mathcal{H}} + T\langle \mu_{\mathbb{P}}, \varphi(x) \rangle_{\mathcal{H}} - \sum_{t=1}^{T} k(x_t, x) \end{aligned}$$

Notice $\langle \mu_{\mathbb{P}}, \varphi(x) \rangle_{\mathcal{H}}$ can be rewritten in the following way

$$\begin{aligned} \langle \mu_{\mathbb{P}}, \varphi(x) \rangle_{\mathcal{H}} &= \langle \int_{\mathcal{X}'} \varphi(x') d\mathbb{P}(x'), \varphi(x) \rangle_{\mathcal{H}} \\ &= \langle \sum_{i=1}^{\infty} \varphi(x'_i) \mathbb{P}(\mathcal{X}'_i), \varphi(x) \rangle_{\mathcal{H}} \\ &= \sum_{i=1}^{\infty} \langle \varphi(x'_i), \varphi(x) \rangle_{\mathcal{H}} \mathbb{P}(\mathcal{X}'_i) \\ &= \sum_{i=1}^{\infty} k(x'_i, x) \mathbb{P}(\mathcal{X}'_i) \\ &= \int_{\mathcal{X}'} k(x', x) d\mathbb{P}(x') \\ &= E_{\mathbb{P}}[k(x', x)] \end{aligned}$$

Furthermore, by initializing $w_0 = \mu_{\mathbb{P}}$ we end up with the following function to be optimized, i.e.

$$\begin{aligned} x_{T+1} &= \arg\max_{x \forall \mathcal{X}} \langle w_0, \varphi(x) \rangle + T\langle \mu_{\mathbb{P}}, \varphi(x) \rangle - \sum_{t=1}^{T} k(x_t, x) \\ &= \arg\max_{x \forall \mathcal{X}} (T+1) E_{\mathbb{P}}[k(x', x)] - \sum_{t=1}^{T} k(x_t, x) \end{aligned}$$

Now consider the error term between the mean kernel operator and its estimation through herding samples

$$
\begin{aligned}
\varepsilon_{T+1} &= \|\mu_{\mathbb{P}} - \frac{1}{T+1}\sum_{i=1}^{T+1}\varphi(x_t)\|_{\mathcal{H}}^2 \\
&= \mathbb{E}_{x,x'\sim\mathbb{P}}[k(x',x)] - \frac{2}{T+1}\sum_{t=1}^{T+1}\mathbb{E}_{x\sim\mathbb{P}}[k(x,x_t)] + \frac{1}{(T+1)^2}\sum_{t,t'=1}^{T+1}k(x_t,x_{t'}) \\
&= \mathbb{E}_{x,x'\sim\mathbb{P}}[k(x',x)] - \frac{2}{T+1}\sum_{t=1}^{T+1}\mathbb{E}_{x\sim\mathbb{P}}[k(x,x_t)] + \frac{1}{(T+1)^2}\sum_{\substack{t=1\\t=t'}}^{T+1}k(x_t,x_{t'}) + \\
&\quad + \frac{2}{(T+1)^2}\sum_{\substack{t=1\\t\neq t'}}^{T+1}k(x_t,x_{t'})
\end{aligned}
$$

So ε_{T+1} depends on x_{T+1} only through $-\frac{2}{T+1}\mathbb{E}_{x\sim\mathbb{P}}[k(x,x_{T+1})] + \frac{2}{(T+1)^2}\sum_{t=1}^{T}k(x_t,x_{T+1})$
The term $k(x_{T+1},x_{T+1})$ is not included, because by assumption it is equal to the constant R.
Recognize that this term is the negative of the objective function maximized with respect to x. So the sample x_{T+1} minimizes the error at time step $T+1$, i.e. ε_{T+1}
During the iterative step of kernel herding we maximize the negative of this quantity, thus we are minimizing the error greedily. In the sense that at each iteration we choose the x that minimizes our current error; however this does not guarantee that the samples states are jointly optimal.
Intuitively, at each iteration, herding searches for a new sample to add to the pool; it is attracted to the regions where p is high and pushed away from regions where samples have already been selected.

- Explain kernel density estimation Explain under which conditions kernel mean embedding is equivalent to kernel density estimation. Kernel mean embedding generalization of kernel density estimation

- Other kernel theory concepts, I may need to restructure the structure of the kernel folder by putting in the right order the varies paper1,2,3,4

Chapter 4

Evaluation metrics

Proper evaluation methods guide researchers in choosing the model that best fits their needs; thus, this chapter is dedicated to the most common evaluation metrics adopted by academics in the field of EF. Error metrics and measures vary depending on wethter we are concerned with point or probabilistic forecasts. Additionally, note that the latter can take different forms which therefore requires different measures.

4.1 Mean absolute error

Consider the time series with actual values given by $\mathrm{L} = (L_{n+1}, L_{n+2}, \ldots, L_{n+h})$ and its h step ahead point forecast $\hat{\mathrm{L}} = (\hat{L}_{n+1}, \hat{L}_{n+2}, \ldots, \hat{L}_{n+h})$ the mean absolute error (MAE) is defined as

Definition 4.1 $\mathrm{MAE}(\mathrm{L}, \hat{\mathrm{L}}) = \frac{1}{h}\|\mathrm{L} - \hat{\mathrm{L}}\|_1 = \frac{1}{h}\sum_{k=1}^{h} |L_{n+k} - \hat{L}_{n+k}|$

4.2 Root mean squared error

Definition 4.2 $\mathrm{RMSE}(\mathrm{L}, \hat{\mathrm{L}}) = \frac{1}{\sqrt{h}}\|\mathrm{L} - \hat{\mathrm{L}}\|_2 = \sqrt{\frac{\sum_{k=1}^{h} (L_{n+k} - \hat{L}_{n+k})^2}{h}}$

MAE and root mean squared error (RMSE) posses the useful property of being expressed in the same units of the data thus enabling meaningful comparisons. However, a drawback of such measures is that we cannot use them to compare accuracy between time series wich have different magnitudes. For instance, a day ahead error of 1kWh is negligible when considering a daily demand of 100kWn while the same error is considerably big when daily demand is 2kWh. This consideration leads to relative accuracy scores, between those the mean absolute percentage error (MAPE) is by far the most popular.

4.3 Mean absolute percentage error

.

Definition 4.3 $\text{MAPE}(\text{L}, \hat{\text{L}}) = \frac{100}{h} \sum_{k=1}^{h} \frac{|L_{n+k} - \hat{L}_{n+k}|}{|L_{n+k}|}$

4.4 Root mean squared percentage error

Definition 4.4 $\text{RMSPE}(\text{L}, \hat{\text{L}}) = 100 \cdot \sqrt{\frac{1}{h} \sum_{k=1}^{h} \left(\frac{|L_{n+k} - \hat{L}_{n+k}|}{|L_{n+k}|} \right)^2}$

Note, MAPE and root mean squared percentage error (RMSPE) may not be appropriate for series which have zero or very small values, for example, electricity demand at the household level; the result is a large score regardless of the absolute errors. Scaled errors constitute a robust family of scores.

4.5 Mean absolute scaled error

Definition 4.5 $\text{MASE}(\text{L}, \hat{\text{L}}) = \frac{1}{h} \sum_{k=1}^{N} \frac{|L_{n+k} - \hat{L}_{n+k}|}{\frac{1}{h-1} \sum_{k=2}^{h} |L_k - L_{k-1}|}$

In the denominator we have the error of the naïve/persistence model. In this model, the current demand makes up the prediction for the next time step; that is $\hat{\text{L}}_{n+1}^{\text{naive}} = \text{L}_n$.

4.6 Root mean squared scaled error

Definition 4.6 $\text{RMSSE}(\text{L}, \hat{\text{L}}) = \sqrt{\frac{1}{h} \sum_{k=1}^{N} \left(\frac{|L_{n+k} - \hat{L}_{n+k}|}{\frac{1}{h-1} \sum_{k=2}^{h} |L_k - L_{k-1}|} \right)^2}$

4.7 Pinball

The pinball score or quantile score is used to measure the accuracy of a quantile forecast.

Definition 4.7 $\text{Pinball}(L_t, \hat{L}_{t,q}, q) = \begin{cases} (1-q)(\hat{L}_{t,q} - L_t) & L_t < \hat{L}_{t,q} \\ q(L_t - \hat{L}_t, q) & L_t \geq \hat{L}_{t,q} \end{cases}$

The pinball loss is an asymmetric function, it weights its score differently depending on the error sign and on the quantile considered, see 4.1. By averaging all the pinball losses over all quantiles and over the whole forecast horizon, we obtain the pinball loss of the probabilistic forecast.

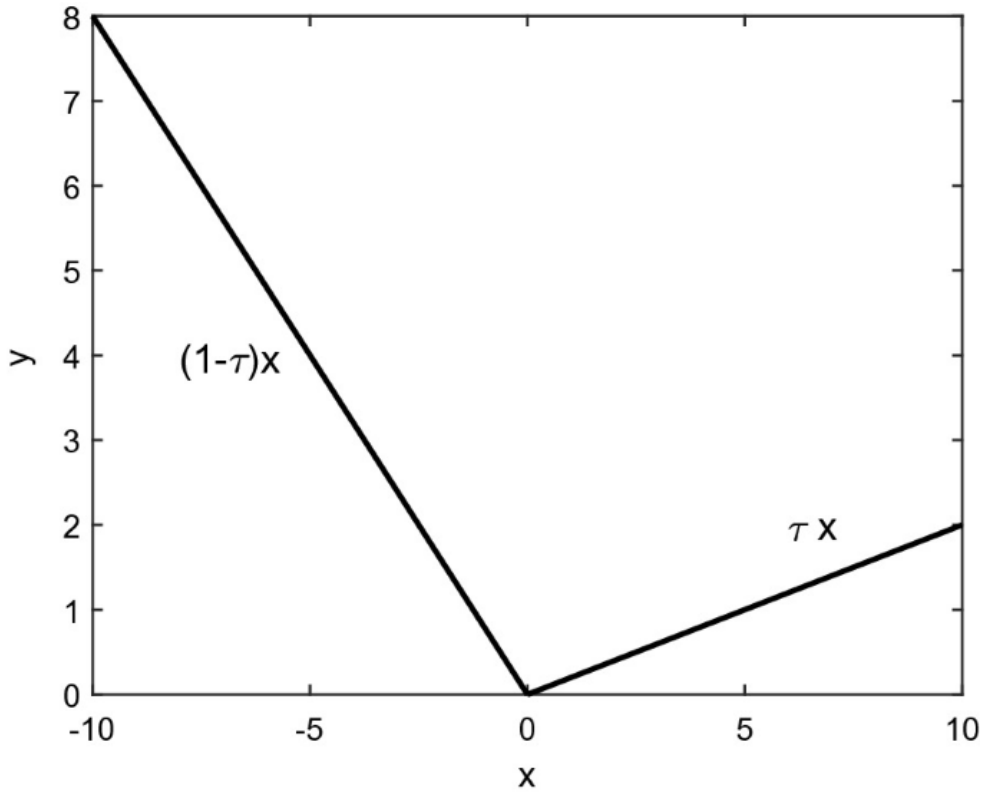


Figure 4.1: Pinball loss with $q = 0.2$ [38]

4.8 Winkler

Definition 4.8 $\text{Winkler}_\alpha = \begin{cases} \delta & Low_t \leq L_t \leq Uppt \\ \delta + 2(Low_t - L_t)/\alpha & L_t < Low_t \\ \delta + 2(Lt - Upp_t)/\alpha & L_t \geq Upp_t \end{cases}$

Where delta is the prediction interval (PI) width, that is $\delta = Upp_t - Low_t$. This score penalizes observations falling outside the PI and rewards narrow PI.

4.9 Continous ranked probability score

The continous ranked probability score (CRPS) measures the differece between the estimated cumulative distribution $\hat{F}$ and the empirical Cdf.

Definition 4.9 $\text{CRPS}(L, \hat{F}) = \int_{-\infty}^{\infty} \left(\hat{F}(x) - \mathbb{1}(x - L)\right)^2 dx$

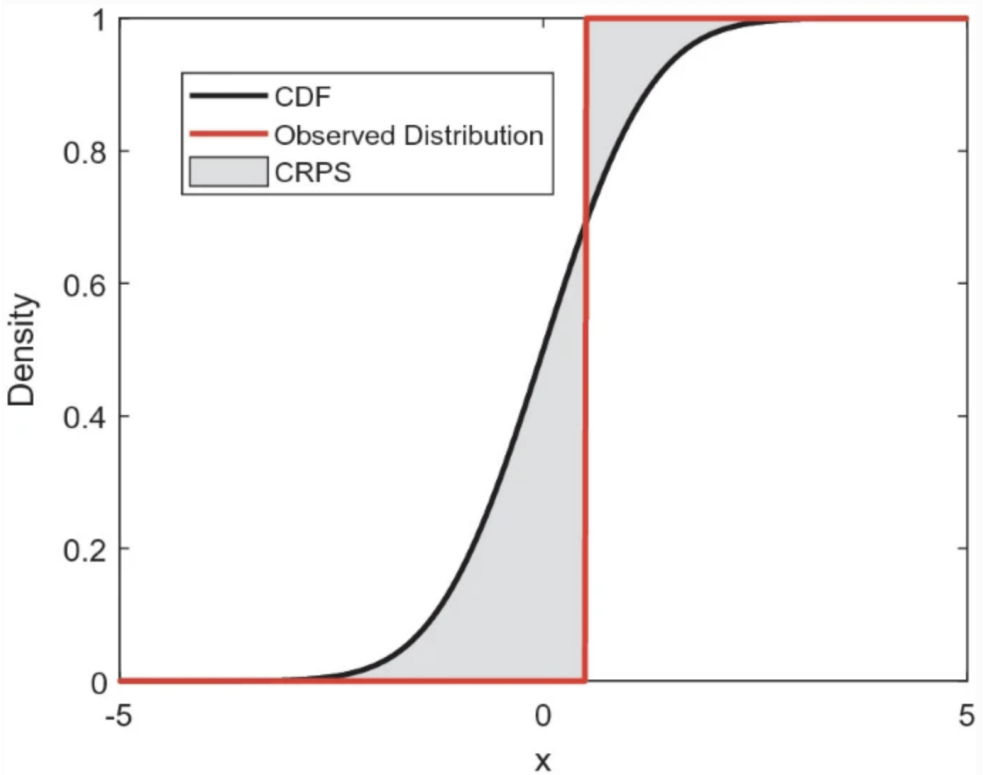


Figure 4.2: CRPS integral [38]

Nevertheless, we can evaluate the integral in closed form. Where the indicator function is defined as $\mathbb{1}(z) = \begin{cases} 0 & z < 0 \\ 1 & z \geq 0 \end{cases}$
For a visualisation see, 4.2. The grey area is what contributes toward the CRPS score. The better the estimated cdf is the smaller the total CRPS score will be.
It is worth noting, that the CRPS integral can be rewritten in terms of expectations. This makes its evaluation easier, since we know that the sample mean converges to the expectation by the law of large numbers. This was first pointed out by [34], where authors take advantage of lemma 2.2 of [12] or equivalently identity 17 of [85].

Lemma 4.10 *Let* X_1, X_2, Y_1, Y_2 *be independent real random variables with finite expectations. Let* X_1, X_2 *be identically distributed with distribution function* F *and*

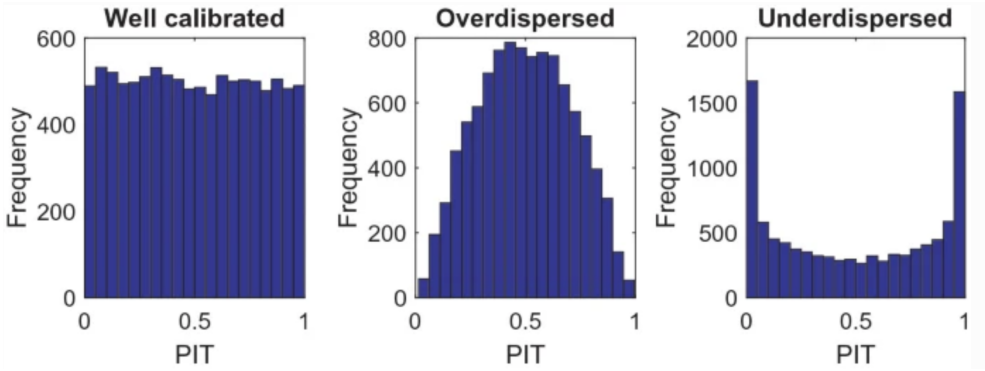


Figure 4.3: PIT types [38]

let Y_1, Y_2 *be identically distributed with distribution function* G. *Then*

$$\mathbb{E}(|X_1 - Y_1|) - \frac{1}{2}\mathbb{E}(|X_1 - X_2|) - \frac{1}{2}\mathbb{E}(|Y_1 - Y_2|) = \int_{-\infty}^{\infty} (F(x) - G(x))^2 \, dx \tag{4.1}$$

Notice that, in our case 4.9, the distribution G of Y_1 and Y_2 is degenerate, with all probability mass on a single point. It follows that, the third addend in the summation is zero. That is because the fact of Y_1 and Y_2 both following distribution G implies that $\mathbb{E}(|Y_1 - Y_2|)$ corresponds to the difference of two equal constant numbers.
Additionally, since Y_1 is just a constant, we have $Y_1 = L$.
Putting everything together we have obtained an alternative way of computing the CRPS score.

$$\int_{-\infty}^{\infty} \left(\hat{F}(x) - \mathbb{1}(x - L)\right)^2 dx = \mathbb{E}(|X_1 - L|) - \frac{1}{2}\mathbb{E}(|X_1 - X_2|) \tag{4.2}$$

4.10 Probability integral transform

The probability integral transform (PIT) is a method to assess visually the quality of probabilistic forecasts. PIT is obtained by applying a cdf F to your data; if applying such cdf to the data results in a uniform distributed PIT then F is a valid prediction. If not, F is not the suited cdf for the considered data. Figure 4.3 provides an example, applying the true cdf results in a well calibrated PIT (left). Alternatively, applying a bad cdf results in either a overdispersed (middle) or underdispersed (right) PIT.

Chapter 5

The Energy Market

This chapter has two intents. Firstly, it summarizes several concepts for the newcomers to the field of electricity markets. Secondly, it discusses trends and challenges undergoing in the power sector. Such developments motivate the need for robust and efficient forecasting techniques in the context of electricity markets.

5.1 Electricity distribution network

Electricity is generated by power plants which transfer it over the so called transmission level. Then, through the transmission network, this energy is transported at a high voltage over long distances. Finally, the voltage is reduced and the energy is moved into the distribution network. In a nutshell, transmision lines carry electric power from stations to substations while distribution lines carry electricity from substations to load points such as businesses, industries and homes, see figure 5.1.

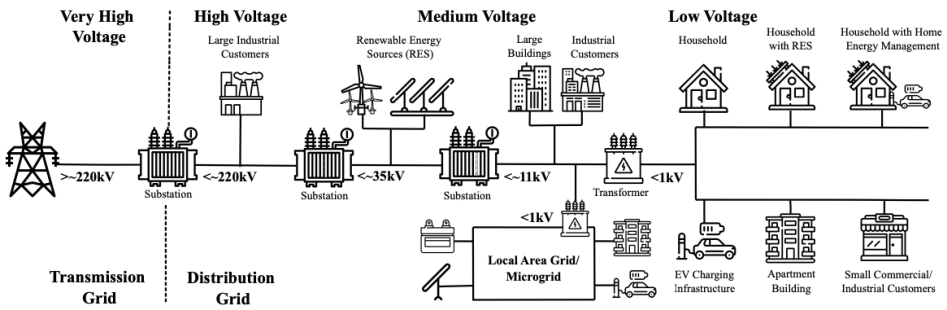


Figure 5.1: Electricity grid [35]

5.2 Electricity markets

Electricity markets have gone rapid changes all over the world during the last three decades. Before this revolution, the power sector was a natural monopoly. In particular, it was characterized by a high vertical integration and by very little market competition. Better technologies in both transmission, generation and distribution are the reasons for such liberal shift. The rationale is simple, a competitive market promotes efficiency and stimulates technical innovation more than a monopoly does.
During the nineties various wholesale electricity markets were established; for example in Chile, Great Britain, the scandinivian block, Australia, New Zealand, the US and Canada. Consequently, new participants, each with a specific task, entered the power market scenario.

- Electricity generators produce electricity to be consumed.
- Electricity transmission owners move high voltage electricity to electric utility companies whilst ensuring safe and reliable supply.
- Electricity grid operators are responsible for scheduling electricity over the transmission network in order to ensure supply demand balance.
- Electric utilities deliver power over local lines to consumers.
- Retail energy suppliers purchase electricity in the wholesale market from electricity generators and resell it to consumers.

Notice, these functions are not mutually exclusive; that is an entity can provide one or more of these services.

5.2.1 The marketplace

We can differential between two kinds of electricity markets: power pools and power exchanges. In power pools, generators bid the prices at which they are willing to produce at different volumes. Then, the market clearing price MCP is determined by intersecting the aggregated supply curve and the estimated demand, left panel figure 5.2. Power pools are created on public initiatives of governments. Conversely, power exchanges are created through private agreements between generators, distributors and traders. This is the model followed by most of european countries. The MCP in power exchanges is determined by the intersection of the aggregate supply curve and the aggregate demand curve, right panel figure 5.2.
It is worth to mention the two biggest power exchanges: Epex and NordPool. Epex operates throughout continental europe. NordPool covers the nordinc and baltic contries.
Moreover, we can differentiate between two popular types of auctions: uniform-price/marginal and pay as bid/discriminatory. Within the uniform price setting, suppliers offering less than the clearing price are paid

that price. Analogously, consumers bidding more than the clearing price pay that price. On the other hand, in pay as bid auctions, suppliers are paid the exact price they bid for.

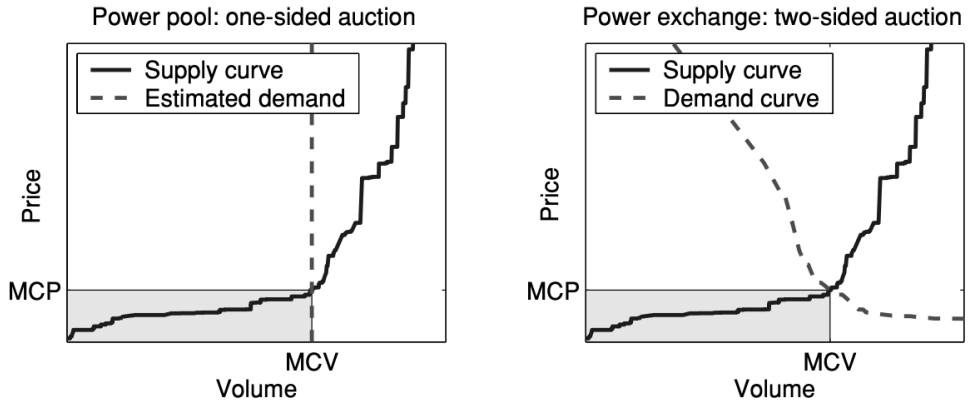


Figure 5.2: Pool market versus power echange [97]

5.2.2 How auctions work

We can drill down on the auction types, the two most popular forms are day ahead and intraday auctions.

5.2.3 Day ahead

The day ahead market is based on a blind auction taking place every day of the year. During these auctions, the prices of all hours for the next day are traded. Market participants can enter their buy sell orders until order book closure, which takes place at 12.00 PM. After that, the market establishes hourly prices by intersecting the demand and supply curves for each hour of the following day.

5.2.4 Intraday

The intraday market consists of continous trading 24/7. A trade is executed as soon as a buy and a sell order mathch. Electricity can be traded an hour, half-hour, quarter or also 5 minutes before delivery. This enables market participants to balance their positions in real time, should they need to do so.

5.2.5 Price peaks

Electricity market are characterized by spikes in their spot prices; when this happens, the system price jumps abruptly and then drops back within a very short period. This spikness follows from the non storability of electricity;

that is, electricity has to be consumed as it is produced. Therefore, extreme load fluctuations combined with generation outages or transmission failures can result in price spikes.

5.2.6 Negative prices

It is not unusual to observe negative prices in electricity markets, even though they are rare. The causes of negative prices are inflexible power generation plants and low demand. With inflexibility, it is mean the fact that power sources cannot be switched off and restarted quickly and efficiently. Thus, producers are faced with the decision of either stopping and restarting their power plant or selling their energy for a negative price; that is they pay consumers for consuming their energy.

5.2.7 Nodal and zonal pricing

Grid configuration and physical limits on electricity lines may lead to congestion. Local marginal price (LMP) and zonal market clearing pricing (ZMCP) are two types of pricing schemes utilized to handle it. The former is adopted in EU contries while the latter is used in the US, figure 5.3 and figure 5.4. With ZMCP, prices may differ between zones but are the same within the same area. On the other hand, LMP is made up by summing the transmission congestion cost, generation marginal cost and the cost of marginal losses at different buses; buses is where a electicity line or several lines are connected.

5.3 Smart grids

A smart meter is a digital device recording energy consumption in real time and communicating this data to the supplier and the consumer. Deployment of smart grids and the digitalisation of the electricity network are intended to increase the energy efficiency of our networks. The reason is that, smart metering and the electricity network digitalisation will enable system operators to better monitor the network, plan their investments and manage infrastructure. All of this motivates for the need of efficient and robust tools to analyze the vast data resulting from the digitalisation of the power sector.

5.4 Renewables

A pressing challenge facing policymakers today is the energy transition towards cleaner energy, see figure 5.5 for an up to date breakdown on energy production by source. Such move is intended to mitigate the effects of climate change and reduce pollution at the same time. This transition passes through renewables integration into the electric grid. Nevertheless,

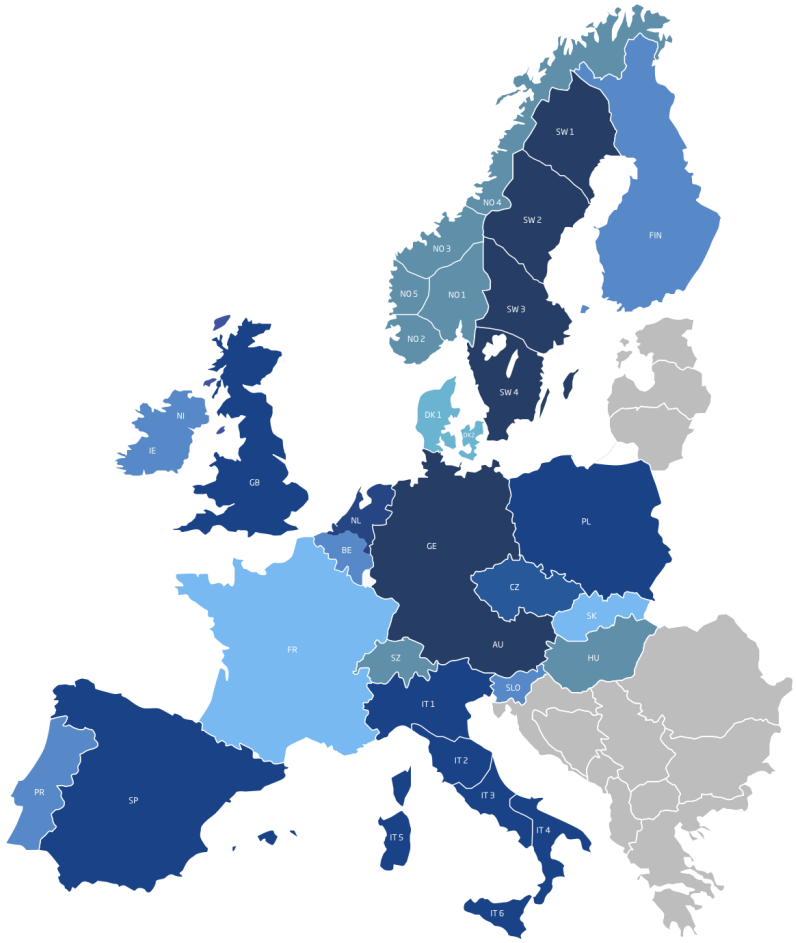


Figure 5.3: Zonal pricing EU [1]

integrating renewables into the existing electric grid involves several technical challenges. First, we need to develop infrastructures and technologies capable of connecting renewable power plants with the existing grid. For example, connecting renewables power plants located in remote and offshore locations to high voltage powerlines is not trivial. Additionally, renewables such as wind and solar are characterized by seasonalities and depend heavily on weather conditions. Therefore, balancing the electricity network and mantaining stability will become harder for grid operators; they will need more flexibility in the grid and develop new approaches in order to achieve their goals.

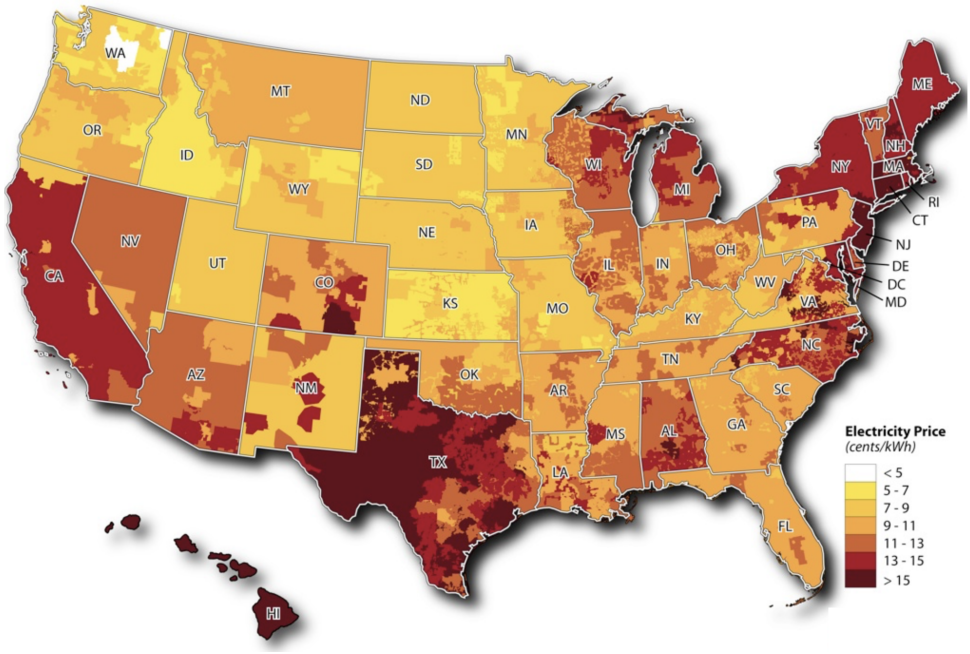


Figure 5.4: Local pricing US [4]

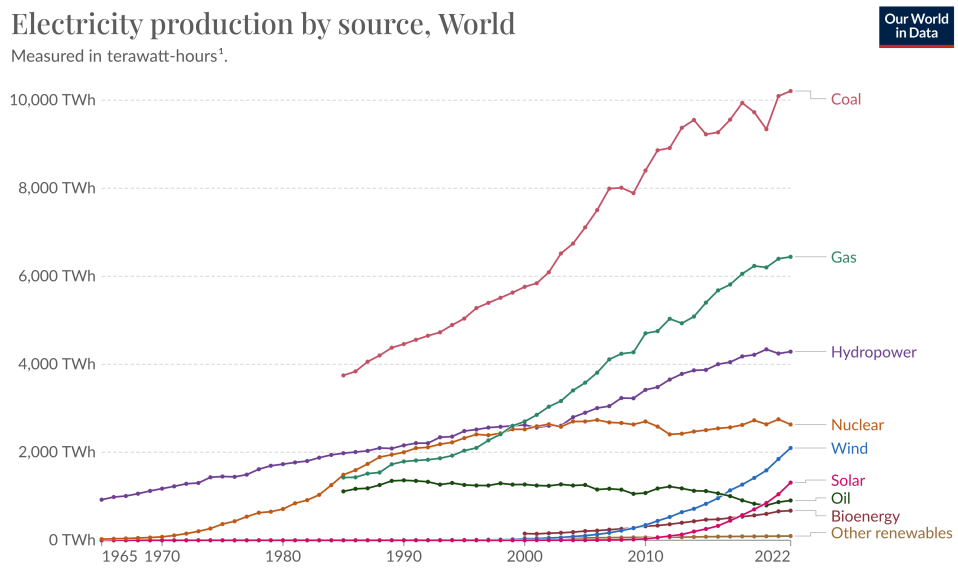


Figure 5.5: Electricity production by source

Chapter 6

Point forecasting

This chapter covers the theory of the most widely used method for point forecasting in the EF literature. Besides discussing the theory underlying such models, this and the following chapter include also a couple of worked examples in order to get acquainted with the practical applications of these models.

6.1 Multiple linear regression

Notwithstanding its simplicity, multiple linear regression models are still popular among the EPF literature. Notice, they are not used per se but usually combined with other more advanced models.

$$L_t = \beta X_t + \varepsilon_t \tag{6.1}$$

6.2 Autoregressive models

Autoregressive models are a standard approach for modelling time series data. Before introducing the popular model in EPF, it is important to remember that autoregressive models assume the time series to be stationary. On the other hand, load and price have been observed to be a non stationary time series. Basically, stationarity means that the distribution of any subsequence of random variables of the stochatic process is invariant to shifts along the time dimension.
Therefore, checking stationarity is a crucial step in applying this class of models. Furthermore, should the time series be non stationary, we have to convert it to a stationary one by a combination of either differencing, detrending and/or log transforming it.

6.2.1 Autoregressive integrated moving average

ARIMA(p,d,q) models are of the form

$$\hat{L}_N = \sum_{i=1}^{p} \varphi_i L_{N-i}^{(d)} + \sum_{j=1}^{q} \theta_j \varepsilon_{N-j} + \varepsilon_N \tag{6.2}$$

where the parameter p is the order of the AR part, d is the degree of integrated differencing and q is the order of the MA model. ε_{N-j} are the observed errors in the past while the differenced term $L_N^{(d)}$ is defined recursively as $L_N^{(d)} = L_N^{(d-1)} - L_{N-1}^{(d-1)}$. ARIMA model are compactly written as

$$\varphi(B)\nabla^d y_t = \theta(B)\varepsilon_t \tag{6.3}$$

where B is the backward shift operator, $B^h y_t = y_{t-h}$ and $\varphi(B)$ is shorthand for $\varphi(B) = 1 - \varphi_1 B - \cdots - \varphi_p B^p$. Similarly, $\theta(B) = 1 + \theta_1 B + \cdots + \theta_q B^q$ and ∇_h is the lag h differencing operator $\nabla_h y_t = (1 - B^h) y_t = y_t - y_{t-h}$
The most popular procedure for estimating this class of models is the Box-Jenkins method [14]. The order p is identified by taking the lags at which the sample partial autocorrelation function (PACF) falls outside its 95% confidence interval. In the same way, the order q is estimated by considering the sample autocorrelation function of the observed time series. Finally, we compare ARIMA models with different p,d,q (in a neighbourhood of the identified orders) and choose the ones minimising either AIC or BIC criteria.

6.2.2 Autoregressive integrated moving average with exogenous variables

ARIMAX model adds explanatory variables to our ARIMA, hence, we have

$$\hat{L}_N = \sum_{k=1}^{h} \mu_i X_{N-k} + \sum_{i=1}^{p} \varphi_i L_{N-i}^{(d)} + \sum_{j=1}^{q} \theta_j \varepsilon_{N-j} + \varepsilon_N \tag{6.4}$$

6.2.3 Seasonal autoregressive integrated moving average and Seasonal autoregressive integrated moving average with exogenous variables

SARIMA and SARIMAX incorporate seasonalities into our time series models. The notation for these models is ARIMA$(p,d,q)(P,D,Q)_m$. The second term $(P,D,Q)_m$ represents the autoregressive structure of the seasonal patter, where m indicates the number of seasonalities of our time series. For instance, in modelling series of hourly data with daily periodicity, we would set m=24. Writing SARIMA in compact notation we have

$$\varphi(B)\Phi(B^s)\nabla^d \nabla_m^D y_t = \theta(B)\Theta(B^s)\varepsilon_t \tag{6.5}$$

6.3 Generalized additive model

Generalized additive models (GAMs) take the form

$$g[\mathbb{E}(Y_i)] = \alpha + f_1(X_1) + \cdots + f_p(X_p) \tag{6.6}$$

f_i are smooth functions which are fitted using cubic smoothing splines. g is called link function and characterizes the specific GAM model [40]. Approximations for f_i are obtained through an iterative procedure, the backfitting algorithm.

Particularly successful in time series forecasting is the Prophet model [88], developed by Meta. This additive model is capable of handling non linear trends, holiday effects and yearly weekly and daily seasonality. Thus, the reason we choose it as a valid benchmark to compare against.

6.4 K-nearest neighbours regression

K-nearest neighbours regression forecasts by averaging the most similar k instances in the training set [63]. K is the algorithm hyperparameter, it stands for the number of neighbours; a small k leads to overfitting while a big k leads to underfitting. Since it is a metric based algorithm, it is important to normalize data in order to give equal weights to features with different scales, see A.4.1. This algorithm is made up of two decisions: first, the choice of the metric and second, the method for combining targets.

6.5 Support vector regression

Developed at the AT&T Bell Laboratories by Vapnik et al. [22] [93], support vector machines SVMs are one of the most popular techniques within the field of statistical learning.
The goal of support vector regression is finding a function $f(x)$ with at most ε deviation from the actual observed data y_i for every i and as flat as possible. Put differently, we would like a model to keep error less than the ε threshold, In standard SVR we have

$$f(x) = \langle w, x \rangle + b \text{ with } w \in \mathcal{X}, b \in \mathbb{R} \tag{6.7}$$

Where $\mathcal{X}$ denotes the space of the input patters x_i. We can translate the flatness requirement into minimising the squared norm of w. Doing so, we can formulate our problem as a convex optimisation problem.

$$\begin{aligned} \min \quad & \frac{1}{2}\|w\|^2 \\ s.t. \quad & y_i - \langle w, x_i \rangle - b \leq \varepsilon \\ & \langle w, x_i \rangle + b - y_i \leq \varepsilon \end{aligned} \tag{6.8}$$

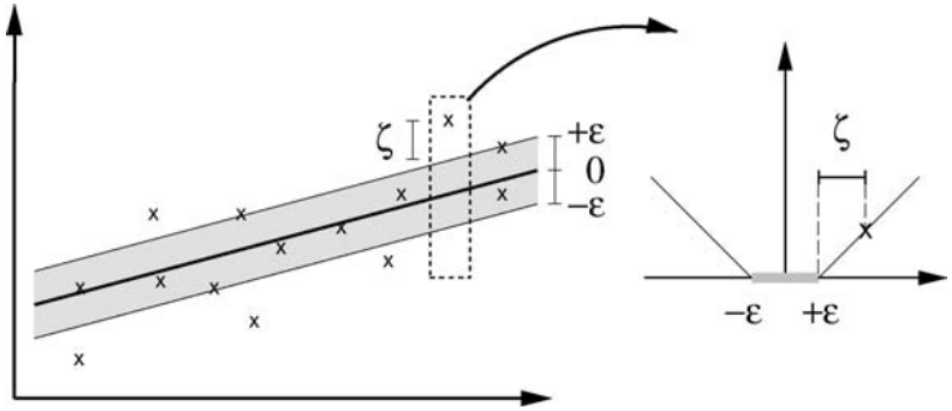


Figure 6.1: Support vector regression,[83]

Next, we can introduce the slack variables ξ and ξ^* and obtain the following equivalent formulation

$$\begin{aligned} \min \quad & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \\ s.t. \quad & y_i - \langle w, x_i\rangle - b \leq \varepsilon + \xi \\ & \langle w, x_i\rangle + b - y_i \leq \varepsilon + \xi^* \\ & \xi_i \geq 0 \\ & \xi_i^* \geq 0 \end{aligned} \tag{6.9}$$

The C constant trades off between ε deviation tolerance and flatness of the function f. A bigger C gives more importance to error minimization while as C gets smaller, flattness gains importance.
We seek to minimise the epsilon insensitive loss function, defined as follows

$$\|\xi\|_\varepsilon := \begin{cases} 0 & \text{if } \|\xi\| \leq \varepsilon \\ \|\xi\| - \varepsilon & \text{if } \|\xi\| > \varepsilon \end{cases} \tag{6.10}$$

The model is depicted in figure 6.1. The grey band is called the epsilon insensitive tube, only the points outside it are accounted by the loss function. Smola et al. [84] point out that considering the dual formulation makes our optimisation problem easier to solve. Doing so we have

$$\begin{aligned} \max_{\alpha_i, \alpha_i^*} \quad & -\frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\langle x_i, x_j\rangle + \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)(y_i - \varepsilon) \\ s.t. \quad & \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0 \\ & \alpha_i, \alpha_i^* \in [0, C] \end{aligned} \tag{6.11}$$

Rearranging the gradient of the lagrangian with respect to w, we obtain the so called support vector expansion.

$$w = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) x_i \tag{6.12}$$

This means that w can be completely described as a linear combination of the training features x_i. Notice, the complexity of the function representation is independent of the feature space dimensionaly, but depends only on the number of support vectors; that is those point i for which $(\alpha_i - \alpha_i^*) \neq 0$ Moreover, we do not needd to compute w explicitly in order to evaluate $f(x)$. Furthermore, equation 6.12 implies

$$f(x) = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) \langle x_i, x \rangle + b \tag{6.13}$$

Employing the Karush-Kuhn-Tucker conditions, b can be retrieved easily. Particulalry, the condition that the product between the constraints and the dual variable has to vanish. These implies that the lagrange multipliers α_i, α_i^* may be nonzero only for the samples inside the epsilon insensitive tube. Consequently for any of these data points, the equality $b = y_i - \langle w, x_i - \varepsilon \rangle$ holds.
To get an idea of how SVR works, see figure 6.2 for how a support vector regression handles a sinusoidal function with noise.

6.6 Artificial neural networks

Artificial neural networks (ANN) have been successfully applied in the context of electricity markets forecasting. The basic building block of this class of methods is the perceptron [77].

$$y = \psi(wx + b) \tag{6.14}$$

where ψ is an activation function and w, b are the parameters of the neuron; such parameters are typically optimized through gradient descent.

6.6.1 Multilayer perceptrons

In order to cover a richer space of models we can stack neurons, doing so we obtain the so called multi layer perceptrons (MLPs). Hyperparameters of these models are the number of hidden layers, the number of neurons in each of those layers and the kind of activation function. Notice, depending on the numbers of layers, MLPs are sometimes referred to as deep neural networks.

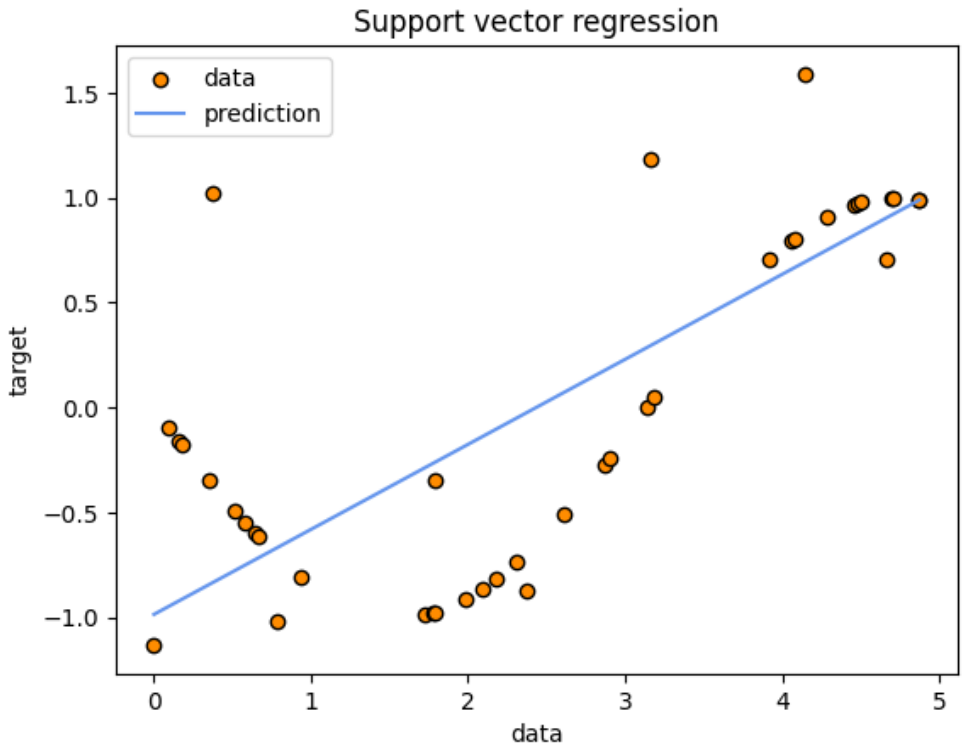


Figure 6.2: support vector regression

6.6.2 Long short term memory

LSTM extends RNN by introducing a cell state in its layers. In this way, the current state of a cell depndes on the current value X_t, on the previous cell activation and on the previous cell state C_{t-1}, see figure 6.3 for a visual representation. The cell is responsible for deciding whether to store/forget

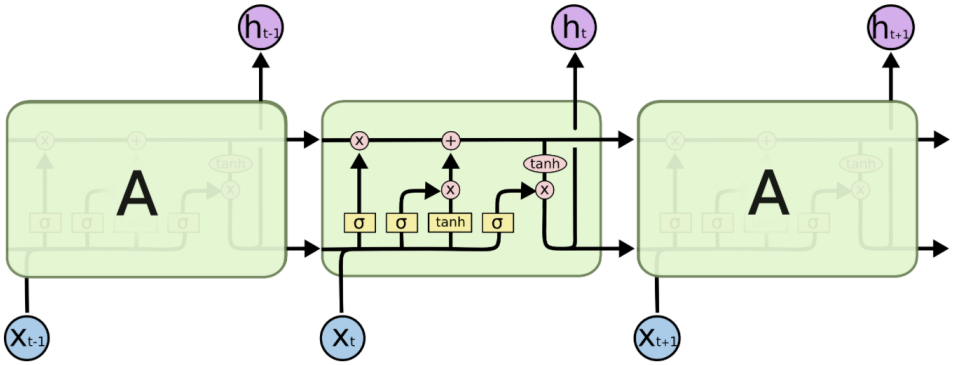


Figure 6.3: Long short term memory cell

long and short term information. To do so, each cell is made of a forget gate, an input gate and an output gate. The forget gate takes in h_{t-1} and x_t and outputs a real valued vector with elements in the domain $[0, 1]$, this corresponds to the ratio of information retention for each element of the cell state C_{t-1}.
Its equation is given by

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{6.15}$$

The input gate is made up of two parts. First, a sigmoid layer decides which values of the cell state will be updated. Second, a tahn layer creates a vector of candidates $\tilde{C}_t$ to add to the current state.

$$\begin{aligned} i_t &= \sigma(W_i[h_{t-1}, x_t] + b_i) \\ \tilde{C}_t &= tahn(W_c[h_{t-1}, x_t] + b_C) \end{aligned} \tag{6.16}$$

The cell state is then updated by

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{6.17}$$

Finally, we output the current cell state and the tanh activated cell state filtered by the sigmoid layer on the current input and the previous hidden state.

$$\begin{aligned} o_t &= \sigma(W_o[h_{t-1}, x_t] + b_o) \\ h_t &= o_t \odot tahn(C_t) \end{aligned} \tag{6.18}$$

6.7 Kernel methods

6.7.1 Kernel ridge regression

In the setting of kernel ridge regression, we aim at estimating f such that $y = f(x) + \varepsilon$. Given a RKHS $\mathcal{H}$, we can estimate$\hat{f}$ by solving the optmisation problem

$$\hat{f} = \arg\min_{f \in \mathcal{H}} \frac{1}{2} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \frac{\lambda}{2} \|f\|_{\mathcal{H}}^2 \tag{6.19}$$

Before proceeding, we need to introduce the representer theorem.

Theorem 6.1 *Let $k : \mathcal{X} \times \mathcal{X} \mapsto R$ on a non empty set $\mathcal{X}$ with corresponding RKHS $\mathcal{H}$ and consider the regularized risk functional $Risk[f] : f \mapsto E\left((x_1, y_1, f(x_1)), \ldots, (x_n, y_n, f(x_n))\right) + g\left(\|f\|\right)$. Then any minimizer of the empirical risk $Risk[f]$ can be represented as $f^*(\cdot) = \sum_{i=1}^{n} \alpha_i k(\cdot, x_i)$*

Proof By orthogonal projection, any $f \in \mathcal{H}$ can be decomposed as the sum of two functions, with the first lying in the span$\{\varphi(x_1), \ldots, \varphi(x_n)\}$ and the

second staying in the orthogonal complement of the former. Letting ν denote the function from the orthogonal complement, we have

$$f = \sum_{i=1}^{n} \alpha_i \varphi(x_i) + \nu \tag{6.20}$$

Applying f to any point, by the reproducing property we have

$$\begin{aligned} f(x_j) =& \langle \varphi(x_j), \sum_{i=1}^{n} \alpha_i \varphi(x_i) + \nu \rangle \\ =& \sum_{i=1}^{n} \alpha_i \langle \varphi(x_j), \varphi(x_i) \rangle \end{aligned} \tag{6.21}$$

This means that setting $\nu = 0$ does not affect the evaluation of f. Next consider the regularization term; notice, the functional g has to be strictly monotonic in order to penalize more complex functions f.

$$\begin{aligned} g(\|f\|) =& g\left(\left\| \sum_{i=1}^{n} \alpha_i \varphi(x_i) + v \right\|_{\mathcal{H}} \right) \\ =& g\left(\sqrt{\left\| \sum_{i=1}^{n} \alpha_i \varphi(x_i) + v \right\|_{\mathcal{H}}^2} \right) \\ =& g\left(\sqrt{\left\| \sum_{i=1}^{n} \alpha_i \varphi(x_i) \right\|_{\mathcal{H}}^2 + \|v\|_{\mathcal{H}}^2 + 2\langle \sum_{i=1}^{n} \alpha_i \varphi(x_i), \nu \rangle_{\mathcal{H}}} \right) \\ =& g\left(\sqrt{\left\| \sum_{i=1}^{n} \alpha_i \varphi(x_i) \right\|_{\mathcal{H}}^2 + \|v\|_{\mathcal{H}}^2} \right) \\ \geq& g\left(\sqrt{\left\| \sum_{i=1}^{n} \alpha_i \varphi(x_i) \right\|_{\mathcal{H}}^2} \right) \end{aligned} \tag{6.22}$$

Hence, we can conclude that setting $\nu = 0$ decreases the second term while the first term is not affected. Thus it follows that any minimizer of the risk functional 6.19 must be of the following form

$$f^*(\cdot) = \sum_{i=1}^{n} \alpha_i \varphi(x_i) = \sum_{i=1}^{n} \alpha_i k(\cdot, x_i) \tag{6.23}$$

□

Employing the representer theorem, we can rewrite 6.19 in matrix notation as

$$\begin{aligned}
\hat{\alpha} =& \arg\min_{\alpha\in\mathbb{R}^n} \frac{1}{2}\|y - K\alpha\|_2^2 + \frac{\lambda}{2}\left\|\sum_{i=1}^{n}\alpha_i k(\cdot, x_i)\right\|_{\mathcal{H}}^2 \\
\hat{\alpha} =& \arg\min_{\alpha\in\mathbb{R}^n} \frac{1}{2}\|y - K\alpha\|_2^2 + \frac{\lambda}{2}\langle\sum_{i=1}^{n}\alpha_i k(\cdot, x_i), \sum_{j=1}^{n}\alpha_j k(\cdot, x_j)\rangle_{\mathcal{H}}^2 \\
\hat{\alpha} =& \arg\min_{\alpha\in\mathbb{R}^n} \frac{1}{2}\|y - K\alpha\|_2^2 + \frac{\lambda}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j\langle k(\cdot, x_i), k(\cdot, x_j)\rangle_{\mathcal{H}}^2 \\
\hat{\alpha} =& \arg\min_{\alpha\in\mathbb{R}^n} \frac{1}{2}\|y - K\alpha\|_2^2 + \frac{\lambda}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j K_{ij} \\
\hat{\alpha} =& \arg\min_{\alpha\in\mathbb{R}^n} \frac{1}{2}\|y - K\alpha\|_2^2 + \frac{\lambda}{2}\alpha^\intercal K\alpha \\
\hat{\alpha} =& \arg\min_{\alpha\in\mathbb{R}^n} \frac{1}{2}(y - K\alpha)^\intercal(y - K\alpha) + \frac{\lambda}{2}\alpha^\intercal K\alpha
\end{aligned} \tag{6.24}$$

Setting the gradient to zero

$$-Ky + K^2\alpha + \lambda K\alpha \stackrel{!}{=} 0 \tag{6.25}$$

We have then, that $\hat{\alpha} = (K + \lambda I)^{-1}y$. It follows the solution takes the form

$$\hat{f}(\cdot) = \hat{\alpha}K(\cdot, :) \tag{6.26}$$

6.7.2 Kernel support vector regression

Support vector regression can be kernelized by swapping the $\mathbb{R}^2$ euclidean dot product of data x with the dot product in the higher feature space F. Doing so, the optimisation problem can be restated as

$$\begin{aligned}
\max_{\alpha_i,\alpha_i^*} \quad & -\frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i - \alpha_i^*)(\alpha_i - \alpha_i^*)k(x_i, x_j) + \sum_{i=1}^{n}(y_i - \varepsilon)(\alpha_i - \alpha_i^*) \\
s.t. \quad & \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0 \\
& \alpha_i, \alpha_i^* \in [0, C]
\end{aligned} \tag{6.27}$$

Our regressor f is then given by

$$f(x) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)k(x_i, x) + b \tag{6.28}$$

Notice, in this setting w is no longer given explicitly. Additionally, with kernel SVR we seek for the flattest function in the feature space not the input

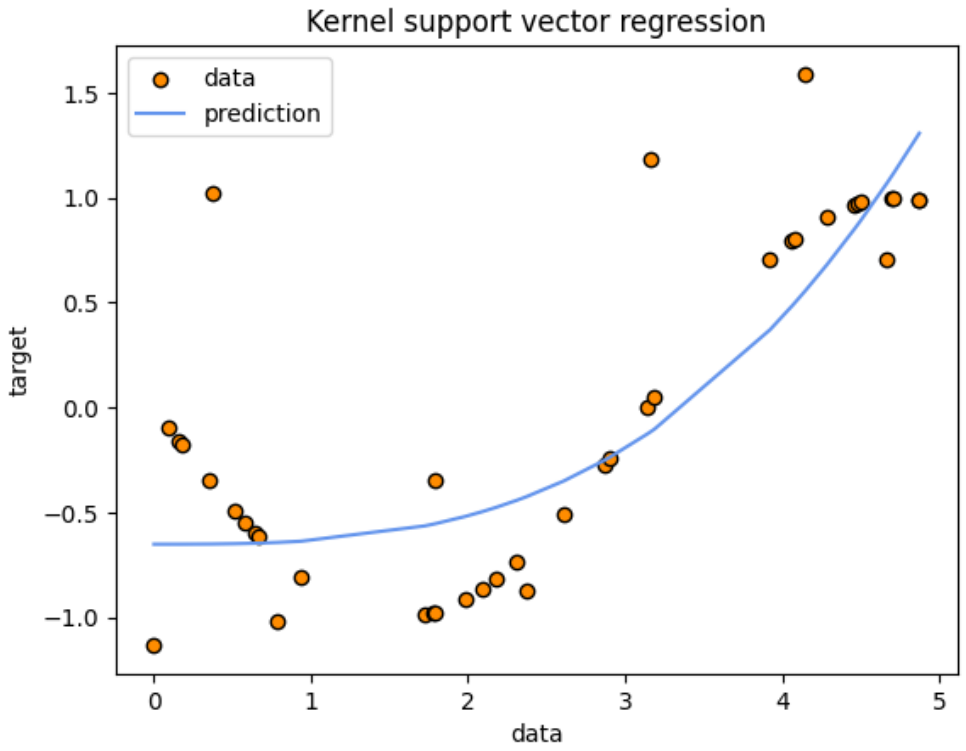


Figure 6.4: polynomial support vector regression

space. See figure 6.4 and figure 6.5 for two examples. In the former, we have used a polynomial kernel while in the latter the standard radial basis function kernel.

Comparing theese pictures with figure 6.2, it can be concluded that introducing kernels allows support vector regression to handle the non linearities in the data.

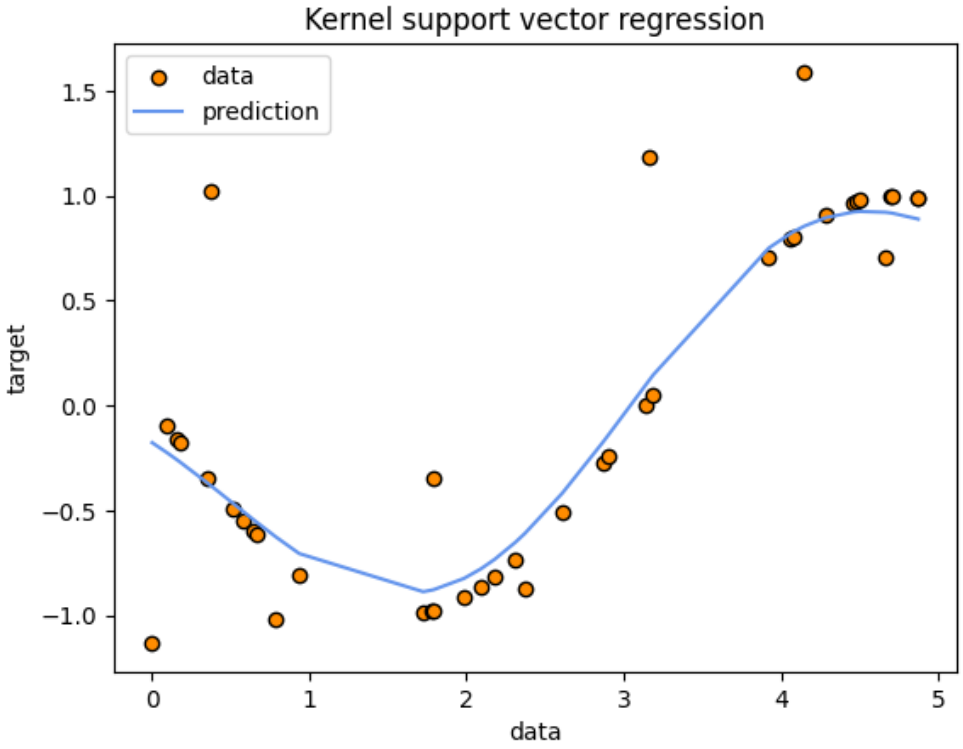


Figure 6.5: rbf support vector regression

Chapter 7

Probabilistic forecasting

When it comes to probabilistic forecasting, there are a couple of standard practical approaches; conceptually they can be grouped into two main categories. The former class of approaches tries modelling the distribution of the observed data directly. The latter family of approaches constructs first a point forecast and then learns the distribution of the model's errors.

7.1 Quantile regression

Quantile regression can be interpreted as an extension of standard regression. In this setting, you basically slice the dependent variables into quantiles and then fit a regression for each quantile. With standard regression, we build a model for the conditional mean, conversely, with quantile regression we model the conditional quantile function for any desired quantile. Therefore, with quantile regression we are able to study the impact of covariates on quantiles directly.

Definition 7.1 *For any real valued random variable Y, we define its associated quantile function.*

$$Q(q) = \inf\{y : F(y) \geq q\} \tag{7.1}$$

Alternatively, in order to ease the posing of the quantile regression problem, we can formulate quantiles as the solutions to simple optimization problems. For any $0 < q < 1$ consider the pinball loss function from section 4.7, $\rho_q(u) = u(q - \mathbb{1}_{\{u<0\}})$. Such loss is minimized by the quantile $Q(q)$. Thus,we can estimate quantiles by minimizing the expectation of $\rho_q(Y - g(x, \beta))$ with respect to the parameter β.
Note, that in the special case $q = \frac{1}{2}$, quantile regression corresponds exactly to standard regression with an absolute value loss function.
It follows that the conditional linear quantile function $Q_Y(q|X = x) = x_i\beta(q)$,

can be estimated by solving

$$\hat{\beta}(q) = \arg\min_{\beta} \sum \rho_q(y_i - x\beta) \tag{7.2}$$

Notice that, this cost function is not differentiable, therefore there is no analytical solution to the quantile regression problem. Nevertheless, we can easily solve it by employing linear programming and convex optimisation [15].
Furthermore, we can extend this framework to non linear quantile regression by choosing a non linear model in place of $x\beta$ in the above equation 7.2.

7.2 Quantile forest

Meinshausen [66] extends the idea of random forest [16] generalising it, the result is the quantile forest algorithm. Quantile forest allows us to estimate conditional quantiles in a non parametric fashion.
In order to understand this algorithm, it is first necessary to first cover the theory of decision trees and random forests.

7.2.1 Decision trees

Decision trees methods partition recursively the feature space in a set of binary rectangles and then fit a simple model in each of those partitions (the most straightforward is fitting just a constant). To get started, we first split the space into two disjoint regions, then we model the response variable by the mean of the observed predicted variables with associated features falling in that specific region. Our goal is selecting the best features and best split point such that to achieve the best generalizing fit. For an example/visualisation consider the pictures 7.1 and 7.2 taken from [39], where the decision tree algorithm is visualised for a regression problem with two independent variables X_2 and X_1.

Suppose to partition the feature space into M regions $R_1, \ldots, R_M$, then the model reads as follows.

$$f(x) = \sum_{m=1}^{M} c_m \mathbb{I}_{\{x \in R_m\}} \tag{7.3}$$

It follows that the function minimising the sum of squares is the one with $\hat{c}_m = mean(y_i | x_i \in R_m)$. Finding the best split in terms of minimum sum of squares is computationally infeasible in practice. Therefore we approximate a solution by approaching the problem in a greedy fashion. Let j denote the splitting variable and s be the the split point we define the two half planes

$$R_1(j,s) = \{X | X_j \leq s\} \quad R_2(j,s) = \{X | X_j \, s\} \tag{7.4}$$

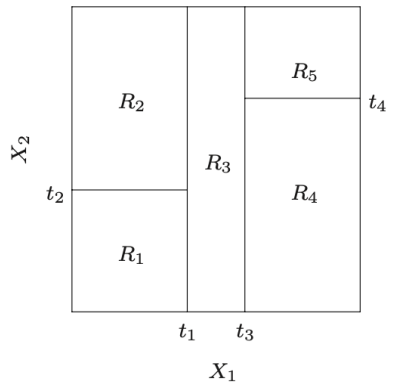


Figure 7.1: two-dimensional feature space partitioned by recursive binary splitting

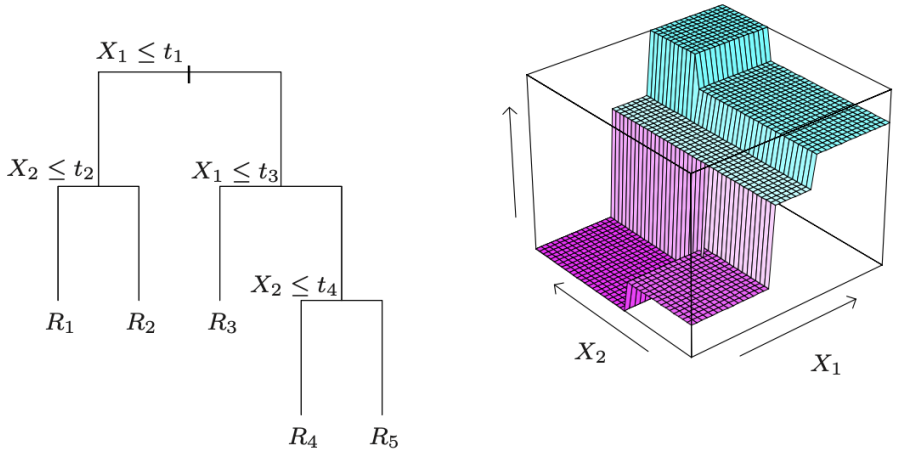


Figure 7.2: partition tree and regression model

Then we search for the s and j that solve

$$\min_{j,s} \left[\min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \right] \tag{7.5}$$

The inner problem is easy, as already pointed out we will have

$$\hat{c}_1 = mean(y_i | x_i \in R_1) \quad \hat{c}_2 = mean(y_i | x_i \in R_2) \tag{7.6}$$

For the outer problem, we scan through all the (j,s) tuples and pick the best pair. Next, one or both of these regions from the previous step are split into two more regions; we recurse this process until some stopping condition is triggered (max number of branches, max deth of tree, threshold on the MSE,

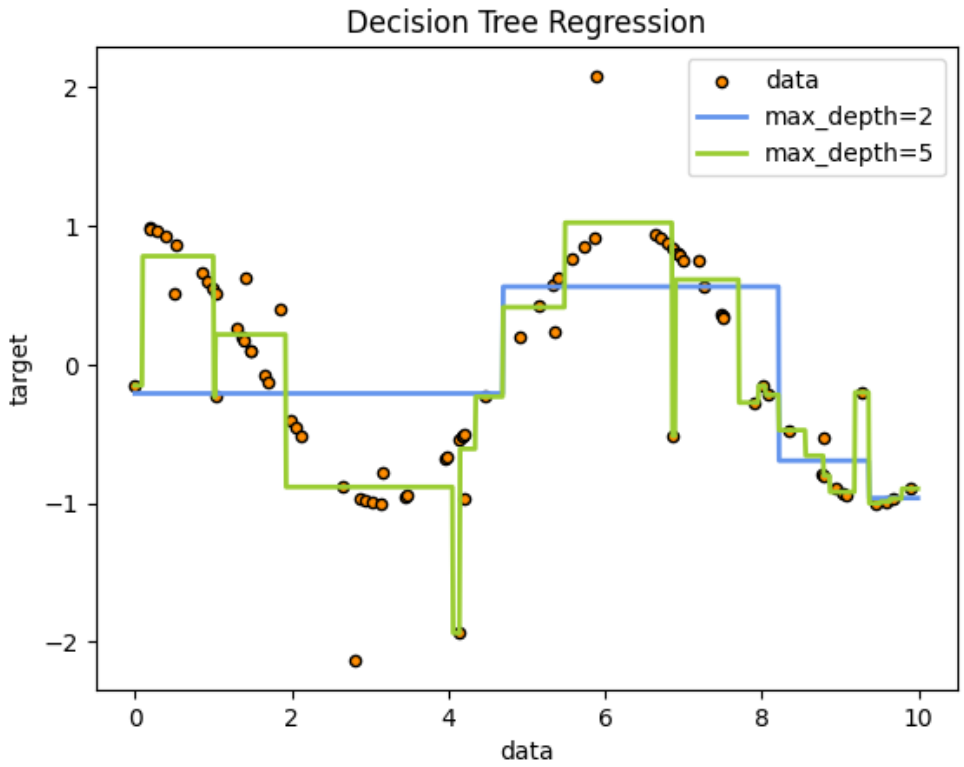


Figure 7.3: decision tree regression

minimum number of observation in each leaf node). Notice, this being a greedy algorithm implies that our final solution is guaranteed to be just a local optimum not a global one. Even though their simplicity, these models have proved themselves to be really powerful. See figure 7.3 for a example where decision tree with different hyperparameters are fitted to a sine wave plus noise. The most popular among these model is the CART [17] tree, its name comes from the fact that it can handle both classification and regression problems.

7.2.2 Bootstrap

The idea behind boostrapping is to randomly sample from the training set with replacement B times and then fitting B models to each of the "artificial" datasets. Boostrapping can serve different tasks; we can use it to assess the accuracy of a parameter estimate or of a prediction but also to improve their estimates.

7.2.3 Bagging

Bagging stands for boostrap aggregation. The bagging estimate is dafined by $\mathbb{E}_P[\hat{f}^*]$ where P is the empirical distribution putting equal probability on each data point of the training set. Basically, for each bootstrap fitted model $\hat{f}^{*b}(x)$, we compute the bagging estimate by

$$\hat{f}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{*b}(x) \tag{7.7}$$

Bagging is particularly useful in reducing the variance of decision trees, resulting in an improved prediction(tradeoff bias variance). Note, this improvement comes from the fact that averaging reduces variance and leaves biases unchanged.

7.2.4 Random forest

Decision trees are characterised by high variance and low bias, thus, they can benefit extremely from bagging. Furthermore, every decision tree generated through bagging will be identically distributed (i.d.), thus the expectation of an average of B trees is probabilistically equivalent to the expectation of any such trees. As a consequence, the bias will stay fixed since the bias of the bagged estimator is the same as that of each individual tree.
Consider positively correlated i.d. random variables, then the variance of their average is

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2 \tag{7.8}$$

The second term disappears as B increases, while the first term depends heavily on the correlation between bagged trees. Random forest consists in is reducing the correlation between the trees by randomly selecting m of the p features as candidates for splitting; m tipically takes a value in the order of $\sqrt{p}$ or even 1 with default value $m = \lfloor \frac{p}{3} \rfloor$, while a good minimum node size is around five.
Letting $T(x;\Theta_b)$ be the bth bagged tree where Θ_b denotes the randomness characterizing its splits, cutpoints and terminal node values, we have that the random forest regressor is given by

$$\hat{f}_{rf}^{B} = \frac{1}{B}\sum_{b=1}^{B}T(x;\Theta_b) \tag{7.9}$$

For a simple visualisation compare figure 7.4 with 7.3

7.2.5 Quantile forest

The key observation here is noting that random forests approximates the conditional mean $\mathbb{E}(Y|X=x)$ by a weighted average over the observed Y.

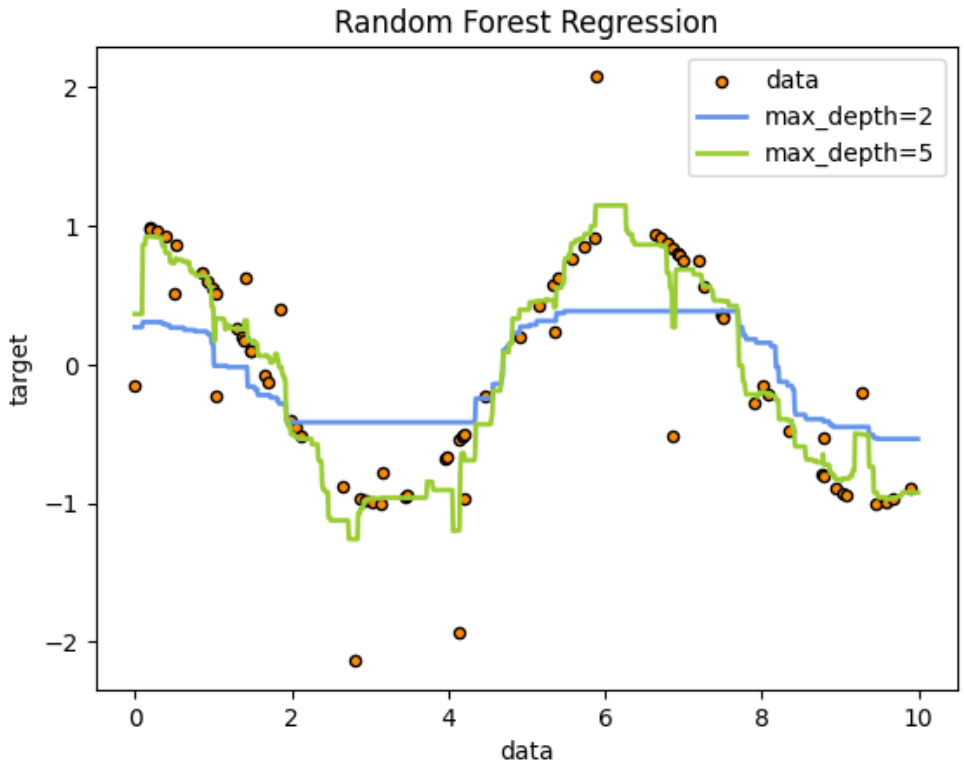


Figure 7.4: random forest regression

Hence, we can extend this idea to the full conditional distribution by

$$F(y|X=x) = P(Y \leq y|X=x) = \mathbb{E}(\mathbb{I}_{\{Y\leq y\}}|X=x) \tag{7.10}$$

All we have to do is approximating $\mathbb{E}(\mathbb{I}_{\{Y\leq y\}}|X=x)$ by a weighted mean over the random variable $\mathbb{I}_{\{Y\leq y\}}$

$$\hat{F}(y|X=x) = \sum_{i=1}^{n} \omega_i(x)\mathbb{I}_{\{Y_i\leq y\}} \tag{7.11}$$

By swapping $F(y|X=x)$ with $\hat{F}(y|X=x)$ in the defintion of conditional quantiles we obtain their respective random forest estimator

$$\hat{Q}_q = \inf\{y : \hat{F}(y|X=x) \geq q\} \tag{7.12}$$

7.3 Quantile gradient boosting machine

7.3.1 Boosting

With boosting we fit an additive expansion of elementary basis functions; with M basis functions we have

$$f(x) = \sum_{m=1}^{M} \beta_m b(x; \gamma_m) \tag{7.13}$$

where $b(x; \gamma)$ are the basis functions while β_m are the coefficients of the expansion. Boosted models are fitted by minimizing a loss function over the training data

$$\min_{\beta_m, \gamma_m} \sum_{i=1}^{N} L\left(y_i, \sum_{m=1}^{M} \beta_m b(x_i; \gamma_m)\right) \tag{7.14}$$

However, such problem is highly intensive in terms of computation. Therefore, what is done in the literature is approximating its solution by iteratively adding new basis function to the current expansion. That is, we construct f_m by solving for the optimal basis function and coefficients to add to f_{m-1}. Considering the square loss, we would have

$$L(y_i, f_{m-1}(x_i) + \beta_m b(x_i; \gamma)) = (e_{im} - \beta_m b(x_i; \gamma))^2 \tag{7.15}$$

7.3.2 Boosted trees

We combine several trees obtaining the boosted tree model

$$f_M(x) = \sum_{m=1}^{M} T(x; \Theta_m) \tag{7.16}$$

Thus, at each step of the iterative optimisation procedure, we have to solve

$$\hat{\Theta}_m = \arg\min_{\Theta_m} \sum_{i=1}^{N} L(y_i, f_{m-1}(x_i) + T(x_i; \Theta_m)) \tag{7.17}$$

Remember, Θ_m referes to the parameters of the mth tree, $\Theta_m = \{R_{jm}, \gamma_{jm}\}_1^{J_m}$

7.3.3 Gradient boosting

In order to robustly solve 7.17, gradient boosting considers the following problem

$$\tilde{\Theta}_m = \arg\min_{\Theta} \sum_{i=1}^{N} (-g_{im} - T(x_i; \Theta))^2 \tag{7.18}$$

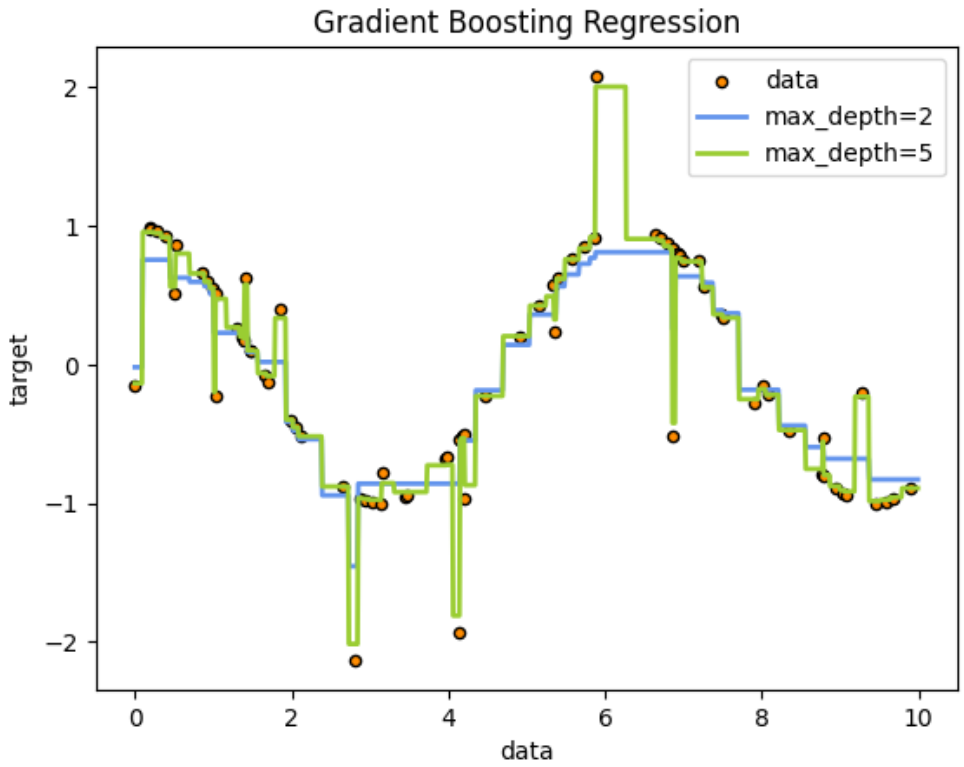


Figure 7.5: gradient boosting regression

where g_{im} is the gradient of $L(f) = \sum_{i=1}^{N} L(y_i, f(x_i))$ evaluated at f_{m-1}. Put simply, we are fitting the m-th tree to the negative of the gradient values of f through least squares. In order to solve our quantile regression tasks through gradient boosting, all we need to do is specifying the pinball loss as our criterion to guide the minimisation algorithm. See figure 7.5 for a visualisation

7.4 Kernel methods

7.4.1 Kernel quantile regression

The idea of quantile regression has been extended to kernel methods by Takeuchi et al. [86]. There, they minimize a risk plus regularizer defined as follows.

$$R[f] := \frac{1}{m} \sum_{i=1}^{m} \rho_q(y_i - f(x_i)) + \frac{\lambda}{2} \|g\|_{\mathcal{H}}^2 \tag{7.19}$$

where $f = g + b$, $g \in \mathcal{H}$ and $b \in R$. Using the link between RKHS and feature spaces, we can rewrite $f(x) = \langle w, \varphi(x) \rangle + b$. Moreover, note that

the RKHS norm is defined as follows $\|f\|_{\mathcal{H}} = \inf\{\|w\|_F : w \in F, f(x) = \langle w, \varphi(x)\rangle_F, \forall x \in X\}$, where F is the feature space. Doing so we obtain a minimization problem equivalent to minimizing equation 7.19.

$$\min_{w,b} \quad C\sum_{i=1}^{m} q(y_i - \langle w, \varphi(x)\rangle - b) + (1-q)(-y_i + \langle w, \varphi(x)\rangle + b) + \frac{1}{2}\|w\|^2 \tag{7.20}$$

Note the division by λ so that $C = \frac{1}{\lambda m}$.
We can next rephrase the optimisation in 7.20 by introducing the slack variables ξ_i and ξ_i^*.

$$\begin{aligned} \min_{w,b,\xi_i,\xi_i^*} \quad & C\sum_{i=1}^{m} q\xi_i + (1-q)\xi_i^* + \frac{1}{2}\|w\|^2 \\ \text{s.t.} \quad & y_i - \langle w, \varphi(x)\rangle - b \leq \xi_i \\ & -y_i + \langle w, \varphi(x)\rangle + b \leq \xi_i^* \\ & \xi_i \geq 0 \\ & \xi_i^* \geq 0 \end{aligned} \tag{7.21}$$

In order to make it more compact, we rewrite equation 7.21 in matrix notation.

$$\begin{aligned} \min_{w,b\xi,\xi^*} \quad & Cq\xi^\intercal \mathbb{1} + Cq(\xi^*)^\intercal \mathbb{1} + \frac{1}{2}w^\intercal w \\ \text{s.t.} \quad & y - \Phi^\intercal w - b \preceq \xi \\ & -y + \Phi^\intercal w + b \preceq \xi^* \\ & \xi \succeq 0 \\ & \xi^* \succeq 0 \end{aligned} \tag{7.22}$$

Consider now, the lagrangian L associated to 7.22

$$\begin{aligned} L(w,b,\xi,\xi^*) =& Cq\xi^\intercal \mathbb{1} + Cq(\xi^*)^\intercal \mathbb{1} + \frac{1}{2}w^\intercal w - \alpha^\intercal(\xi - y + \Phi^\intercal w + b) - (\alpha^*)^\intercal(\xi^* + y - \Phi^\intercal w - b) \\ & - \nu^\intercal \xi - (\nu^*)^\intercal \xi^* \end{aligned} \tag{7.23}$$

The next step is deriving its dual formulation, since it is easier and more efficient to solve. This because the dual problem has the useful property of being always convex.

Definition 7.2 *The dual function associated to the lagrangian $L(x,\lambda)$ is given by $g(v) = \inf_x L(x,\lambda)$*

where λ is called the lagrange multiplier of the optimization problem. Such dual formulation has an useful property, that is

$$g(\lambda) \leq p^* \tag{7.24}$$

where p^* is the optimal value of your optimization problem. Consider a simple lagrangian $L(x, \lambda) = f(x) + \sum \lambda_i r_i(x) + \sum v_i h_i(x)$, where $r_i(x)$ are inequality constraints while $h_i(x)$ are equality constraints of the problem. Then it can be noted that, the lower bound on p^* is non trivial only when the lagrange multiplier $\lambda \succeq 0$. Therefore, the idea is that by maximizing the dual function subject to the contraint $\lambda \succeq 0$, we can obtain an approximate or perfect solution to the primal problem.
To explain why we may or not be able to attain the best solution to the primal problem by maximizing its dual, we have to introduce the concept of duality. We use d^* to denote the optimal value of the lagrange dual problem; we can think of it as the best lower bound on p^*.
The inequality 7.24 is called weak duality. The difference $p^* - d^*$ is said the optimal duality gap; it is the gap between the optimal value of the primal problem and the best lower bound on it that can be obtained from the Lagrange dual function. Moreover, note that the optimal duality gap is always nonnegative.
We say that strong duality holds, when the optimal duality gap is zero; in other words, the lagrange dual bound is tight.
Constraint qualifications are condtions under which strong duality holds; one of the most popular is Slater's condition.

$$\begin{aligned} \exists x \in \text{relint } D \text{ s.t. } r_i(x) < 0, \quad i = 1, \dots, m \\ h_i(x) = 0 \end{aligned} \tag{7.25}$$

Where relint D is the relative interior of $D := \cap_{i=0}^{m} \text{dom}(r_i)$
Slater's theorem naturally follows.

Theorem 7.3 *If Slater's condtion holds and the problem is convex then strong duality holds.*

We can now check that our optimization problem posseses strong duality by checking Slater's condition.
In our case we don't have any equality constraint, so we do not have to worry about the $h_i(x) = 0$ term in 7.25. All we have to check is the convexity of our problem and that there exist a x such that $r_i(x) < 0$. For convexity, a sufficient condition is the positive definiteness of Q in the quadratic programming problem

$$\begin{aligned} \min \quad & x^\intercal Q x + c^\intercal x \\ s.t \quad & Ax \preceq b \end{aligned} \tag{7.26}$$

In our case, equation 7.23, we have $w^\intercal w$, thus Q is just the identity matrix which satisfies the positive definiteness requirement. Therefore, our problem is convex. Next we check that Slater's condition holds. Considering first the two non negative constraints on ξ and ξ^*, we conclude that ξ and ξ^* have to be greater or equal to zero for the existence of an x satisfying Slater's

condition. Thus, let us suppose that $0 \leq \xi \leq \alpha$ and $0 \leq \xi^* \leq \alpha$.
Next, let us consider the other two inequalities and make the following ansatz.

$$\begin{aligned} w =& \Phi^\intercal (\Phi\Phi^\intercal)^{-1} y \\ b <& \alpha \end{aligned} \tag{7.27}$$

We then have

$$\begin{aligned} -\xi + y - \Phi\Phi^\intercal(\Phi\Phi^\intercal)^{-1}y - b <& 0 \\ -\xi^* - y + \Phi\Phi^\intercal(\Phi\Phi^\intercal)^{-1}y + b <& 0 \end{aligned} \tag{7.28}$$

Hence, we conclude that our problem satisfies Slater's condition. Therefore the solution of the dual and primal problem are equivalent.
We end this section with the derivation of the dual problem; that is the convex problem, we will solve in order to get the quantiles prdeiction of our quantile kernel algorithm.
First, derive the dual function of 7.23.

$$\begin{aligned} g(\alpha, \alpha^*, \nu, \nu^*) &= \inf_x L(x, \lambda) \\ &= \inf_{\xi, \xi^*, w, b} L(w, b, \xi, \xi^*) \end{aligned} \tag{7.29}$$

Setting its derivates to zero

$$\begin{cases} \frac{\partial L}{\partial w} = 0 \implies w = \Phi^\intercal(\alpha - \alpha^*) \\ \frac{\partial L}{\partial b} = 0 \implies (\alpha - \alpha^*)^\intercal \mathbb{1} = 0 \\ \frac{\partial L}{\partial \xi} = 0 \implies Cq\mathbb{1} - \alpha - \nu = 0 \\ \frac{\partial L}{\partial \xi^*} = 0 \implies C(1-q)\mathbb{1} - \alpha^* - \nu^* = 0 \end{cases} \tag{7.30}$$

As pointed out previously, the lower bound resulting from the dual formulation is non trivial only when the lagrange multipliers are $\succeq 0$. Looking at the last two equations of the system 7.30, this implies the following two constraints $\alpha \in [0, Cq\mathbb{1}]$ and $\alpha^* \in [0, C(1-q)\mathbb{1}]$.
Substitute the conditions for an optimum into 7.23, we obtain the dual formulation.

$$\begin{aligned} g(\alpha, \alpha^*) =& \xi^\intercal(Cq\mathbb{1} - \alpha - \nu) + (\xi^*)^\intercal(C(1-q)\mathbb{1} - \alpha^* - \nu^*) - (\alpha - \alpha^*)^\intercal \Phi\Phi^\intercal(\alpha - \alpha^*) \\ &+ (\alpha - \alpha^*)^\intercal y - (\alpha - \alpha^*)^\intercal b + \frac{1}{2}(\alpha - \alpha^*)^\intercal \Phi\Phi^\intercal(\alpha - \alpha^*) \end{aligned}$$

$$\begin{aligned} g(\alpha, \alpha^*) =& 0 + 0 - \frac{1}{2}(\alpha - \alpha^*)^\intercal \Phi\Phi^\intercal(\alpha - \alpha^*) + (\alpha - \alpha^*)^\intercal y - 0 \\ g(\alpha, \alpha^*) =& -\frac{1}{2}(\alpha - \alpha^*)^\intercal \Phi\Phi^\intercal(\alpha - \alpha^*) + (\alpha - \alpha^*)^\intercal y \end{aligned} \tag{7.31}$$

Defining $\mathrm{a} = (\alpha - \alpha^*)$ and letting K the kernel matrix, we have that the dual optimisation problem reads as follows

$$\begin{aligned} \max_{\mathrm{a}} \quad & -\frac{1}{2}\mathrm{a}^\intercal K \mathrm{a} + \mathrm{a}^\intercal y \\ \text{s.t.} \quad & C(q-1)\mathbb{1} \preceq \mathrm{a} \preceq Cq\mathbb{1} \\ & \mathrm{a}^\intercal \mathbb{1} = 1 \end{aligned} \tag{7.32}$$

Switching sign, we rephrase it as a minimisation problem, which is the common practice in convex optimisation.

$$\begin{aligned} \min_{\mathrm{a}} \quad & +\frac{1}{2}\mathrm{a}^\intercal K \mathrm{a} - \mathrm{a}^\intercal y \\ \text{s.t.} \quad & C(q-1)\mathbb{1} \preceq \mathrm{a} \preceq Cq\mathbb{1} \\ & \mathrm{a}^\intercal \mathbb{1} = 1 \end{aligned} \tag{7.33}$$

The kernel quantile regression estimator is then given by

$$f(x) = \sum_i \mathrm{a}_i k(x_i, x) + b \tag{7.34}$$

Since our optimisation problem possesses strong duality and it is differentiable in both the objective and the constraint, we have that it must satisfy the Karush Kuhn Tucker conditions, see section 5.5.3 [15]. Thanks to the KKT conditions on the primal optimisation problem we have that $f(x_i) = y_i$ for $\mathrm{a}_i \notin \{C(q-1), Cq\}$. To see this, we have to consider the KKT conditions.

$$\begin{aligned} \lambda^* r_i(x^*) &= 0, \quad i = 1, \dots, m \\ \nabla L(x^*) &= 0 \end{aligned} \tag{7.35}$$

In our setting we have

$$\begin{aligned} \alpha_i(\xi_i - y_i + r_i) &= 0 \\ \alpha_i^*(\xi_i^* + y_i - r_i) &= 0 \\ \nu_i \xi_i &= 0 \\ \nu_i^* \xi_i^* &= 0 \\ \nabla L &= 0 \end{aligned} \tag{7.36}$$

Using the gradient of the lagrangian of equation 7.30 we end up with

$$\begin{aligned} \alpha_i(\xi_i - y_i + r_i) &= 0 \\ \alpha_i^*(\xi_i^* + y_i - r_i) &= 0 \\ (Cq - \alpha_i)\xi_i &= 0 \\ (C(1-q) - \alpha_i)\xi_i^* &= 0 \end{aligned} \tag{7.37}$$

Now, let us break into cases

$$\begin{cases} \alpha_i = Cq,\ \alpha_i^* = 0 & \implies \alpha_i - \alpha_i^* = Cq, \xi_i \leq 0, \xi^* = 0 \implies \xi_i - y_i + f_i = 0 \\ \alpha_i = 0,\ \alpha_i^* = C(1-q) & \implies \alpha_i - \alpha_i^* = C(q-1), \xi_i = 0, \xi^* \leq 0 \implies \xi_i^* + y_i - f_i = 0 \\ 0 \leq \alpha_i \leq Cq,\ 0 \leq \alpha_i^* \leq C(1-q) & \implies \xi_i = 0, \xi_i^* = 0 \implies -y_i + f_i = 0,\ y_i - f_i = 0 \end{cases} \tag{7.38}$$

Therefore in order to retrieve b we simply have to choose an index i such that $a_i \notin \{C(q-1), Cq\}$ and let

$$b = y_i - \sum_i a_i k(x_i, x) \tag{7.39}$$

7.4.2 Weather quantiles

In order to get acquainted with the inner workings of the presented methods, this section covers an application explaining practical details and comparing results.
The dataset used is the Melbourne daily maximum temperatures [53]. It contains the daily maximum temperatures in Melbourne, Australia, from 1981-1990, excluding leap days, see figure 7.6. Due to the bimodality of the data, such dataset is commonly used to give a difficult quantile regression problem [52], thus why we chose it. The observed bimodality is that a hot day is likely to be followed by either an equally hot day or one much cooler. Hereafter, the results of the four presented methods on the Melbourne dataset are reported, see figure 7.7 for a visualisation. Hyperparameter tuning have been optimised through cross validation, see A.3.

Table 7.1: Pinball loss Melbourne data

	Linear qr	Gbm qr	Quantile forest	Kernel qr
Pinball loss	11.278895	10.317612	10.340842	10.031708

As already pointed out, the quantile regression with q=0.5 corresponds to the standard regression problem, hence we can compare the proposed methods also in terms of the mean absolute error. From these tables, we can see that kernel quantile regression outperformed the simple quantile regressor as well as the more complex models like quantile forest and gradient boosting quantile regression for the Melbourne temperatures dataset. Not only kernel quantile regression was the best in terms of total pinball loss but also the best in terms of each quantile pinball loss and mean absolute error. Comparison has been carried out on further datasets, yielding similar conclusion as the one of above, to know more see A.2.

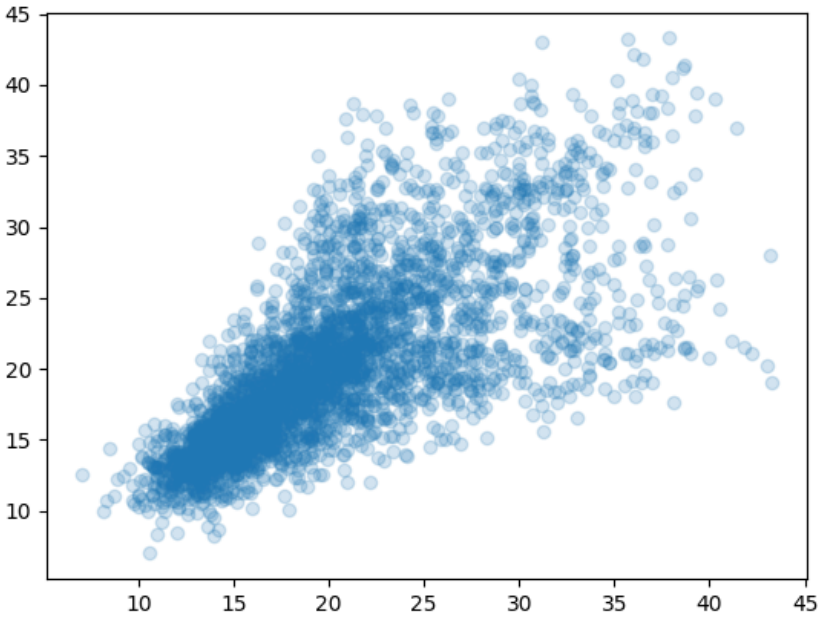


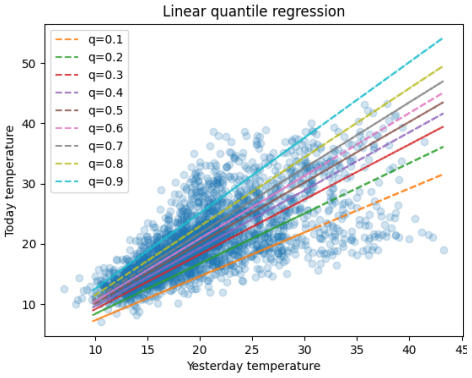
Figure 7.6: Melbourne temperatures dataset

Table 7.2: Pinball loss quantile-wise Melbourne data

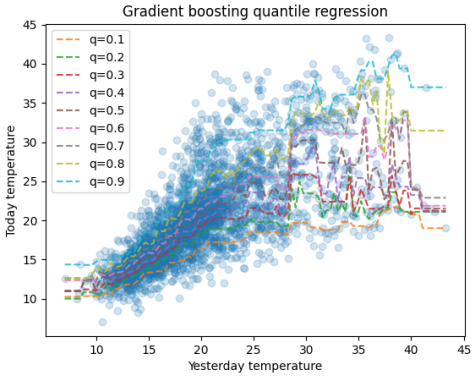
Quantiles	Linear qr	Gbm qr	Quantile forest	Kernel qr
0.100000	0.710644	0.549232	0.562888	0.540235
0.200000	1.155014	0.938561	0.946712	0.903213
0.300000	1.417805	1.212671	1.222407	1.173803
0.400000	1.540108	1.399925	1.409293	1.367943
0.500000	1.574957	1.517281	1.484589	1.455849
0.600000	1.525114	1.498608	1.495474	1.447470
0.700000	1.397918	1.372183	1.362173	1.331503
0.800000	1.170140	1.115077	1.123195	1.096606
0.900000	0.787195	0.714075	0.734112	0.715085

Table 7.3: Mean absolute error Melbourne data

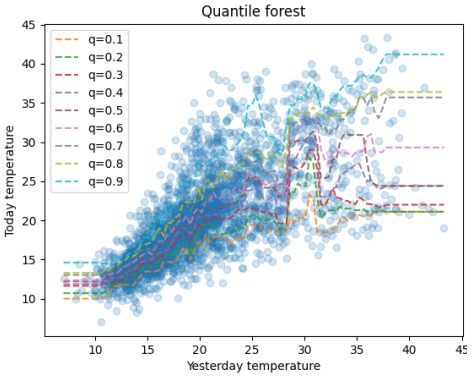
	Linear qr	Gbm qr	Quantile forest	Kernel qr
MAE	3.253882	3.134805	3.095041	3.024336



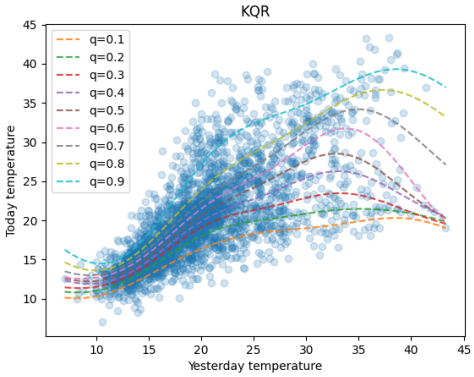
(a) Linear quantile regression



(b) Gradient boosting quantile regression



(c) Quantile forest



(d) Kernel quantile regression

Figure 7.7: Quantile regressors

Chapter 8

Exploratory Analysis and Data ETL

- Explain how data has been retrieved.

- Data provider EPEX(entsoe retrieves its data)

- Explain the ETL(Extract Transform Load) pipeline I set up.

- Explore data in order to get useful insights for how to tune the models to get the most of them.

- correlation between temperatures and load

- Correlation and auto correlation plots

- Split train and test dataset Explain carefully why it is important to carry out an out of sample test and not in sample. In sample test involves look ahead bias because we are fitting the model on the data we want to predict, thus it overfits on the data considered but it does not generalize well.

This section covers the datasets used in our experiments, attributes and features will be thoroughly described. Next, we will explain the ETL pipeline that we set up in order to ease the workflow of our comparison studies. Finally, in order to to bettern understand patters within the data, we carry out an exploratory data analysis.

8.1 Dataset

The chosen datasets for probabilistic load forecasting, were the data from the GEFCom 2014 competition. The data is freely hosted on Dr. Hong blog [48]. The main reason was that, these dataset are considered an a excellent test case for comparing predictive models between the EF community. Additionally, the scores of the competing models are freely available, this enables us to carry out a clean and transparent comparative study. The GEFCom 2014 consisted of four tracks: price, load, wind power and solar power forecasting.

The GEFCom2014 folder contains a zip file for each of the four tracks. This thesis work is focuesed on providing forecasts for the load and price quantities.

8.1.1 Price track

In the price forecasting track, the goal was to predict the electricity price for the next 24 hours of a single zone. The data can be found in the GEFCom2014-P_V2 zip file alongside a set of instructions and the benchmark forecasts. This data consisted of time series for the locational marginal price, for the zonal load and for the system load. The data covers the time interval ranging from the 1st of January 2011 to the 15th of June 2013, see figure 8.1. Additionally, as the competition went on, the real observed data of the previous tasks were made available. The fifteen target days ordered by task number are: 16/06/2013, 17/06/2013, 24/06/2013, 04/07/2013, 09/07/2013, 13/07/2013, 16/07/2013, 18/07/2013, 19/07/2013, 20/07/2013, 24/07/2013, 25/07/2013, 07/12/2013, 08/12/2013.

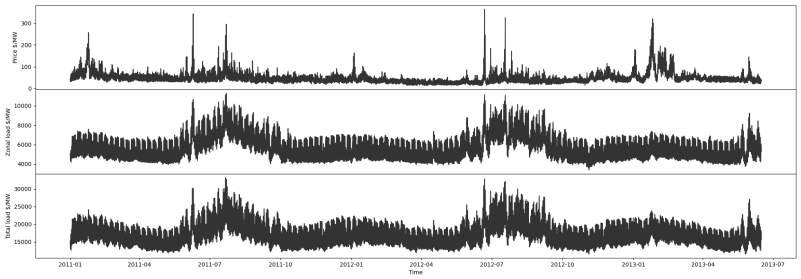


Figure 8.1: Price track

8.1.2 Load track

In the load track, contestant were asked to provide one month ahead hourly probabilistic forecasts on a rolling basis for 15 consecutive rounds. To get started, in the first round, organizers provided 69 months of hourly load data (from 01/01/2005 to 30/10/2010) and 117 months of hourly temperature data (from 01/01/2001 to 30/10/2010). As for the price track, the true observed data from the previous track were made available as the competition progressed.
The data for load forecasting is contained in the GEFCom2014-L_V2 zip file. Within this subfolder, we a txt with the competition instructions and 15 folders one for each of the consecutive tasks. For each of those we have the prediction of the competition benchmark model and the train file upon which

we will fit our model. Each data from the various task folders are shifted by one months between others.
In order to compare our model performance with the winning entries of the GEFCom2014 load competition, we will refer to the Provisional_Leaderboard_V2 file contained also in the GEFCom2014 Data directory.
Notice, the pinball scores for the load track are stored inside the subtab L-score-0/L-score-2 of Provisional_Leaderboard_V2.

8.2 ETL

8.3 EDA

Chapter 9

Implementation

This section is inteneded to explain and aid for reproducibilty studies. Hereafter the specific libraries used and the custom implementatios are thoroughly documented.
For the list of python packages needed see the requirement.txt file.

Section documenting code - indicate computer specifics

- Explain how methods' implementation has been adapted to my specific setting.

- Explain in detail how to my src code has been implemented its rationale and how to use it.

- As I explain code scripts go over the test, to explain better my ideas.

- Indicate also hyperparameters maybe in each subsection

9.1 Point forecasting

9.1.1 Multiple linear regression

For multiple linear regression the one from the sklearn library has been used.

9.1.2 Tbats

Tbats implementation is available at `https://github.com/intive-DataScience/tbatsx`. In our application we specified as hyperparameters the lenght of seasons; 24 for the daily seasonality and 168 for the weekly seasonality.

9.1.3 Prophet

The prophet models has been applied by employing the python API provided by Meta prophet.

9.1.4 K-nearest neighbours

The object KNeighborsRegressor of the sklearn module neighbors has been used.

9.1.5 Support vector regression

The object SVR of the sklearn module svm has been used by specifying the liner kernel.

9.1.6 LSTM

The LSTM predictor has been built using the torch library.

9.1.7 Kernel ridge regression

The object KernelRidge of the sklearn module kernel_ridge has been used.

9.1.8 Kernel support vector regression

The object SVR of the sklearn module svm has been used by specifying rbf as the kernel parameter.

9.2 Probabilistic forecasting

9.2.1 Linear quantile regression

The implementation of statsmodels.regression.quantile_regression.QuantReg has been used. The model is fitted through iterative reweighted least squares.

9.2.2 Quantile gradient boosting machine

The implementation of sklearn.ensemble.GradientBoostingRegressor has been used.

9.2.3 Quantile forest

The implementation of quantile_forest.RandomForestQuantileRegressor has been used. This estimator is compatible with scikit-learn.

9.2.4 Kernel quantile regression

Kernel quantile regression had no previously implemented open source library, thus the need of implementing our own version.
The scikit-learn team provides a project template for the creation of estimators

compatible with scikit-learn functionalities. Therefore, the KQR class is derived from the scikit-learn BaseEstimator and the mixin class RegressorMixin. Our KQR class is initialized by providing a quantile, the regularization term C and the rbf kernel bandwidth gamma; the latter are the two hyperparameters of our custom estimator.
In the fit method, we set up and solve the convex optimisation problem by using the cvxopt library. When using this library, it is important to keep two things in mind. First this library assumes the quadratic term of the optimisation problem to be multiplied by the 0.5 factor, thus we just have to provide the Q matrix with no 0.5 in front. Secondly, in order to specify multiple inequalities we have to stack them an provide as a unique matrix.
Once a solution to the convex problem has been found, we create a mask for the support vectors of the estimator in order to estimate the constant term of our kernel quantile regressor.
In the predict method, we pass a matrix X_eval of independent variables, next we compute the kernel matrix between X_train and X_eval and obtain y_eval with the formula $y = \alpha^\intercal K + b$.
This estimator is compatible with useful scikit-learn in built methods like gridsearch, crossvalidation and scoring rules.
Up to now, there are only two open source implementation of the quantile kernel regression. Nevertheless they are both in R, that is there exists no python, matlab or julia open source implementation. Following are reported the results of a comparative study between our own implementation and the one of the R library kernlab

Python versus R implementation

In this section, a comparison study has been carried out in order to inspect the competitiveness of our implementation with the existing one for the R programming language.

Chapter 10

Experiments Analysis

Analysis of experiments and results
Building on the theory introduced in 6 and 7, this section covers the experiments carried out and the results obtained.

10.1 Point forecasting

This section carries out a comparative study between the state of the art methods for point forecasting, introduced in 6. As use case, we will consider the task of predicting the electric load from the GEFCom 2014 dataset. In such setting we considered the following regressors

- Day
- Hour
- Month
- Day of the week
- Is holiday
- Weather temperature

Methods will be compared by means of the RMSE, MAE and MAPE scores, see section 4.

10.1.1 Multiple linear regression

To get started, standard multiple linear regression has been applied, see fig 10.1 for a visualisation.

The resulting RMSE is 30.59. What can be concluded, is that multiple linear regression is capable of catching the daily seasonality. Nevertheless, it cannot catch the range of the price series properly.

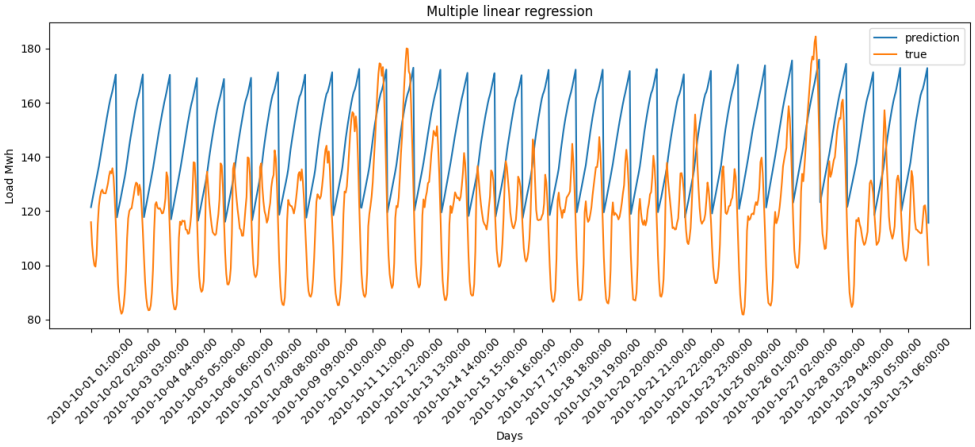


Figure 10.1: Multiple linear regression prediction

10.1.2 Trigonometric seasonality Box-Cox transformation ARMA errors trend and seasonal components

Next, we tried with the autoregressive approach. Unfortunately, AR, ARIMA and SARIMA models did not perform as expected, their output was slightly better than the one of linear regression. This is probably due to the fact that, the data considered entails two kinds of seasonalities, while ARIMA models can only handle one at a time. We remind the reader, that the electricity load time series involves both daily and weekly seasonalities. Hence, the need for a more advanced time series model. The Tbats [25] model is a forecasting method capable of handling complex patterns in the data. Its name stands for trigonometric seasonality, Box-Cox transformation, ARMA errors, trend and seasonal components. Tbats forecast is visualized in figure 10.2, meanwhile its RMSE is 15.08. From the plot we see that Tbats is capable of catching the lows and average trend, conversely it has some difficulties handling the price peaks.

10.1.3 Prophet

Following, the prophet model has been considered. We get started by considering the base implementation. In such setting, prophet takes in as input the time series object and learns its data generating process. This method achieves a RMSE of 23.96, its prediction is visualized in figure 10.3. What can be seen is that, prophet models correctly the average trend but does not model peaks and lows precisely. Next, a more complex model was trained. We added the weather temperature, the square of it and the categorical variable for holidays effect as regressors. Furthermore, we also applied a log transformation to the dependet variable. Doing so, RMSE went down to 10.29. Forecast is visualized in 10.4, moreover, figure 10.5 and figure 10.6

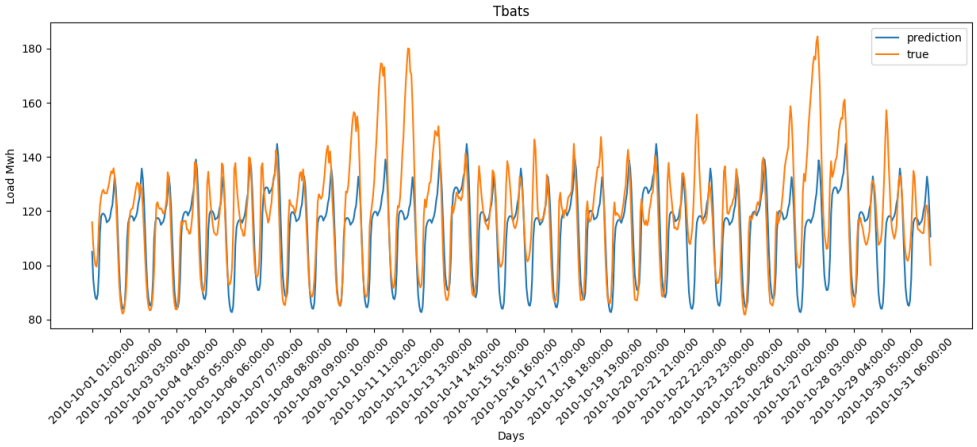


Figure 10.2: Tbats prediction

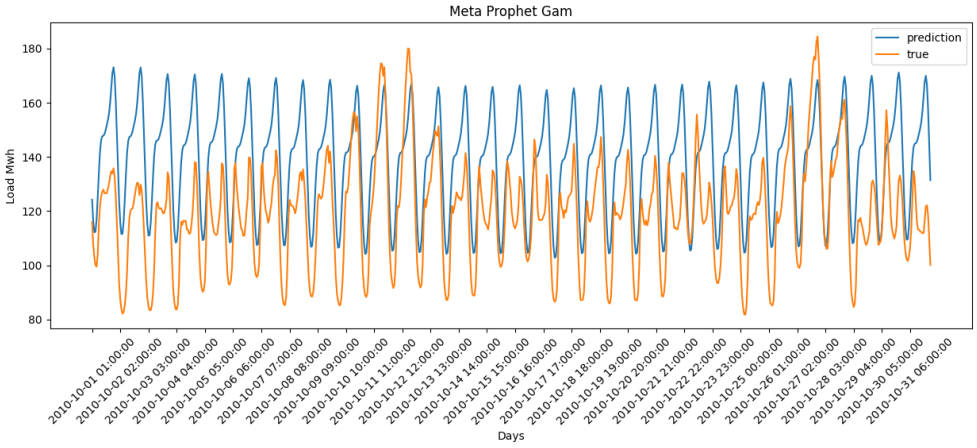


Figure 10.3: Prophet prediction

break down the trend and the seasonalities respectively.

Prophet alongside with the right features is a good model in the context of electricity price forecasting, it is capcable of catching the trend, the peak and the lows.

10.1.4 K-nearest neighbours

Afterwards, K-nearest neighbours has been considered. In doing so, we cross validated the number of neighbours to get the best model instance. The forecast is visualized in figure 10.7, RMSE is 12.54. What it can be said is that, K-nearest neighbour is capable of predicting prices well by averaging past data.

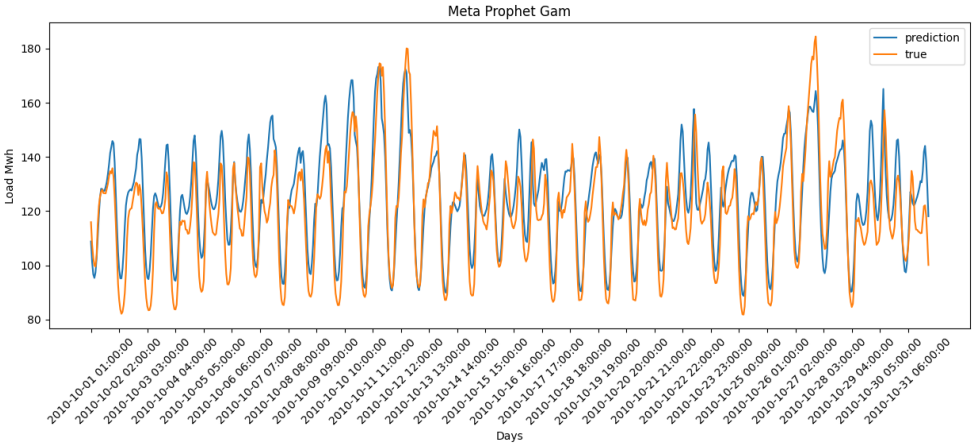


Figure 10.4: Prophet v2 prediction

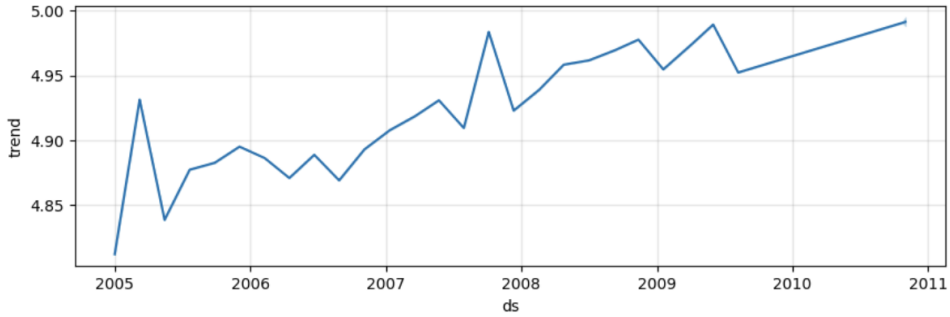


Figure 10.5: Seasonalities breakdown

10.1.5 Support vector regression

Coming up we have support vector regression. In applying this model we used gridsearch crossvalidation to search for the best regularization parameter C. Support vector regression achieves an RMSE of 23.24, for the prediction see figure 10.8. Visually, we see that the SVR performs similar to multiple linear regression, as expected. Like multiple linear regression, SVR can model the daily seasonality but the point prediction is off in terms of highs and lows.

10.1.6 Long short term memory

We used the torch library in order to code our LSTM regressor. Various hyperparameters combinations were tried during hyperparameter tuning. The final hyperparameters we set on follows.

- hidden_size= 64
- num_layers= 2

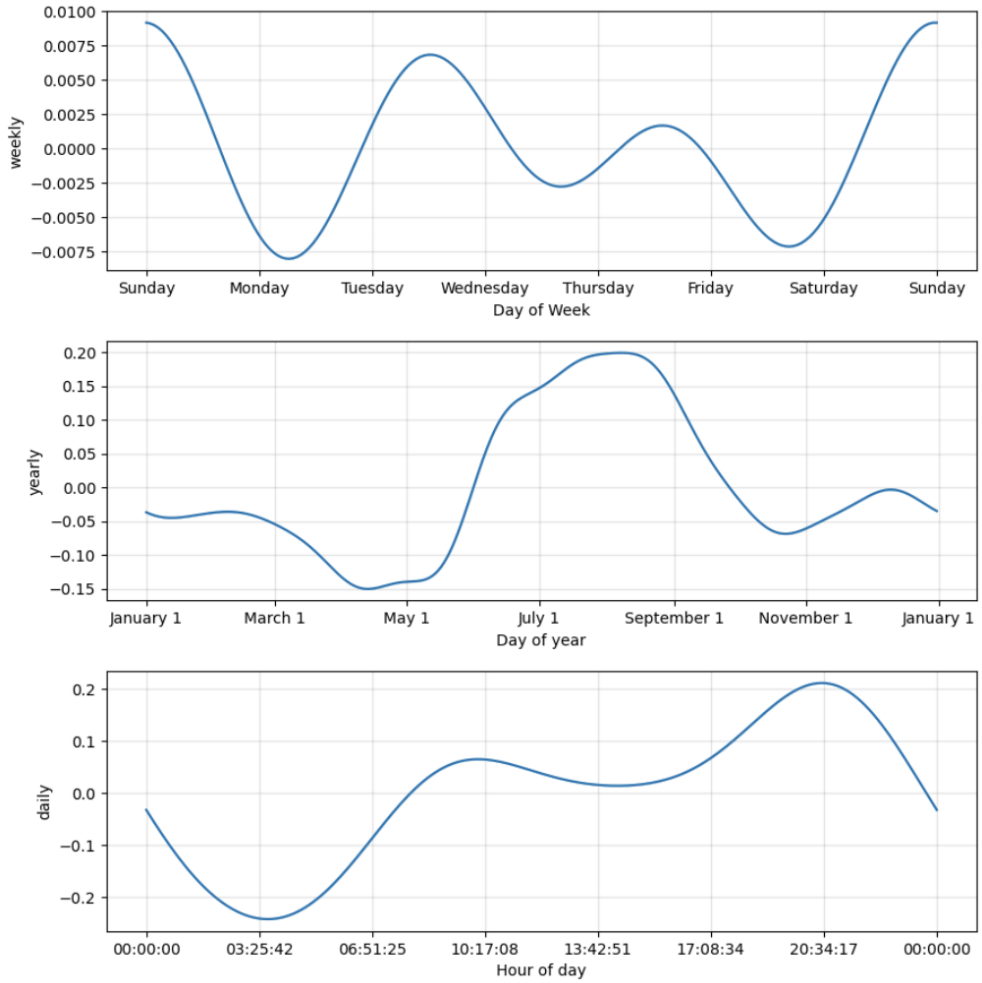


Figure 10.6: Prices trend

- output_size= 1
- num_epochs= 30
- learning_rate= 0.001
- batch_size= 32
- window_size= 24

LSTM forecast is reported in table 10.9, it achieves a RMSE of 9.1782

10.1.7 Kernel ridge regression

Subsequently, kernel ridge regression was considered. The kernel considered is the radial basis gaussian function. Cross validation was carried jointly over

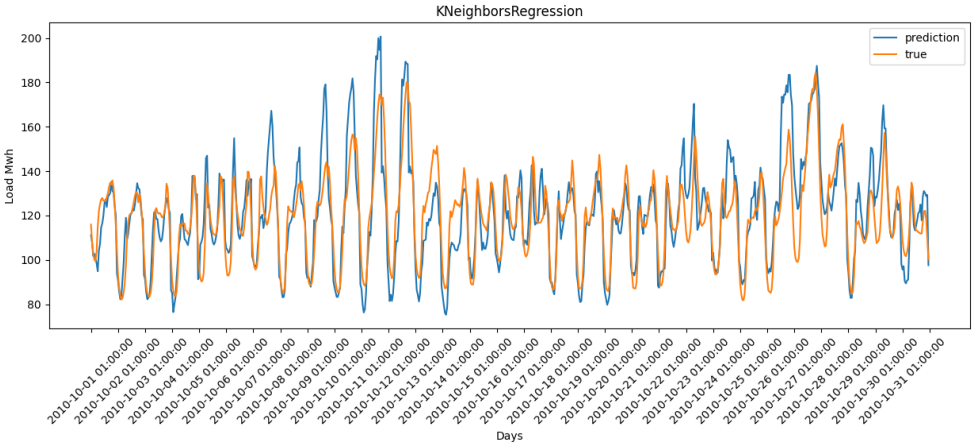


Figure 10.7: K-nearest neighbour regression

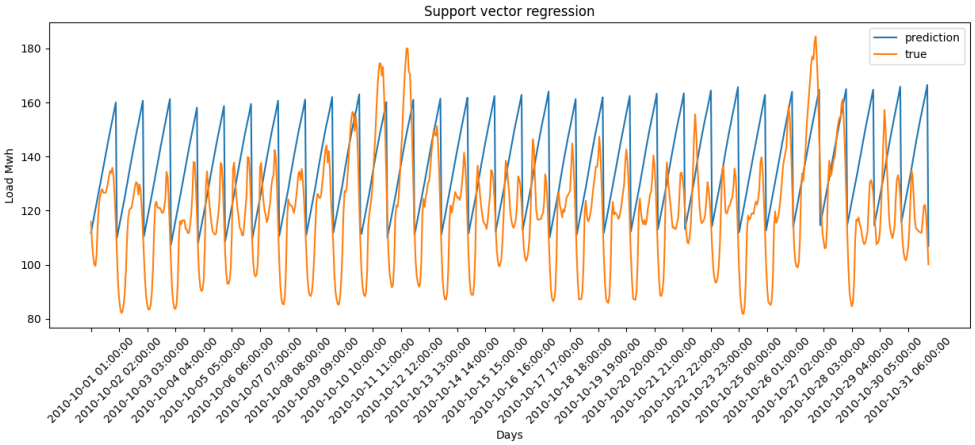


Figure 10.8: Support vector regression prediction

the rbf kernel bandwith and the regularization constant. The RMSE achieved is 9.86, figure 10.10 reports the prediction. We can observe that, kernel ridge regression models accurately the electricity time series.

10.1.8 Kernel support vector regression

The last model we considered in this setting was the kernel support vector regression. RMSE achieved with this method is 9.10, forecast is reported in figure 10.11.

We can observe that kernel support vector regression is one of the best performing techniques between the one considered.

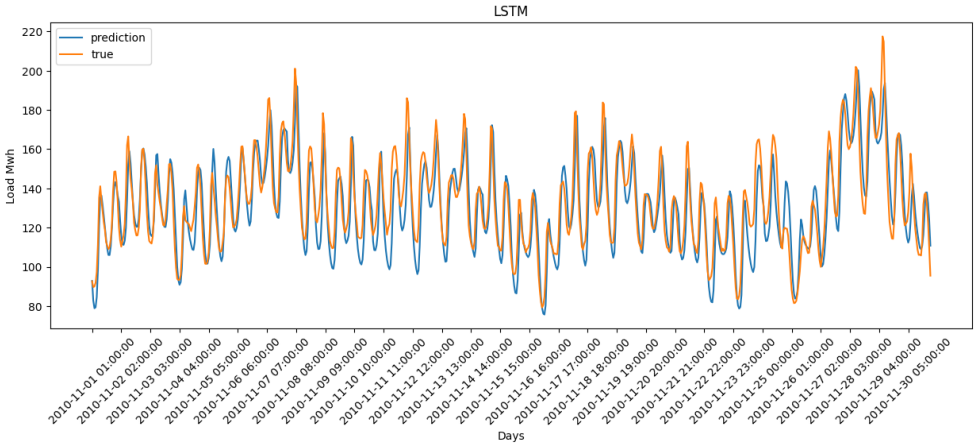


Figure 10.9: Long short term memory prediction

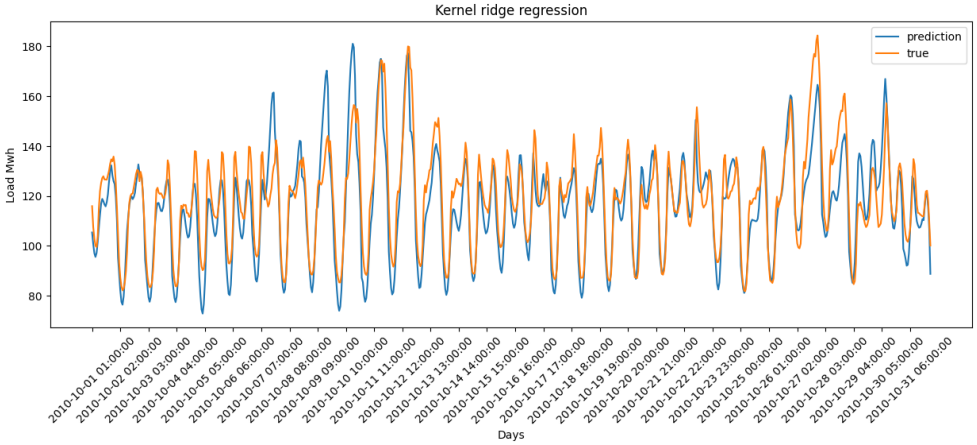


Figure 10.10: Kernel ridge prediction

10.1.9 Results

This section reports the table comparing the considered methods scores

Table 10.1 reporst the RMSE scores for each of the considered methods. It can be seen that kernel based methods are the ones achieving the lowest RMSE; specifically, we have KrnSVR, KrnRidge and KNN being among the top methods on the majority of tasks.

The MAE scores are contained in table 10.2. Similarly to above, we can conclude that kernel methods stand out for achieving also the lowest MAE score.

Finally table 10.3 reports the MAPE score for each method. What can be concluded is that kernel regressors are also the best in terms of MAPE score; with KrnSVR being the top one.

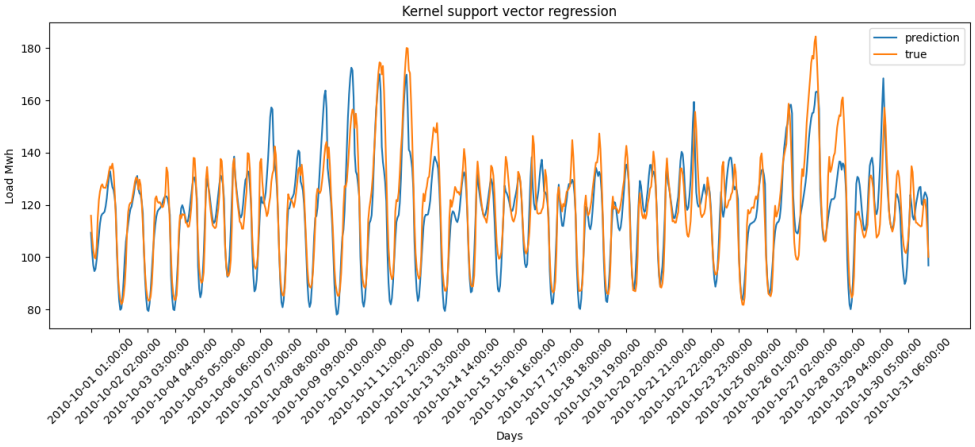


Figure 10.11: Kernel support vector prediction

Table 10.1: Root mean squared errors

Method/RMSE	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6	Task 7	Task 8	Task 9	Task 10	Task 11	Task 12	Task 13	Task 14	Task 15
MLR	30.5892	29.0347	75.1682	63.1376	42.1156	36.6233	39.2496	38.8286	47.1307	68.0494	61.6005	30.3751	34.5523	33.4287	33.5581
TBATS	15.0886	31.9720	99.1260	84.4890	51.9740	34.9055	18.4445	38.3246	74.1117	98.6600	84.3050	38.8433	16.6080	28.9902	41.6194
Prophet	10.2936	14.4358	38.7551	63.4787	19.7474	17.5065	12.6926	14.2665	17.5466	23.5944	43.6666	20.8637	16.9493	19.1626	23.3889
KNN	12.5429	11.6699	24.5057	18.3310	13.1821	**11.9238**	12.0044	14.5165	16.3132	15.1831	37.6457	16.4690	12.0324	11.3102	14.1717
SVR	23.2421	25.8525	78.8043	67.6209	42.3748	32.7845	30.5971	35.3660	55.4213	77.9660	68.5799	30.2451	27.0345	28.3652	32.1480
LSTM	9.1782	12.0669	**22.7048**	**16.2087**	16.0964	12.7936	**10.8559**	14.6173	19.7303	18.0200	43.2051	17.1856	10.3106	12.1347	17.5849
KrnRidge	9.8619	11.6101	37.7824	35.2459	**12.5160**	14.5911	12.8791	17.4385	16.1131	17.1938	37.6961	14.0076	9.8441	10.7491	13.2975
KrnSVR	**9.1028**	**11.3117**	22.9281	19.2132	12.6331	12.6018	11.3537	**12.9506**	**14.9731**	**11.7765**	**37.3797**	**13.2694**	**8.8522**	**10.6185**	**13.2602**

Table 10.2: Mean absolute errors

Method/MAE	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6	Task 7	Task 8	Task 9	Task 10	Task 11	Task 12	Task 13	Task 14	Task 15
MLR	27.7219	24.3823	65.4016	52.3873	34.8465	31.2663	35.9420	34.7316	37.0273	54.7075	48.9327	26.1975	31.8720	29.2306	28.3428
TBATS	10.5408	23.8417	91.3153	74.5734	40.3258	26.0092	14.2141	24.7818	62.0452	86.7125	72.6912	28.4235	11.1184	21.1038	31.8555
Prophet	8.6139	11.4307	28.3509	46.4480	15.6290	14.1278	10.1574	11.2360	14.0072	18.9565	27.1142	16.9678	14.6615	16.1750	18.3043
KNN	9.4003	9.1126	19.6213	14.4906	10.6461	**9.4354**	8.7441	10.0065	12.4714	11.2006	20.3087	12.5416	9.2558	8.5937	10.5451
SVR	20.2443	21.2526	69.3704	57.1107	34.3651	27.5875	27.4651	29.4204	43.7009	64.3522	55.4307	25.1080	24.0294	24.4049	26.5869
LSTM	7.5217	9.4767	**17.9851**	**13.2158**	11.8118	9.8607	**8.0561**	10.8480	15.1008	13.8108	25.9906	13.3588	8.1526	9.8000	14.3186
KrnRidge	7.5965	9.2130	28.2484	24.2366	**10.0589**	10.7758	9.5788	11.2166	12.3427	12.3306	20.0213	10.8434	7.1625	**7.9710**	10.0767
KrnSVR	**6.9581**	**8.6989**	18.7739	15.1398	10.1395	9.5927	8.5731	**9.0033**	**11.0618**	**8.5486**	**19.7383**	**10.5545**	**7.0395**	8.1926	**9.8125**

Table 10.3: Mean absolute percentage errors

Method/MAPE	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6	Task 7	Task 8	Task 9	Task 10	Task 11	Task 12	Task 13	Task 14	Task 15
MLR	0.2470	0.1935	0.3030	0.2601	0.2474	0.2667	0.3498	0.3058	0.1954	0.2432	0.4234	0.2114	0.2926	0.2502	0.2145
TBATS	0.0833	0.1650	0.4285	0.3731	0.2482	0.1849	0.1234	0.1634	0.3181	0.4021	0.4880	0.1761	0.0905	0.1555	0.2058
Prophet	0.0737	0.0854	0.1324	0.2453	0.1134	0.1169	0.0938	0.0912	0.0880	0.0992	0.3757	0.1389	0.1343	0.1356	0.1370
KNN	0.0778	0.0682	0.0924	0.0761	0.0751	0.0740	0.0784	0.0744	0.0725	0.0555	**0.3115**	0.0888	0.0816	0.0699	0.0765
SVR	0.1808	0.1640	0.3211	0.2829	0.2317	0.2260	0.2676	0.2451	0.2215	0.2867	0.4354	0.1870	0.2216	0.2044	0.1933
LSTM	0.0642	0.0706	0.0896	**0.0722**	0.0798	0.0772	**0.0710**	0.0854	0.0909	0.0741	0.3578	0.0976	0.0719	0.0778	0.1037
KrnRidge	0.0625	0.0685	0.1281	0.1207	0.0718	0.0797	0.0812	0.0760	0.0696	0.0587	0.3127	0.0779	**0.0591**	**0.0637**	0.0722
KrnSVR	**0.0568**	**0.0642**	**0.0892**	0.0788	**0.0714**	**0.0724**	0.0740	**0.0665**	**0.0629**	**0.0428**	0.3122	**0.0781**	0.0607	0.0646	**0.0713**

10.2 Probabilistic forecasting

In this subsection we compare performance of quantile regressors.
- experiment for quantile estimator on gefcom2014 and we are good

List of Symbols

AIC	Akaike information criterion
ANN	Artificial neural network
ACF	Autocorrelation function
AR	Autoregressive model
ARMA	Autoregressive moving average model
ARIMA	Autoregressive integrated moving average model
ARX	Autoregressive exogenous model
BIC	Bayesian information criterion
CI	Computational intelligence
CKD	Conditional kernel density
CRPS	Continous ranked probability score
DDNN	Distributional neural network
EDA	Exploratory data analysis
EF	Electricity forecasting
ELF	Electricity load forecasting
EMD	Empirical mode decomposition
EPF	Electricity load forecasting
ESM	Exponential smoothing models
ETL	Extract transform load
EVs	Electric vehicles
GAMs	Generalized additive models
GBRT	Gradient boosting regression tree
GEFCom	Global energy forecasting competition
GPR	Gaussian process regression
HWT	Holt-Winters-Taylor exponential smoothing method
KDE	Kernel density estimation
LMP	Load marginal price

LSTM	Long short term memory
LCTs	Low carbon technologies
LSSVR	Least squares support vector regression
LV	Low voltage
MCP	Market clearing price
MCV	Market clearing volume
MAE	Mean absolute error
MASE	Mean absolute scaled error
MAPE	Mean absolute percentage error
MGM	Minimal gated memory network
MA	Moving average
MLR	Multiple linear regression
NARX	Nonlinear autoregressive exogenous models
OLS	Ordinary least squares
RF	Random forest
RKHS	Reproducing kernel hilbert space
RMSE	Root mean squared error
PACF	Partial autocorrelation function
PX	Power exchange
PI	Prediction interval
PF	Price Forecasting
PPF	Probabilistic Price Forecasting
QR	Quantile regression
QRA	Quantile regression averaging
SME	Small and medium-sized enterprises
SNARX	Smoothed nonparametric ARX
SSE	Sum of squared errors
SVMs	Support vector machines
SDGs	Sustainable development goals
TOU	Time of use tariffs
TSO	Transmission system operator
ZMCP	Zonal market clearing price
d^*	dual optimum
$\mathbb{E}$	expectation
$\Phi(x)$	feature matrix
F	feature space
$\varphi(x)$	feature vector
$g(v)$	dual function
$\mathbb{I}$	Indicator function
K	kernel matrix
$k(x, x')$	kernel function
L	lagrangian
ρ_q	pinball loss
p^*	primal optimum

relint	relative interior
$\mathcal{H}$	reproducing kernel hilbert space
$\odot$	Hadamard product
$\mathbb{1}$	vector of ones

Appendix A

Appendix

A.1 Feature map normalization

Proof

$$\begin{aligned}
\|\varphi^{new}(x)\|_{\mathcal{H}}^2 &= \left\| \frac{\varphi(x)}{\|\varphi(x)\|_{\mathcal{H}}} \right\|_{\mathcal{H}}^2 \\
&= \left\| \frac{\varphi(x)}{\sqrt{k(x,x)}} \right\|_{\mathcal{H}}^2 \\
&= \langle \frac{\varphi(x)}{\sqrt{k(x,x)}}, \frac{\varphi(x)}{\sqrt{k(x,x)}} \rangle_{\mathcal{H}} \\
&= \frac{1}{\sqrt{k(x,x)^2}} \langle \varphi(x), \varphi(x) \rangle_{\mathcal{H}} \\
&= 1
\end{aligned}$$

□

A.2 Quantile regressor extensive comparison

This section contains extensive comparison between our kernel quantile regression and other state of the art quantile regressors. We benchmark it on popular machine learning datasets.

A.2.1 Boston housing dataset

The Boston housing dataset https://www.kaggle.com/datasets/altavish/boston-housing-dataset contains information about various attributes for suburbs in Boston. There are 13 indipendent variables:

- CRIM per capita crime rate by town
- ZN proportion of residential land zoned for lots over 25,000 sq.ft.

Table A.1: Pinball loss Boston housing data

	Linear qr	Gbm qr	Quantile forest	Kernel qr
0	13.785678	11.418540	10.587686	10.297572

Table A.2: Pinball loss quantile-wise Boston data

	Linear qr	Gbm qr	Quantile forest	Kernel qr
0.100000	0.729749	0.771714	0.588441	0.578898
0.200000	1.122582	1.033442	0.932824	0.869145
0.300000	1.479486	1.170642	1.153765	1.142783
0.400000	1.712577	1.436263	1.352667	1.331955
0.500000	1.911385	1.344361	1.408333	1.396300
0.600000	1.989514	1.448885	1.464902	1.431705
0.700000	1.938362	1.508741	1.427912	1.382772
0.800000	1.658058	1.497901	1.275059	1.245288
0.900000	1.243965	1.206591	0.983784	0.918725

Table A.3: Mean absolute error Boston data

	Linear qr	Gbm qr	Quantile forest	Kernel qr
0	3.826326	2.845989	2.965490	2.810494

- INDUS proportion of non-retail business acres per town.
- CHAS Charles River dummy variable (1 if tract bounds river; 0 otherwise)
- NOX nitric oxides concentration (parts per 10 million)
- RM average number of rooms per dwelling
- AGE proportion of owner-occupied units built prior to 1940
- DIS weighted distances to five Boston employment centres
- RAD index of accessibility to radial highways
- TAX full-value property-tax rate per 10,000
- PTRATIO pupil-teacher ratio by town
- B $1000(Bk - 0.63)^2$ where Bk is the proportion of afroamericans by town
- LSTAT lower status of the population

The dependent variable is MEDV, that is the median value of owner occupied homes in $1000's

Table A.4: Pinball loss Abalone data

	Linear qr	Gbm qr	Quantile forest	Kernel qr
0	5.613975	5.531938	5.212990	5.252491

Table A.5: Pinball loss quantile-wise Abalone data

	Linear qr	Gbm qr	Quantile forest	Kernel qr
0.100000	0.277903	0.290531	0.274079	0.269287
0.200000	0.469361	0.488079	0.453110	0.457286
0.300000	0.621961	0.625633	0.580766	0.596791
0.400000	0.729757	0.715875	0.689904	0.691310
0.500000	0.794695	0.766185	0.735945	0.740834
0.600000	0.810691	0.785769	0.744928	0.746636
0.700000	0.769587	0.730318	0.700287	0.710392
0.800000	0.667776	0.656913	0.609378	0.608334
0.900000	0.472244	0.472635	0.424593	0.431621

Table A.6: Mean absolute error Abalone data

	Linear qr	Gbm qr	Quantile forest	Kernel qr
0	1.627627	1.574179	1.499522	1.498583

A.2.2 Abalone dataset

The abalone data https://archive.ics.uci.edu/dataset/1/abalone consist of measurements of abalone molluscs, the goal is predicting their age by building a model for estimating its number of rings; age is the number of rings plus 1.5 The data has 8 attributes:

- Sex Categorical variable either male, female or infant
- Length
- Diameter
- Height
- Whole height
- Shucked height
- Viscera weight
- Shell weight

Table A.7: Pinball loss Vehicle data

	Linear qr	Gbm qr	Quantile forest	Kernel qr
0	4.054449	2.289554	2.844410	2.204343

Table A.8: Pinball loss quantile-wise Vehicle data

	Linear qr	Gbm qr	Quantile forest	Kernel qr
0.100000	0.254649	0.139849	0.170489	0.182490
0.200000	0.403772	0.236339	0.285875	0.223165
0.300000	0.548820	0.244086	0.357375	0.242963
0.400000	0.576918	0.263169	0.389305	0.262835
0.500000	0.554367	0.306123	0.410738	0.295878
0.600000	0.563046	0.335363	0.398125	0.306062
0.700000	0.516019	0.287490	0.326572	0.283716
0.800000	0.407742	0.261882	0.313698	0.235793
0.900000	0.229115	0.215253	0.192233	0.171440

Table A.9: Mean absolute error Vehicle data

	Linear qr	Gbm qr	Quantile forest	Kernel qr
0	1.117714	0.606971	0.752197	0.594292

A.2.3 Vehicle dataset

This data contains info about used cars https://www.kaggle.com/datasets/nehalbirla/vehicle-dataset-from-cardekho, the predictors are:

- Year
- Present_price ex showroom price
- Kms Driven
- Fuel type
- Seller type
- Transmission
- Owner number of previous owners

The dependent variable is the selling price.

What can be concluded from these numerical examples is that, on average kernel quantile regression yields better results than quantile forest [66] and gradient boosting machine quantile regression [28] in terms of the pinball loss as well as in terms of the mean absolute error.

A.3 Cross validation

Cross validation directly estimates the expected test error

$$Err = \mathbb{E}\left[L\left(Y, \hat{f}(X)\right)\right] = \mathbb{E}\left[Err_{\mathcal{T}}\right] \tag{A.1}$$

$Err_{\mathcal{T}}$ is the prediction error over an independent test sample

$$\mathbb{E}_{\mathcal{T}} = \mathbb{E}\left[L\left(Y, \hat{f}(X)\right)|\mathcal{T}\right] \tag{A.2}$$

That is *Err* is the average over everything that is random: X,Y and the training set $\mathcal{T}$ used to learn $\hat{f}$. Hence, our interest lies in estimating the *Err* quantity in order to guide model selection.

A.3.1 K-fold cross validation

K-fold cross validation splits the data into K roughly equally sized parts, then K models are trained. For k from 1 to K we train the kth model on the whole dataset except the kth partition. Next, the prediction error of the kth fitted model is computed. Finally averaging all the k prediction erros we obtain an estimate for the expected test error. We select the model which performs best in terms of expected prediction error.
Usually, K is set equal to 5 or 10. Leave-one-out cross validation is the case when K is set equal to the size of the data.

A.3.2 Gridsearch

Model performance depends higly on the choice of hyperparameters. Notice, there is no way to get to know them in advance, therefore, all we can do is trying a lot of combinations until we fit a good enough set of hyperparameters. Essentially, gridsearch carries out hyperparameter tuning by performing a search over a predefined hyperparemeters grid. During its search, it tries all possible combination and evaluates the different models using cross validation. For example, suppose that the grid contains 100 possible candidates and that we are doing 5-fold cross-validation, then the gridsearch algorithm will carry out 500 iterations. Therefore, we get an estimate of prediction error for each considered model and this guides us in hyperparameters (model) selection.

A.3.3 Randomized search

Randomized search controls the number of steps by choosing smartly the hyperparameters to try in each iterations. Let us say, there are 100 candidates and we set the number of iterations to 20 then the search will stop after the 20th iteration and return the best set among the hyperparameters observed.

The advantage is that it is much more quicker than gridsearch, but on the other hand its performance is worse than gridsearch.

A.3.4 Halving gridsearch

Gridsearch has been the to go choice for hyperparameters tuning for the past years. However, such method is brute forcing all possible combinations, hence it is highly computationally intensive, especially when it comes to large datasets.
The scikit-learn team addressed this disadvantage of gridsearch by introducing the halving gridsearch method [5] (2020). Such technique has proved itself to greatly speed up hyperparameter tuning. The ground concept underlying this method is successive halving. During the first iteration of halving gridsearch, all candidates are trained on a small subset of the training set. Next, we keep only the candidates which performed best and compare them again on a bigger subset of the training set. As the iterations pass, the surviving candidates will be given more and more training samples. The algorithm stops when we are left with only the best set of hyperparameters.

A.3.5 Cross validation for time series data

When data points are dependent on preceding values, we cannot use standard K-fold cross validation. The rationale is that K-fold will randomize the order of the data, thus it might happen to use future data to predict the past; we want to avoid such behaviour in a time series setting. When carryout out any kind of cross validation, we must keep consistency in the way we evaluate our predictors during model selection and in the way we perform evaluation of the test data. Hence, for time series crossvalidation we have the timeseries split procedure, see figure A.1 for a visualisation; the blue observations make up the training sets while the orange observations form the test sets. We then average all the observed losses in order to get an estimate for Err. Notice, in the literature this procedure is sometimes also referred to evaluation on a rolling forecasting origin. This comes from the fact that at each iteration we push forward the origin of our forecast. The same concept applies for multi step ahead forecasting, see picture A.2. That is in predicting $\hat{L}_{N+m}$ we use as inputs $L_1, L_2, \ldots, \hat{L}_N, \hat{L}_{N+1}, \ldots, \hat{L}_{N+m-1}$; where $\hat{L}_N, \hat{L}_{N+1}, \ldots, \hat{L}_{N+m-1}$ are one step ahead forecasts.

A.4 Kernel methods best practices

Following, are reported a couple of considerations important to keep in mind when working with kernel methods.

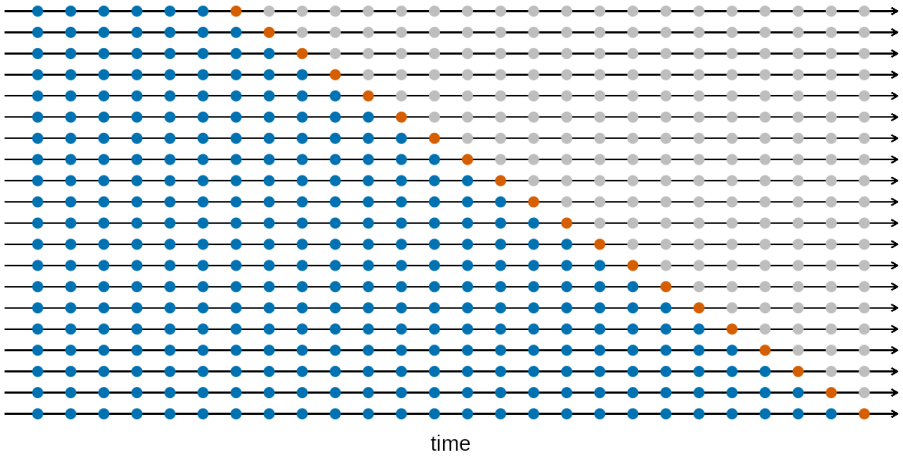


Figure A.1: Cross validation for one step ahead timeseries data [51]

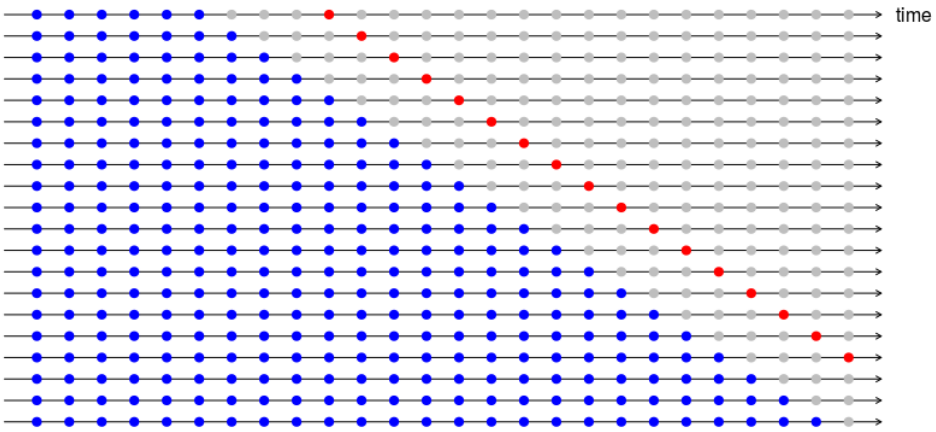


Figure A.2: Cross validation for m step ahead timeseries data [51]

A.4.1 Data normalization

With data normalization we transform the range of the feautures to a standard scale. Such preprocessing step is essential when employing distance based algorithms like svm or k-nearest neighbors. The rationale behind it is that by normalizing data we give an uniform weight to each feature in the learning process; in this way we do not favour larger scale features. Examples of the most popular features scaling are:

- Standard scaler= it computes the standard score z of a sample x z=$\frac{x-\mu}{\sigma}$.
- MinMax scaler= it maps every data sample to the range 0, 1. $\frac{x-\min(X)}{\max(X)-\min(X)}$

- Robust scaler= it scales features using statistics that are robust to outliers. Essentially, it subtracts the median and then scales the data according to the interquantile range.

Many other data scaling algorithms exists, we refer the reader to a thorough comparison [7].
We conclude this subsection with a custom example that motivates the need of feature scaling [3]. The idea is to compare the results of modelling the data with k nearest neighbors on the unscaled data against the scaled data. The considered data is the wine recognition dataset https://archive.ics.uci.edu/dataset/109/wine. The goal for this dataset is recognising from whose cultivator the wine comes based on two features with a completely different scale. The first feature has values in the [0,1000] range while the second feature has values contained in [1,10]. The unscaled and scaled version are compared in figure A.3 What

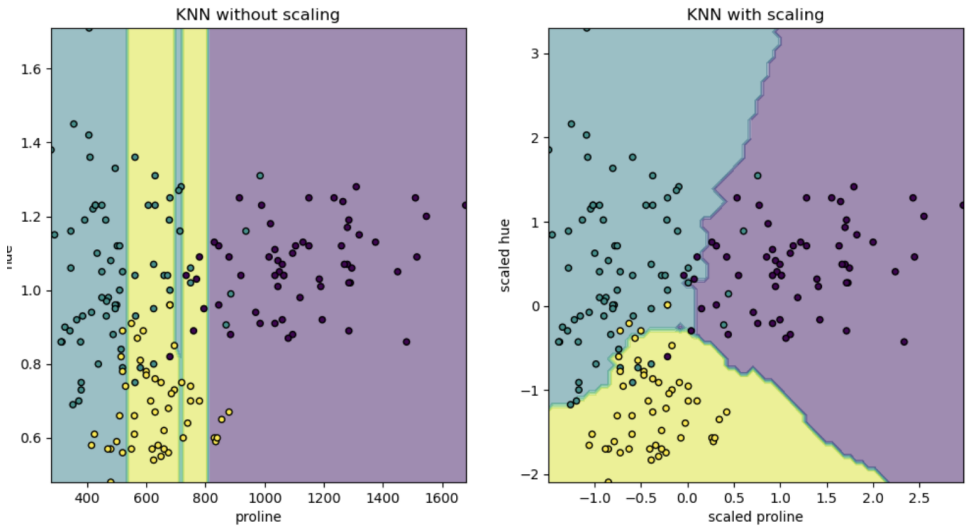


Figure A.3: importance of feature scaling

we can conclude from the image is that, the model trained on scaled data is greatly better than the other. On the left, we can see that distances between categories are impacted solely by the larger scale feature. Conversely, on the right, we have that the two features contributing equally in determining the neighbors.

A.4.2 Data compression

When the number of points n is large, kernel methods suffer from high computational costs. Using kernel methods, we have that the storage cost is of the order of $O(n^2)$ while the computational cost for finding the solution is of the order $O(n^3)$. A significant speed up can be obtained thanks to low

rank approximation.

Nystrom decomposition

The Nyström approximation involves storing a submatrix of the whole kernel matrix. Thus, storage and computational cost are reduced to $O(nm)$ and $O(nm^2)$ respectively. Nyström works by selecting $m < n$ points, callled representative points; it approximates K as

$$\tilde{K} = K_{n,m} K_{m,m}^{-1} K_{n,m}^{\mathsf{T}} \tag{A.3}$$

Pivoted Cholesky decomposition

Pivoted Cholesky approximate the Cholesky decomposition of a matrix. Since kernel matrices are positive definite, they can decomposed in terms of the Cholesky decomposition. Hence, we have that pivoted Cholesky can be used to approximate the full kernel matrix.

A.5 Source code

The whole code for the project is hosted on https://github.com/luca-pernigo/ThesisKernelMethods.

- query: folder containing Scopus data and scripts to generate bibliometric survey plots in section 2
- kqr.py: file implementing our custom kernel quantile regression

Bibliography

[1] Active participation in the development of the electricity marke. https://raport2016.pse.pl/en/active-participation-in-the-development-of-the-electricity-market#national-electricity-market.

[2] GM22TutorialProbabilisticEnergyForecasting. https://resourcecenter.ieee-pes.org/education/tutorials/pes_ed_tut_gm22_0717_pefm_sld.

[3] Importance of feature scaling. https://scikit-learn.org/stable/auto_examples/preprocessing/plot_scaling_importance.html.

[4] Roberts, B. Electricity Price Map in the US, NREL, 2021. https://geocurrents.info/wp-content/uploads/2012/06/electricity-price-map.jpg.

[5] Scikit HalvingGridSearchCV. https://scikit-learn.org/stable/modules/grid_search.html#successive-halving-user-guide.

[6] THE 17 GOALS Sustainable Development - the United Nations. https://sdgs.un.org/goals.

[7] The effect of different scalers on data. https://scikit-learn.org/stable/auto_examples/preprocessing/plot_all_scaling.html#plot-all-scaling-robust-scaler-section.

[8] Ilyas Agakishiev, Wolfgang Karl Härdle, Karel Kozmik, Milos Kopa, and Alla Petukhina. Multivariate probabilistic forecasting of electricity prices with trading applications. *SSRN Electronic Journal*, 01 2023.

[9] A Aizerman. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and remote control*, 25:821–837, 1964.

[10] Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American mathematical society*, 68(3):337–404, 1950.

[11] Siddharth Arora and James W Taylor. Forecasting electricity smart meter data using conditional kernel density estimation. *Omega*, 59:47–59, 2016.

[12] Ludwig Baringhaus and Carsten Franz. On a new multivariate two-sample test. *Journal of multivariate analysis*, 88(1):190–206, 2004.

[13] Ashutosh Bhagwat. Institutions and long term planning: Lessons from the california electricity crisis. *Administrative Law Review*, 55(1):95–125, 2003.

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

[15] Stephen P Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[16] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[17] Leo Breiman. *Classification and regression trees*. Routledge, 2017.

[18] Robert Bringhurst. *The Elements of Typographic Style*. Hartley & Marks, 1996.

[19] Robert Goodell Brown. Statistical forecasting for inventory control. 1959.

[20] Antoine Chatalic, Nicolas Schreuder, Alessandro Rudi, and Lorenzo Rosasco. Nyström kernel mean embeddings, 2022.

[21] Yutian Chen, Max Welling, and Alex Smola. Super-samples from kernel herding. 2012.

[22] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20:273–297, 1995.

[23] Mathieu David, F Ramahatana, Pierre-Julien Trombe, and Philippe Lauret. Probabilistic forecasting of the solar irradiance with recursive arma and garch models. *Solar Energy*, 133:55–72, 2016.

[24] Jan G De Gooijer and Rob J Hyndman. 25 years of time series forecasting. *International journal of forecasting*, 22(3):443–473, 2006.

[25] Alysha M De Livera, Rob J Hyndman, and Ralph D Snyder. Forecasting time series with complex seasonal patterns using exponential smoothing. *Journal of the American statistical association*, 106(496):1513–1527, 2011.

[26] Grzegorz Dudek. Probabilistic forecasting of electricity prices using kernel regression. In *2018 15th International Conference on the European Energy Market (EEM)*, pages 1–5. IEEE, 2018.

[27] Israt Fatema, Gang Lei, and Xiaoying Kong. Probabilistic forecasting of electricity demand incorporating mobility data. *Applied Sciences*, 13(11):6520, 2023.

[28] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[29] Pierre Gaillard, Yannig Goude, and Raphaël Nedellec. Additive models and robust aggregation for gefcom2014 probabilistic electric load and electricity price forecasting. *International Journal of forecasting*, 32(3):1038–1050, 2016.

[30] Pierre Gaillarda, Yannig Goudea, and Raphaël Nedelleca. Semi-parametric models and robust aggregation for gefcom2014 probabilistic electric load and electricity price forecasting.

[31] Tilmann Gneiting and Matthias Katzfuss. Probabilistic forecasting. *Annual Review of Statistics and Its Application*, 1:125–151, 2014.

[32] Tilmann Gneiting and Adrian E Raftery. Weather forecasting with ensemble methods. *Science*, 310(5746):248–249, 2005.

[33] Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

[34] Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

[35] Stephen Haben, Siddharth Arora, Georgios Giasemidis, Marcus Voss, and Danica Vukadinović Greetham. Review of low voltage load forecasting: Methods, applications, and recommendations. *Applied Energy*, 304:117798, 2021.

[36] Stephen Haben and Georgios Giasemidis. A hybrid model of kernel density estimation and quantile regression for gefcom2014 probabilistic

load forecasting. *International Journal of Forecasting*, 32(3):1017–1022, 2016.

[37] Stephen Haben, Georgios Giasemidis, Florian Ziel, and Siddharth Arora. Short term load forecasts of low voltage demand and the effects of weather. *arXiv preprint arXiv:1804.02955*, 2018.

[38] Stephen Haben, Marcus Voss, and William Holderbaum. *Core Concepts and Methods in Load Forecasting: With Applications in Distribution Networks*. Springer Nature, 2023.

[39] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

[40] Trevor J Hastie. Generalized additive models. In *Statistical models in S*, pages 249–307. Routledge, 2017.

[41] Yaoyao He, Yun Wang, Shuo Wang, and Xin Yao. A cooperative ensemble method for multistep wind speed probabilistic forecasting. *Chaos, Solitons & Fractals*, 162:112416, 2022.

[42] David Hilbert. Grundzuge einer allgemeinen theorie der linearen integralgleichungen. *Gott. Nachr.*, pages 307–338, 1904.

[43] Thomas Hofmann, Bernhard Schölkopf, and Alexander J Smola. A review of kernel methods in machine learning. *Mac-Planck-Institute Technical Report*, 156, 2006.

[44] CE Holt. Forecasting seasonals and trends by exponentially weighted averages (onr memorandum no. 52). *Carnegie Institute of Technology, Pittsburgh USA*, 10, 1957.

[45] Tao Hong. *Short term electric load forecasting*. North Carolina State University, 2010.

[46] Tao Hong and Shu Fan. Probabilistic electric load forecasting: A tutorial review. *International Journal of Forecasting*, 32(3):914–938, 2016.

[47] Tao Hong, Pierre Pinson, and Shu Fan. Global energy forecasting competition 2012, 2014.

[48] Tao Hong, Pierre Pinson, Shu Fan, Hamidreza Zareipour, Alberto Troccoli, and Rob J Hyndman. Probabilistic energy forecasting: Global energy forecasting competition 2014 and beyond, 2016.

[49] Tao Hong, Jingrui Xie, and Jonathan Black. Global energy forecasting competition 2017: Hierarchical probabilistic load forecasting. *International Journal of Forecasting*, 35(4):1389–1399, 2019.

[50] Christian Huurman, Francesco Ravazzolo, and Chen Zhou. The power of weather. *Computational Statistics & Data Analysis*, 56(11):3793–3807, 2012.

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

[52] Rob J Hyndman, David M Bashtannyk, and Gary K Grunwald. Estimating and visualizing conditional densities. *Journal of Computational and Graphical Statistics*, 5(4):315–336, 1996.

[53] Rob J Hyndman, Jochen Einbeck, and Matthew P Wand. *hdrcde: Highest Density Regions and Conditional Density Estimation*, 2023. R package version 3.4.

[54] Tony Jebara and Risi Kondor. Bhattacharyya expected likelihood kernels. volume 2777, pages 57–71, 01 2003.

[55] Tryggvi Jónsson, Pierre Pinson, Henrik Madsen, and Henrik Aalborg Nielsen. Predictive densities for day-ahead electricity prices using time-adaptive quantile regression. *Energies*, 7(9):5523–5547, 2014.

[56] Motonobu Kanagawa and Kenji Fukumizu. *Recovering Distributions from Gaussian RKHS Embeddings*, volume 33 of *Proceedings of Machine Learning Research*. PMLR, Reykjavik, Iceland, 22–25 Apr 2014.

[57] Devinder Kaur, Shama Naz Islam, Md Apel Mahmud, Md Enamul Haque, and Zhao Yang Dong. Energy forecasting in smart grid systems: recent advancements in probabilistic deep learning. *IET Generation, Transmission & Distribution*, 16(22):4461–4479, 2022.

[58] Yoshihito Kazashi and Fabio Nobile. Density estimation in RKHS with application to korobov spaces in high dimensions. *SIAM Journal on Numerical Analysis*, 61(2):1080–1102, apr 2023.

[59] Andrei Nikolaevitch Kolmogorov. Stationary sequences in hilbert space. *Bull. Math. Univ. Moscow*, 2(6):1–40, 1941.

[60] Alireza Koochali, Peter Schichtel, Andreas Dengel, and Sheraz Ahmed. Random noise vs. state-of-the-art probabilistic forecasting methods: A case study on crps-sum discrimination ability. *Applied Sciences*, 12(10):5104, 2022.

[61] Jesus Lago, Grzegorz Marcjasz, Bart De Schutter, and Rafał Weron. Forecasting day-ahead electricity prices: A review of state-of-the-art algorithms, best practices and an open-access benchmark. *Applied Energy*, 293:116983, 2021.

[62] Katarzyna Maciejowska and Jakub Nowotarski. A hybrid model for gefcom2014 probabilistic electricity price forecasting. *International Journal of Forecasting*, 32(3):1051–1056, 2016.

[63] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[64] Grzegorz Marcjasz, Michał Narajewski, Rafał Weron, and Florian Ziel. Distributional neural networks for electricity price forecasting. *Energy Economics*, 125:106843, September 2023.

[65] James E Matheson and Robert L Winkler. Scoring rules for continuous probability distributions. *Management science*, 22(10):1087–1096, 1976.

[66] Nicolai Meinshausen and Greg Ridgeway. Quantile regression forests. *Journal of machine learning research*, 7(6), 2006.

[67] James Mercer. Xvi. functions of positive and negative type, and their connection the theory of integral equations. *Philosophical transactions of the royal society of London. Series A, containing papers of a mathematical or physical character*, 209(441-458):415–446, 1909.

[68] Jan Kloppenborg Møller, Henrik Aalborg Nielsen, and Henrik Madsen. Time-adaptive quantile regression. *Computational Statistics & Data Analysis*, 52(3):1292–1303, 2008.

[69] Krikamol Muandet, Kenji Fukumizu, Bharath Sriperumbudur, and Bernhard Schölkopf. Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends® in Machine Learning*, 10(1-2):1–141, 2017.

[70] Michael Multerer, Paul Schneider, and Rohan Sen. Fast empirical scenarios, 2023.

[71] Elizbar A Nadaraya. On estimating regression. *Theory of Probability & Its Applications*, 9(1):141–142, 1964.

[72] Jakub Nowotarski and Rafał Weron. Computing electricity spot price prediction intervals using quantile regression and forecast averaging. *Computational Statistics*, 30:791–803, 2015.

[73] Jakub Nowotarski and Rafał Weron. Recent advances in electricity price forecasting: A review of probabilistic forecasting. *Renewable and Sustainable Energy Reviews*, 81:1548–1568, 2018.

[74] Evert J Nyström. Über die praktische auflösung von integralgleichungen mit anwendungen auf randwertaufgaben. 1930.

[75] Fotios Petropoulos, Daniele Apiletti, Vassilios Assimakopoulos, Mohamed Zied Babai, Devon K Barrow, Souhaib Ben Taieb, Christoph Bergmeir, Ricardo J Bessa, Jakub Bijak, John E Boylan, et al. Forecasting: theory and practice. *International Journal of Forecasting*, 38(3):705–871, 2022.

[76] Tomaso Poggio and Federico Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9):1481–1497, 1990.

[77] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[78] Murray Rosenblatt. Conditional probability density and regression estimators. *Multivariate analysis II*, 25:31, 1969.

[79] Erhard Ing Grad Schmidt. Über die auflösung linearer gleichungen mit unendlich vielen unbekannten. *Rendiconti del Circolo Matematico di Palermo (1884-1940)*, 25:53–77, 1908.

[80] Isaac J Schoenberg. Metric spaces and positive definite functions. *Transactions of the American Mathematical Society*, 44(3):522–536, 1938.

[81] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.

[82] David W. Scott. *Multivariate Density Estimation: Theory, Practice, and Visualization*. Wiley Series in Probability and Statistics. Wiley, 2 edition, 2015.

[83] Alex J Smola and Bernhard Schölkopf. *Learning with kernels*, volume 4. Citeseer, 1998.

[84] Alex J Smola and Bernhard Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14:199–222, 2004.

[85] Gábor J Székely and Maria L Rizzo. A new test for multivariate normality. *Journal of Multivariate Analysis*, 93(1):58–80, 2005.

[86] Ichiro Takeuchi, Quoc Le, Timothy Sears, Alexander Smola, et al. Nonparametric quantile estimation. 2006.

[87] Hong Tao. Crystal ball lessons in predictive analytics. *EnergyBiz*, pages 35–37, 2015.

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

[89] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288, 1996.

[90] Dennis W Van der Meer, Joakim Widén, and Joakim Munkhammar. Review on probabilistic forecasting of photovoltaic power production and electricity consumption. *Renewable and Sustainable Energy Reviews*, 81:1484–1512, 2018.

[91] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 1995.

[92] Vladimir Vapnik and Alexey Chervonenkis. Theory of pattern recognition, 1974.

[93] Vladimir N Vapnik. The support vector method. In *International conference on artificial neural networks*, pages 261–271. Springer, 1997.

[94] Grace Wahba. *Spline models for observational data*. SIAM, 1990.

[95] Can Wan, Jin Lin, Jianhui Wang, Yonghua Song, and Zhao Yang Dong. Direct quantile regression for nonparametric probabilistic forecasting of wind power generation. *IEEE Transactions on Power Systems*, 32(4):2767–2778, 2016.

[96] Geoffrey S Watson. Smooth regression analysis. *Sankhyā: The Indian Journal of Statistics, Series A*, pages 359–372, 1964.

[97] Rafal Weron. *Modeling and forecasting electricity loads and prices: A statistical approach*, volume 396. John Wiley & Sons, 2006.

[98] Rafał Weron. Electricity price forecasting: A review of the state-of-the-art with a look into the future. *International Journal of Forecasting*, 30, 10 2014.

[99] Christopher KI Williams and David Barber. Bayesian classification with gaussian processes. *IEEE Transactions on pattern analysis and machine intelligence*, 20(12):1342–1351, 1998.

[100] Peter R Winters. Forecasting sales by exponentially weighted moving averages. *Management science*, 6(3):324–342, 1960.

[101] Jingrui Xie and Tao Hong. Gefcom2014 probabilistic electric load forecasting: An integrated solution with forecast combination and residual simulation. *International Journal of Forecasting*, 32(3):1012–1016, 2016.

[102] Jiawei Zhang, Yi Wang, Mingyang Sun, and Ning Zhang. Two-stage bootstrap sampling for probabilistic load forecasting. *IEEE Transactions on Engineering Management*, 69(3):720–728, 2020.

[103] Lei Zhang, Lun Xie, Qinkai Han, Zhiliang Wang, and Chen Huang. Probability density forecasting of wind speed based on quantile regression and kernel density estimation. *Energies*, 13(22):6125, 2020.

[104] Yao Zhang, Jianxue Wang, and Xifan Wang. Review on probabilistic forecasting of wind power generation. *Renewable and Sustainable Energy Reviews*, 32:255–270, 2014.

[105] Zhendong Zhang, Hui Qin, Yongqi Liu, Liqiang Yao, Xiang Yu, Jiantao Lu, Zhiqiang Jiang, and Zhongkai Feng. Wind speed forecasting based on quantile regression minimal gated memory network and kernel density estimation. *Energy conversion and management*, 196:1395–1409, 2019.

[106] Florian Ziel and Bidong Liu. Lasso estimation for gefcom2014 probabilistic electric load forecasting. *International Journal of Forecasting*, 32(3):1029–1037, 2016.

[107] Florian Ziel and Rick Steinert. Probabilistic mid-and long-term electricity price forecasting. *Renewable and Sustainable Energy Reviews*, 94:251–266, 2018.