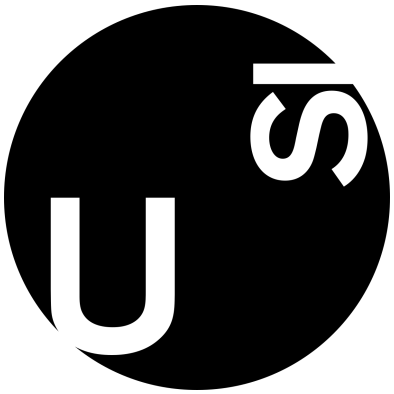


Energy Market Analysis Using Kernel Methods

Master Thesis

L. Pernigo

June 17, 2024

Advisor: Prof. Dr. M. Multerer

Co-Advisor: Dr. D. Baroli

Faculty of Informatics, USI Lugano

Abstract

This thesis is concerned with electricity forecasting. Specifically, we concentrated on the probabilistic framework. This choice was motivated by its importance in systems planning and operations as a consequence of the inception of competitive power markets, smart grids and renewable integration requirements. In doing so, we focused on the family of kernel methods. We have compared them against several other statistical and machine learning techniques. Our results showed the feasibility of kernel methods in the field of electricity forecasting both in terms of point and probabilistic forecasting. In particular, in the probabilistic context, our experiments show the validity of kernel quantile regression equipped with the Laplacian kernel. These findings indicate that kernel methods are well suited to the characteristics of electricity. Anyone interested in forecasting energy should consider them when faced with the choice of the model. They can be employed stand-alone or combined with other valid methods into ensembles.

Acknowledgements

First and foremost I would like to express my deepest gratitude to my advisor and co-advisor, Prof. Multerer and Dr. Baroli for their invaluable feedback, continuous support and patience during my Master's Thesis. Furthermore, I thank them for having given me the opportunity to collaborate through this thesis work.
I could not have undertaken this journey without my Master directors Prof. Schenk and Prof. Wit. I am extremely grateful for their guidance, mentorship and advice throughout my Master studies.
I would like to extend my deepest appreciation to all professors I had during my studies at Usi. Their stimulating questions and push to enquire motives behind theory challenged my curiosity, shaped my interests and fostered my independent thinking.
I also had great pleasure of discussing with the research group of Prof. Multerer the challenges I faced during this thesis work. I thank them for their affability and insightful suggestions.
Special thanks to my cohort members and to the friends I made along the way. I will never forget the good times we had together.

Contents

List of Figures

List of Tables

Chapter 1

Problem description

Individuals and organizations constantly face situations of uncertainty, thus the need for robust forecasting methods. Such methods are crucial in the process of taking informed decisions 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 [14]. Moreover, the process of deregulation lead to an increasing interest in the field of electricity forecasting (EF) within the academic community, see figure 2.2. In addtion, the United Nations have identified the right to access affordable, reliable, sustainable and modern energy as one of their 17 sustainable development goals (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 [94]. Furthermore, the adoption of smart meters provides power market

companies with a huge amount of consumer data. This enables 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 network. 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 enable 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 [106], is all about predicting a particular value in time. On the other hand, with probabilistic forecasting we aim at predicting either an interval, quantiles or a probability distribution for each point in time [79]. 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 expectation. 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 extract-load-transform (ETL) pipeline and the exploratory data analysis (EDA). Implementation details are included

in Section 9. Finally, Section 10 presents experiments 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 [106]. As a solution, [67] 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 [49] as

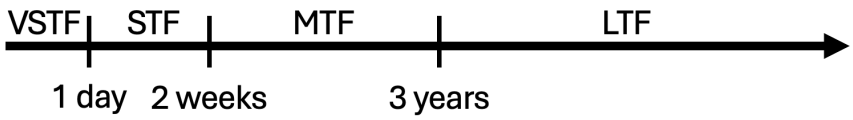


Figure 2.1: Time classsification [50]

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

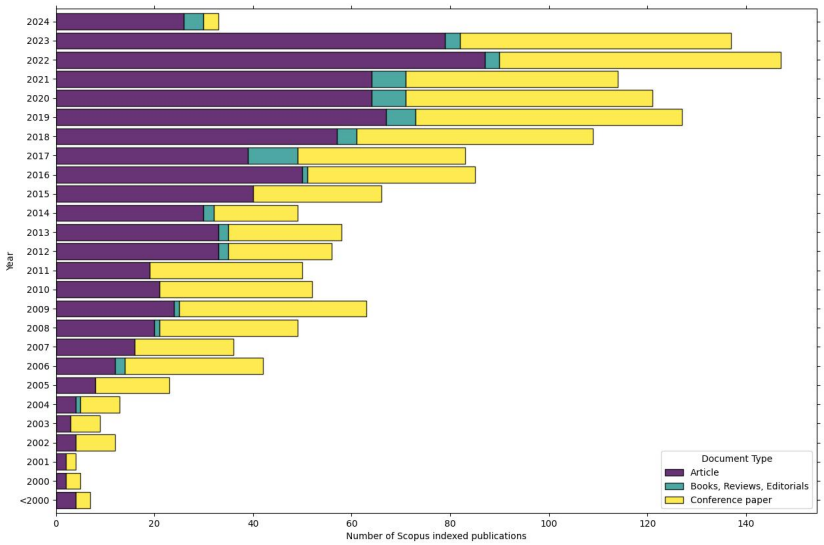


Figure 2.2: Electricity forecasting publications over the past years

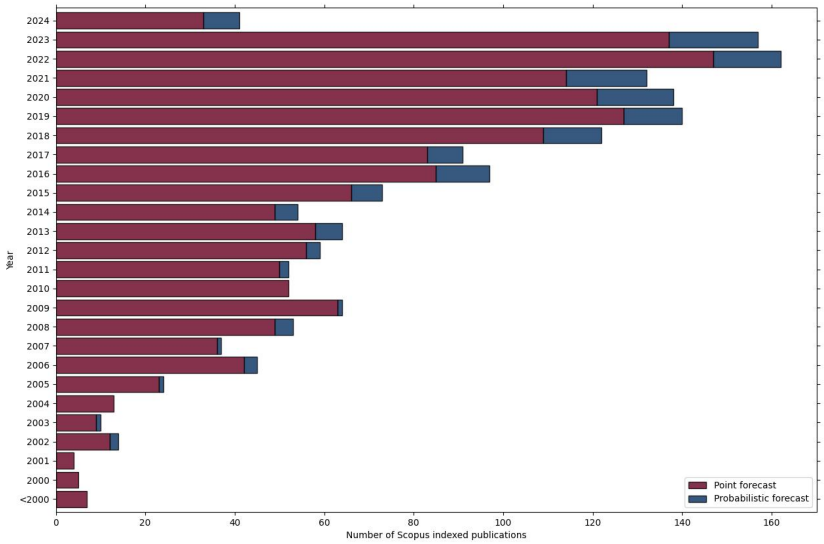


Figure 2.3: Point versus probabilistic pubblications over the past years

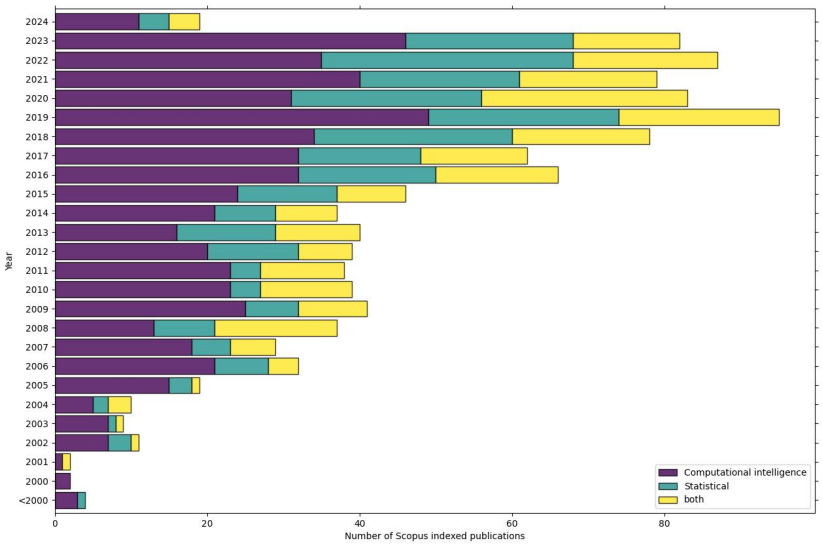


Figure 2.4: Pubblications by method over the past years

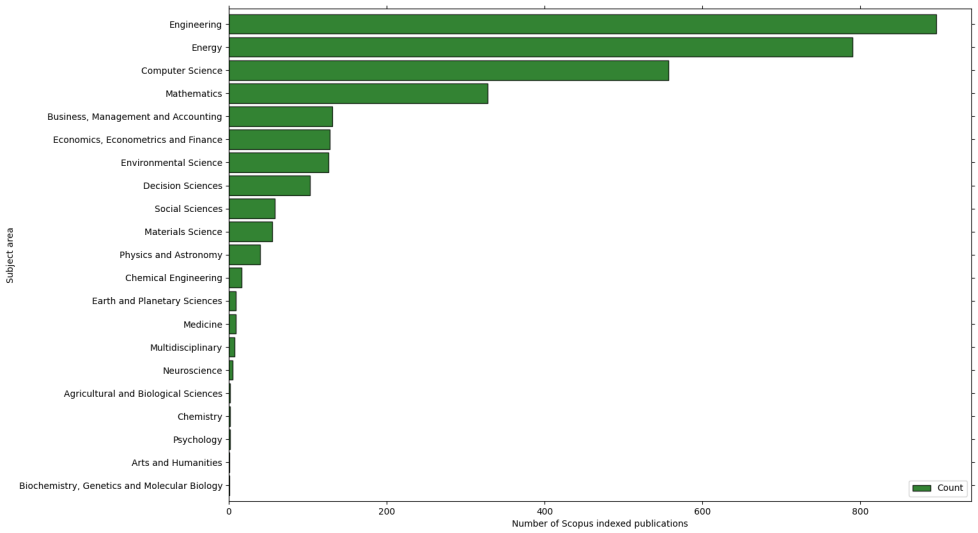


Figure 2.5: Pubblications by subject area

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.

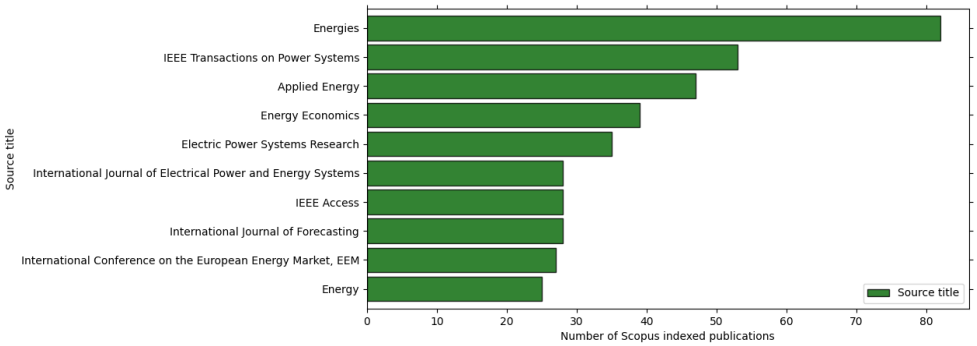


Figure 2.6: Most popolar sources/outlets

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 [106] 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. [50] 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. [79] carried out a review of probabilistic forecasting. Weron et al. [67] 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. [112] considers state of the art methods in wind power probabilistic forecasting and describes current challenges and possible future developments. Ziel et al. [115] provides detailed tables, groups research papers by the time dimension and objective and reports dataset, model and accuracy measures adopted. David et al. [24] 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 [26] reviews 25 years of time series forecasting for the period 1990-2005, highlighting the most influential works. He et al. [45] 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. [103] proposes a combination of QR and extreme learning machine (ELM) to generate non parametric probabilistic forecasts of wind generation. Zhang et al. [113] forecasts wind speed by adopting QRMGM, which is a combination of QR with a minimal gated memory network. Hyndman et al. [56] explains kernel estimator of conditional density, analyses its asymptotic behaviour and covers an application for the daily temperatures of Melbourne. Kaur et al. [63] carries out a comparative study of techniques spanning ideas from statistic and artificial intelligence. Van der Meer et al. [97] 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. [70] uses distributional neural networks to create probabilistic forecasts for the day ahead electricity prices in the German market. Nowotarski et al. [78] 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. [12] focus on modelling electricity smart meter data by proposing a non parametric probabilistic technique based on kernel density estimation and conditional kernel density estimation [84, 56]. 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. [111] 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. [40] 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. [66] reviews various existing methods for assessing probabilistic forecast models and discusses their advantages and disadvantages. Matheson et al. [71] develops classes of scoring rules for continuous probability distributions. Gneiting et al. [36] provides a thorough overview of the theory of scoring rules for interval and density forecasts. Gneiting et al. [34] covers theory and state of the art techniques in probabilistic forecasting. Zhang et al [110] 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. [61] introduces a density model for the

day ahead market extending the adaptive QR framework of [74] by modelling the tails of the predicted density with an exponential distribution. Fatema et al. [29] 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 [28] proposes a probabilistic forecasting model based on the Nadaraya Watson estimator [77, 104]. Huurman et al. [54] 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 [81]. In the literature, combined forecasts are called ensemble [35]. 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 [51] for a comprehensive review.
The focus of GEFCom 2014 was on probabilistic forecasting, Hong et al. [52] 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. [109] 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. [68] proposes a new probabilistic model extending the idea of quantile regression averaging (QRA) [78]. Haben et al. [39] mixes conditional kernel density (CKD) and quantile regression in their competition entry. Gaillard et al. [32, 33] combines quantile regression with generalized additive models [43]. Ziel et al. [114] estimates an AR model through the LASSO [96] instead of the standard OLS.
The last GEFCom was held in 2017, its focus was providing probabilistic

load forecasts, see [53]. 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 [46] where he introduced the term "definite kernels". Following, Hilbert's and Schmidt's work [85] lead to the introduction of a new space, the Hilbert space. In 1909, James Mercer improved Hilbert's work by proposing his theorem [73]. This theorem underlies the power of kernel methods, that is the kernel trick. In 1938, Schoenberg [86] developed the mathematical results that allow us to find the kernel associated to a specific feature space metric. In 1941, Kolmogorov [65] carried out stuides on representing kernel in linear spaces. In 1950, Aronszajn [11] 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 [100] [99], Gaussian process classifiers [107], spline methods [102], neural networks [82] and principal component analysis [87]. 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 [90, 47] and [89].
Kanagawa et Fukumizu introduces to the concept of kernel mean embedding [62]. Muandet et al. [75] 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. [21] proposes an efficient approximation procedure based on the Nyström method [80], providing also an upper bound for the approximation error. Smola et al. [22] 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 covers the theory of kernel methods and introduces the building blocks underlying it.

Many algorithms use the inner product as similarity measure between the data instances $x, y \in \mathcal{X}$. 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.

$$\begin{aligned} \varphi(x) \colon \mathcal{X} &\to \mathcal{H} \\ x &\to \varphi(x) \end{aligned}$$

We take then the inner product in the high dimensional space $\mathcal{H}$ mapped by $\varphi(x)$, i.e.

$$k(x, y) := \langle \varphi(x), \varphi(y) \rangle_{\mathcal{H}}$$

where $\varphi(x)$ is referred to as feature map while k is the kernel function.
Therefore, we can kernelize any algorithm involving a dot product by substituting $\langle x, y \rangle_{\mathcal{X}}$ with $\langle \varphi(x), \varphi(y) \rangle_{\mathcal{H}}$
One would expect constructing the feature maps explicitly and then evaluating their inner product in $\mathcal{H}$ to be computationally expensive, and indeed it is. However, we do not have to explicitly perform such calculations. This is because of the kernel trick.
To illustrate the idea behind the kernel trick consider the following example. Suppose $x, y \in \mathbb{R}^2$ and assume $\varphi(x) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)$, then the inner product in the feature space is $x_1^2y_1^2, x_2^2y_2^2 + 2x_1x_2y_1y_2$. Notice that this is the same of $\langle \varphi(x), \varphi(y) \rangle_{\mathcal{H}}$; thus the kernel trick consists of just using $k(x, y) := (x^\intercal y)^2$.
For instance:

$$\langle \varphi([2,4]), \varphi([8,9]) \rangle_{\mathcal{H}} = \begin{bmatrix} 4 & 16 & \sqrt{2} \cdot 2 \cdot 4 \end{bmatrix} \begin{bmatrix} 64 \\ 81 \\ \sqrt{2} \cdot 8 \cdot 9 \end{bmatrix}$$
$$= 4 \cdot 64 + 16 \cdot 81 + 2 \cdot 2 \cdot 4 \cdot 8 \cdot 9 = 2704$$

While $k([2,4],[8,9]) = (16+36)^2 = 2704$ We can see that the latter calculation is much quicker and compact. This gives a taste of how powerful kernels are thanks to the kernel trick. Indeed, the idea of the the kernel trick can be extended to feature maps φ involving an infinite feature space. In these cases, calculating the respective kernel is equal to calculating the inner product between an infinite number of features of the data points.

3.1 Theory of kernels

Subsequently we introduce 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 $N = N(\varepsilon) \in \mathbb{N}$ such that $\|v_n - v_m\|_{\mathcal{V}} < \varepsilon \ \ \forall m, n \geq N$*

Definition 3.2 *A sequence $\{v_n\}_{n=1}^{\infty}$ is convergent if for every $\varepsilon > 0$ there exists $N = N(\varepsilon) \in \mathbb{N}$ and a point $v \in \mathcal{V}$ such that $\|v_n - v\|_{\mathcal{V}} < \varepsilon \ \forall n \geq N$.*

Definition 3.3 *A complete metric space is a metric space in which every Cauchy sequence is convergent.*

Definition 3.4 *A Hilbert space is a vector space $\mathcal{H}$ with an inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ such that the norm defined by $\|f\|_{\mathcal{H}} = \sqrt{\langle f, f \rangle_{\mathcal{H}}}$ turns $\mathcal{H}$ into a complete metric space.*

Definition 3.5 *A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is said to be finitely positive semidefinite if it is a symmetric function for which the matrices formed with it are positive semidefinite.*

Theorem 3.6 (Characterisation of kernels) *A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ which is either continous or has finite domain can be decomposed as $k(x,y) = \langle \varphi(x), \varphi(y) \rangle_{\mathcal{H}}$ if and only if k satisfy the positive semidefinite property.*

That is the function k can be written as the inner product in the Hilbert space $\mathcal{H}$ of the features evaluated at the arguments $\varphi(x)$.

Next, we make this characterisation explicit by showing how to construct a feature mapping φ of a kernel k satisfying the finitely positive semidefinite property. Let the feature space be the set of functions
$\mathcal{F} = \{ \sum_{i=1}^{n} \alpha_i k(x_i, \cdot) : n \in \mathbb{N}, x_i, y_i \in \mathcal{X}, a_i \in \mathbb{R}, i = 1, \cdots, n \}$. This space is

clearly closed under addition of functions and scalar multiplication, therefore it is a vector space. Letting $f(x) = \sum_{i=1}^{n} \alpha_i k_i(x_i, x)$ and $g(x) = \sum_{i=1}^{m} \beta_i k_i(y_i, x)$, we can introduce an inner product on $\mathcal{F}$ as
$\langle f, g \rangle_{\mathcal{F}} =: \sum_{i=1}^{n} \sum_{j=1}^{m} a_i \beta_j k_i(x_i, y_j) = \sum_{i=1}^{n} a_i g(x_i) = \sum_{j=1}^{m} \beta_j f(y_j)$
Notice, $\langle f, f \rangle_{\mathcal{F}} = \alpha^\intercal K \alpha \geq 0 \; \forall f \in \mathcal{F}$, where K is the kernel matrix. Moreover, $\langle f, g \rangle_{\mathcal{F}}$ is real valued, symmetric and bilinear. All these facts implies that the just introduced inner product is a valid metric.
Additionally, letting $g(x) = k(\cdot, x)$, we have that

$$\langle f, k(\cdot, x) \rangle_{\mathcal{F}} = \sum_{i=1}^{n} a_i k(x_i, x) = f(x) \tag{3.1}$$

This is known as the reproducing property.
Finally, to show the characterisation of kernels we have to check that the inner product makes $\mathcal{F}$ a complete metric space. Consider a fixed input x and a Cauchy sequence $\{f_n\}_{n=1}^{\infty}$, we then have by means of the Cauchy-Schwarz inequality that

$$(f_n(x) - f_m(x))^2 = \langle f_n - f_m, k(\cdot, x) \rangle_{\mathcal{F}}^2 \leq \|f_n - f_m\|_{\mathcal{F}}^2 k(x, x) \tag{3.2}$$

We have then, that $f_n(x)$ is a Cauchy sequence of real numbers. Thus, we can let $g(x) = \lim_{n \to \infty} f_n(x)$ and include all such limits functions in $\mathcal{F}$, doing so we complete $\mathcal{F}$ and we obtain the Hilbert space $\mathcal{H}$ associated to the kernel k. Having constructed the feature space $\mathcal{H}$, the mapping $\varphi(x)$ is defined as follows

$$\varphi : x \in \mathcal{X} \to \varphi(x) = k(\cdot, x) \in \mathcal{H} \tag{3.3}$$

Notice, that by the reproducing property, f can be represented as the inner product with itself in the feature space, that is

$$f(x) = \langle f, \varphi(x) \rangle_{\mathcal{H}} = \langle f, k(\cdot, x) \rangle_{\mathcal{H}} = f(x) \tag{3.4}$$

Definition 3.7 *Let $(\mathcal{H}, \langle \cdot, \cdot \rangle_{\mathcal{H}})$ be a Hilbert space of real-valued functions on $\mathcal{X}$. A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is called a reproducing kernel of $\mathcal{H}$ if and only if*

$$k(x, \cdot) \in \mathcal{H} \quad \forall x \in \mathcal{X} \tag{3.5}$$
$$\langle f, k(x, \cdot) \rangle_{\mathcal{H}} = f(x) \quad \forall f \in \mathcal{H}, x \in \mathcal{X} \tag{3.6}$$

Notice, the finitely positive semidefinite property of kernel k is a sufficient condition for $\mathcal{H}$ to have an associated RKHS.

The analysis of above applies to any kernel, that is given a valid kernel function we can always construct its RKHS in the way we just illustrated.

Theorem 3.8 *If a symmetric function $k(\cdot,\cdot)$ satisfies the reproducing property in a Hilbert space $\mathcal{H}$, then k satisfies the finitely positive semidefinite property.*

Proof

$$\sum_{i,j=1}^{n} a_i a_j k(x_i, x_j) = \sum_{i,j=1}^{n} a_i a_j \langle k(\cdot, x_i), k(\cdot, x_j)\rangle_{\mathcal{H}} \tag{3.7}$$

$$= \left\langle \sum_{i=1}^{n} a_i k(\cdot, x_i), \sum_{j=1}^{n} a_j k(\cdot, x_j) \right\rangle_{\mathcal{H}} \tag{3.8}$$

$$= \left\| \sum_{i=1}^{n} a_i k(x_i, x_i) \right\|_{\mathcal{H}}^2 \geq 0 \tag{3.9}$$

□

Theorem 3.9 (Riesz Representation) . *If $L : \mathcal{H} \to \mathbb{R}$ is a bounded linear operator on a Hilbert space $\mathcal{H}$, there exists some $h_0 \in \mathcal{H}$ such that $L(f) = \langle f, h_0\rangle_{\mathcal{H}}$, $\forall f \in \mathcal{H}$.*

Proof It is known that the null space of L, call it M, is a closed linear subspace in $\mathcal{H}$. This implies that the Hilbert space direct sum $M \bigoplus M^{\perp}$ is an isomorphism of $\mathcal{H}$. When $L(f) \neq 0$ we have that $L(f) = L(m + w)$ with $m \in M$ and $w \in M^{\perp}$, it follows that $L(f) = 0 + L(w) \neq 0 \implies M^{\perp} \neq 0$.
Using this fact, we can now choose a $z \in M^{\perp}$ with norm 1. By linearity of L, for any $f \in \mathcal{H}$ we have $L(zL(f) - fL(z)) = L(z)L(f) - L(f)L(z) = 0$. This means that $zL(f) - fL(z)$ belongs to the null space of L, that is $zL(f) - fL(z) \in M$.
Consider now

$$\begin{aligned} L(f) =& L(f) \cdot 1 && (3.10)\\ =& L(f)\langle z, z\rangle_{\mathcal{H}} && (3.11)\\ =& \langle zL(f), z\rangle_{\mathcal{H}} && (3.12)\\ =& \langle zL(f) - fL(z) + fL(z), z\rangle_{\mathcal{H}} && (3.13)\\ =& \langle zL(f) - fL(z), z\rangle_{\mathcal{H}} + \langle fL(z), z\rangle_{\mathcal{H}} && (3.14)\\ =& \langle fL(z), z\rangle_{\mathcal{H}} && (3.15)\\ =& \langle f, z\overline{L(f)}\rangle_{\mathcal{H}} && (3.16)\\ & && (3.17) \end{aligned}$$

Therefore, $L(f) = \langle f, h_0\rangle$ with $h_0 = z\overline{L(z)}$
Finally, notice that h_0 is unique. Assuming h_0 and h_1 to both satisfy the above

equation for all $f \in \mathcal{H}$ we have that

$$\langle h_0, f\rangle_{\mathcal{H}} = \langle h_1, f\rangle_{\mathcal{H}} \tag{3.18}$$
$$\langle h_0, f\rangle_{\mathcal{H}} - \langle h_1, f\rangle_{\mathcal{H}} = 0 \tag{3.19}$$
$$\langle h_0 - h_1, f\rangle_{\mathcal{H}} = 0 \tag{3.20}$$
□

Setting $f = h_0 - h_1$, we have the uniqueness of h_0.

The Riesz representation theorem results in the following proposition for reproducing kernel hilbert spaces.

Proposition 3.10 *For each $x \in \mathcal{X}$ there exists a function $k(\cdot, x) \in \mathcal{H}$ such that the evaluation functional $F_x(f) = \langle k(\cdot, x), f\rangle_{\mathcal{H}} = f(x)$*

The function $k(\cdot, x)$ is the reproducing kernel evaluated at point x. Furthermore, note that k_x is itself a function lying in it.

$$k(y, x) = F_y(k(\cdot, x)) = \langle k(\cdot, x), k(\cdot, y)\rangle_{\mathcal{H}} = \langle \varphi(x), \varphi(y)\rangle_{\mathcal{H}}$$

Definition 3.11 *Considering a set of vectors $x_1, \cdots, x_n$, the Gram matrix is defined as the $n \times n$ matrix whose entries are $\langle x_i, x_j\rangle_{\mathcal{X}}$.*

Notice, it follows from the definition that Gram matrix is symmetric. In the kernel setting we evaluate the inner products in the feature space generated by the feature map φ, therefore the Gram matrix will have entries $\langle \varphi(x_i), \varphi(x_j)\rangle_{\mathcal{H}} = k(x_i, x_j)$. Such matrix is also referred as the kernel matrix K.

$$K_{n\times n} = \begin{bmatrix} k(x_1, x_1) & \dots & k(x_1, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \dots & k(x_n, x_n) \end{bmatrix}$$

All kernel methods take as input the kernel matrix, it is the central data type for all such techniques.

Proposition 3.12 *In addition to being symmetric, K possesses also the property of being positive semidefinite.*

Proof

$$z^\intercal Kz = \sum_{i,j=1}^{n} z_i z_j k(x_i, x_j) \tag{3.21}$$

$$= \sum_{i,j=1}^{n} z_i z_j \langle \varphi(x_i), \varphi(x_j) \rangle_{\mathcal{H}} \tag{3.22}$$

$$= \left\langle \sum_{i=1}^{n} z_i \varphi(x_i), \sum_{j=1}^{n} z_j \varphi(x_j) \right\rangle_{\mathcal{H}} \tag{3.23}$$

$$= \left\| \sum_{i=1}^{n} z_i \varphi(x_i) \right\|_{\mathcal{H}}^2 \geq 0 \tag{3.24}$$

□

We know that positive semidefinite matrices form a cone in the vector space of $n \times n$ matrices. Because of this we have that such matrices are convex, this implies that all the nice theory of convex analysis can be used when optimising over them.

Kernel functions are characterized by a couple of properties that make them a closure. Thus, there exists ways of combining known kernels in order to obtain new kernels.

Proposition 3.13 *Let k_1 and k_2 be kernels over $\mathcal{X} \times \mathcal{X}, \mathcal{X} \subset \mathbb{R}^n$, $a \in \mathbb{R}^+$ and $\varphi : \mathcal{X} \to \mathbb{R}^N$ with k_3 a kernel over $\mathbb{R}^N \times \mathbb{R}^N$. Then the following functions are valid kernels:*

$$k(x, y) = k_1(x, y) + k_2(x, y) \tag{3.25}$$

$$k(x, y) = a k_1(x, y) \tag{3.26}$$

$$k(x, y) = k_1(x, y) k_2(x, y) \tag{3.27}$$

$$k(x, y) = k_3(\varphi(x), \varphi(y)) \tag{3.28}$$

.

Proof Consider a finite set of points $x_1, \cdots, x_n$, a vector $z \in \mathbb{R}^n$ and let K_1 and K_2 be the kernel matrices associated to k_1 and k_2 evaluted on these points.
We have:

$$z^\intercal (K_1 + K_2) z = z^\intercal K_1 z + z^\intercal K_2 z \geq 0 \tag{3.29}$$

$$z^\intercal a K z = a z^\intercal K z \geq 0 \tag{3.30}$$

Let $K = K_1 \otimes K_2$, we have that the eigenvalues of K are made up by the product pairs of the eigenvalues of K_1 and K_2. Thus, K is positive semidefinite. Next, the matrix given by $k_1 \cdot k_2$ corresponds to the Schur product of K_1 and

Table 3.1: Kernel types

Kernel function	Equation	Hyperparameters
Linear	$k(x,y) = xy$	
Polynomial	$k(x,y) = (x^\intercal y + c)^d$	c, d
Gaussian RBF	$k(x,y) = \exp\left(-\frac{\|x-y\|_2^2}{2\sigma^2}\right)$	σ
Exponential RBF/Laplacian	$k(x,y) = \exp\left(-\frac{\|x-y\|_1}{\gamma}\right)$	γ
Hyperbolic/Sigmoid Kernel	$k(x,y2) = \tanh(\gamma x^\intercal y + r)$	γ, r
Periodic	$k(x,y) = \exp\left(\frac{-2sin^2\left(\frac{\pi}{p}\|x-y\|_2\right)}{l^2}\right)$	p, l
Chi-squared kernel	$k(x,y) = \exp\left(-\gamma \sum_i \frac{(x_i-y_i)^2}{x_i+y_i}\right)$	γ
Cosine	$k(x,y) = \frac{xy^\intercal}{\|x\|_2\|y\|_2}$	
Matern	$k(x,y) = \frac{1}{\Gamma(\nu)2^{\nu-1}}\left(\frac{\sqrt{2\nu}}{l}\|x-y\|\|_2\right)^\nu Y_\nu\left(\frac{\sqrt{2\nu}}{l}\|x-y\|\|_2\right)$	l, ν

K_2, denote it H. Notice, H is a principal submatrix of K, hence for any $z \in \mathbb{R}^n$ there exist a $t \in \mathbb{R}^{n^2}$ such that $z^\intercal Hz = t^\intercal Kt$. Using the fact that K is positive semidefinite we have that for any z, it holds

$$z^\intercal Hz = t^\intercal Kt \geq 0 \tag{3.31}$$

□

3.2 Kernel families

The selection of kernel determines the class of functions that will be searcehd by the learning algorithm. Prior knowledge of the problem domain can help restricting the candidate families of kernel functions. Table 3.1 contains popular kernel families in literature and applications.

Notice, in the Matern family $\Gamma(\cdot)$ is the gamma function while $Y(\cdot)$ is the Bessel function of the second kind.
When $\nu = p + \frac{1}{2}$, $p \in \mathbb{N}^+$, the Matern kernel can be rewritten as a product of an exponential and a polynomial of degree p.

$$k(x,y) = \exp\left(-\frac{\sqrt{2p+1}\|x-y\|\|_2}{l}\right)\frac{p!}{(2p)!}\sum_{i=0}^{p}\frac{(p+i)!}{i!(p-i)!}\left(\frac{2\sqrt{2p+1}\|x-y\|\|_2}{l}\right)^{p-i}$$

For example:

- $p = 0, \nu = \frac{1}{2} \implies k(x,y) = \exp\left(\frac{-\|x-y\|\|_2}{l}\right)$
- $p = 1, \nu = \frac{3}{2} \implies k(x,y) = \left(1 + \frac{\sqrt{3}\|x-y\|\|_2}{l}\right)\exp\left(-\frac{\sqrt{3}\|x-y\|\|_2}{l}\right)$
- $p = 2, \nu = \frac{5}{2} \implies k(x,y) = \left(1 + \frac{\sqrt{5}\|x-y\|\|_2}{l} + \frac{5\|x-y\|\|_2^2}{3l^2}\right)\exp\left(-\frac{\sqrt{5}\|x-y\|\|_2}{l}\right)$

Furthermore, as $\nu \to \infty$ we have that the Matern kernel converges to the Gaussian RBF kernel. $k(x,y) = \exp\left(\frac{-\|x-y\|\|_2^2}{2l^2}\right)$

Any algorithm involving inner products between inputs can be combined with a kernel method. The chosen kernel function will calculate the inner product between the images of two inputs in the feature space. This makes the orginal algorithm operate in a high dimesional space. Kernel methods show good modularity because they work the same for any kernel and any data type. This concept is illustrated in figure 3.1, where an abstract kernel methods workflow is visualized. First we build the kernel matrix by using a kernel function to process the input data. Next, the kernel matrix is passed to the pattern analysis algorithm, which returns the learned function.

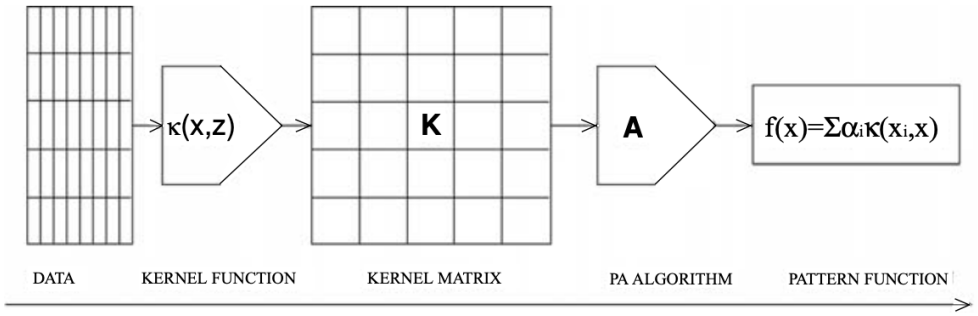


Figure 3.1: Workflow for the application of kernel methods [89]

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 electricity forecasting. Error metrics and measures vary depending on whethter 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 $y = (y_{n+1}, y_{n+2}, \ldots, y_{n+h})$ and its h step ahead point forecast $\hat{y} = (\hat{y}_{n+1}, \hat{y}_{n+2}, \ldots, \hat{y}_{n+h})$ the mean absolute error (MAE) is defined as

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

4.2 Root mean squared error

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

Mean absolute error and root mean squared error 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 which 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 $\text{MAPE}(y, \hat{y}) = \frac{100}{h} \sum_{k=1}^{h} \frac{|y_{n+k} - \hat{y}_{n+k}|}{|y_{n+k}|}$

4.4 Root mean squared percentage error

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

Note, mean absolute percentage error and root mean squared percentage error 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 $\text{MASE}(y, \hat{y}) = \frac{1}{h} \sum_{k=1}^{N} \frac{|y_{n+k} - \hat{y}_{n+k}|}{\frac{1}{h-1} \sum_{k=2}^{h} |y_k - y_{k-1}|}$

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 $\hat{y}_{n+1}^{\text{naive}} = y_n$.

4.6 Root mean squared scaled error

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

4.7 Pinball

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

Definition 4.7 $\text{Pinball}(y_t, \hat{y}_{t,q}, q) = \begin{cases} (q-1)(\hat{y}_{t,q} - y_t) & y_t > \hat{y}_{t,q} \\ q(\hat{y}_{t,q} - y_t) & y_t \leq \hat{y}_{t,q} \end{cases}$

The pinball loss is an asymmetric function, it weights its score differently depending on the error sign and on the quantile considered, see figure 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.

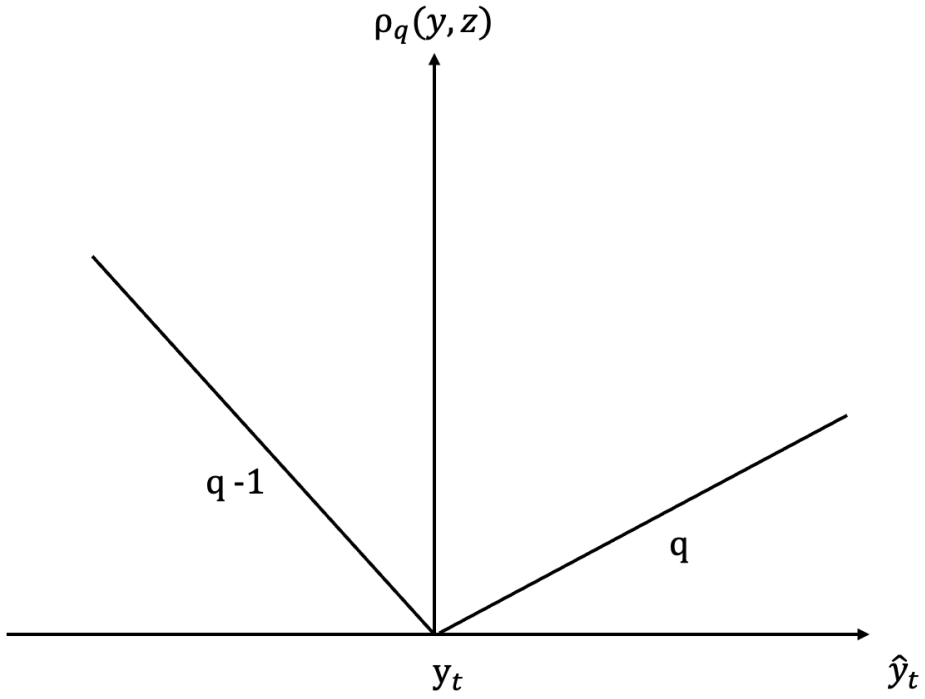


Figure 4.1: Pinball loss

4.8 Winkler

Definition 4.8 $\text{Winkler}(y_t, \hat{y}_t, q) = \begin{cases} \delta & l_t \leq y_t \leq u_t \\ \delta + 2(l_t - y_t)/\alpha & y_t < l_t \\ \delta + 2(l_t - u_t)/\alpha & y_t \geq u_t \end{cases}$

Where delta is the prediction interval (PI) width, that is $\delta = u_t - l_t$ where u_t is the prediction interval upper threshold and l_t is the prediction interval lower threshold at time step t. This score penalizes observations falling outside the prediction interval and rewards narrow prediction intervals.

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 cumulative density function (CDF).

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

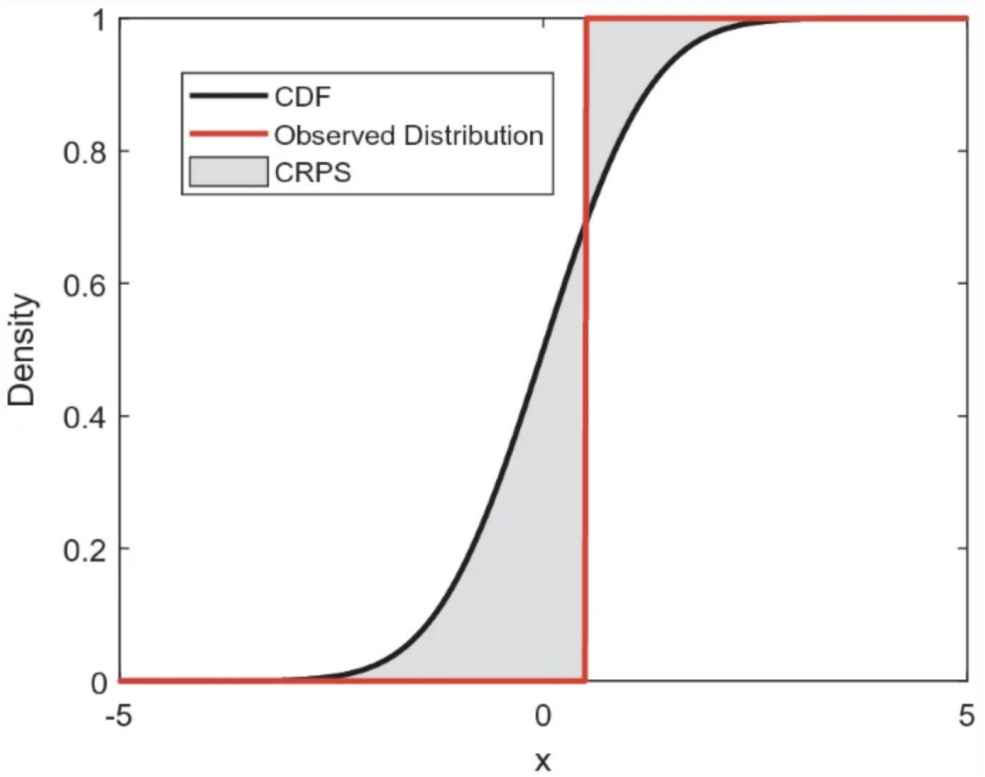


Figure 4.2: CRPS integral [41]

Where the indicator function is defined as $\mathbb{I}_{\{z\}} = \begin{cases} 0 & z < 0 \\ 1 & z \geq 0 \end{cases}$
For a visualisation see, 4.2. The grey area is what contributes toward the CRPS score. The better the estimated cumulative density function 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 [37], where authors take advantage of lemma 2.2 of [13] or equivalently identity 17 of [92].

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

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

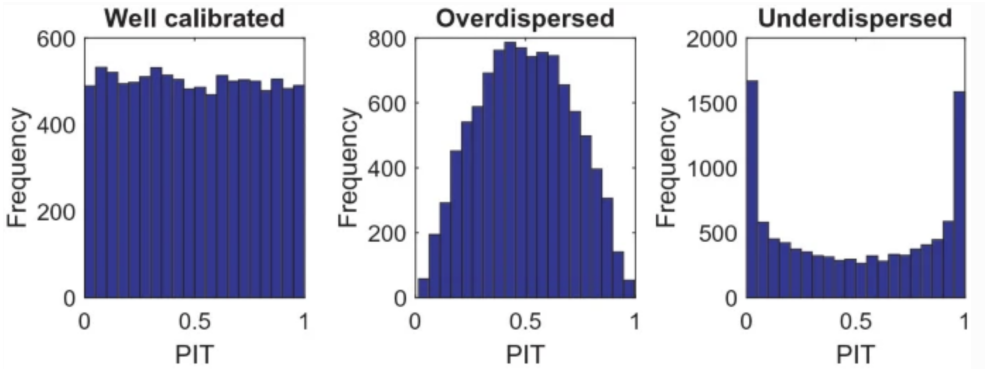


Figure 4.3: Probability integral transform types [41]

Notice that, in our case, equation 4.9, the distribution G of Y_1 and Y_2 is degenerate, with all probability mass on a single point y. 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 $Y_1 = y$.
Putting everything together we have obtained an alternative way of computing the CRPS score.

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

4.10 Probability integral transform

The probability integral transform (PIT) is a method to assess visually the quality of a probabilistic forecast. PIT is obtained by applying the predicted cumulative density function $\hat{F}$ to your data; if applying such CDF to the data results in a uniform distributed PIT, then $\hat{F}$ is a valid prediction. If not, $\hat{F}$ 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.

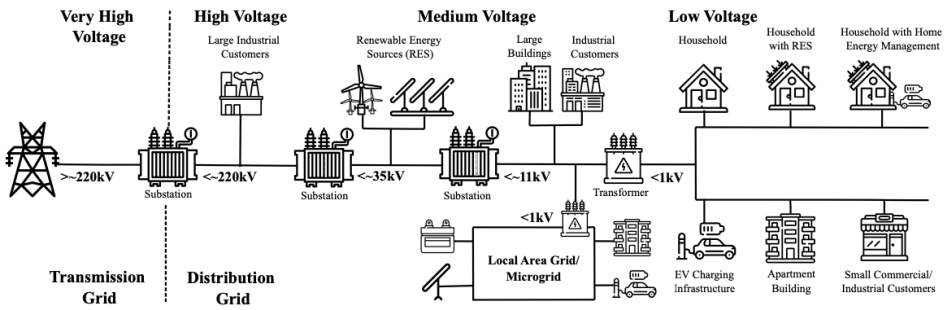


Figure 5.1: Electricity grid [38]

5.2 Electricity markets

Electricity markets have undergone 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 differentiate 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 for 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.

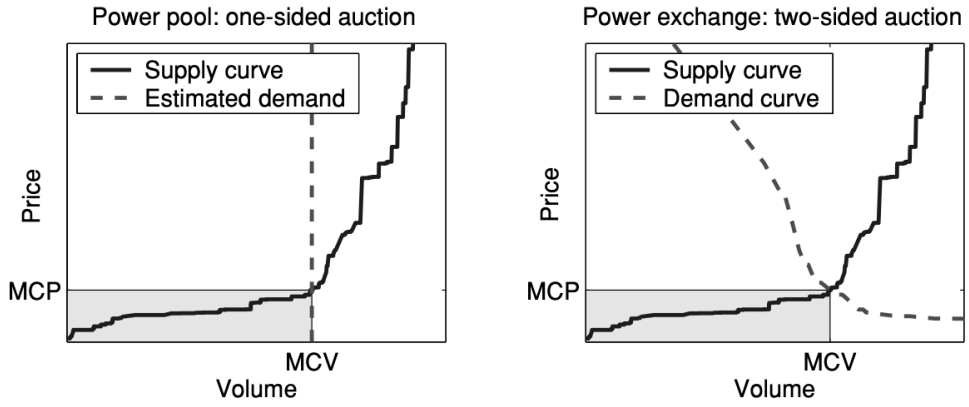


Figure 5.2: Pool market versus power echange [105]

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.

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 and 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.

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.

5.2.3 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.

5.2.4 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 then restarting their power plant or selling their energy for a negative price; that is they pay consumers for consuming their energy.

5.2.5 Nodal and zonal pricing

Grid configuration and physical limits on electricity lines may lead to congestion. Zonal market clearing pricing (ZMCP) and local marginal price (LMP) 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 zonal market clearing pricing, prices may differ between zones but are the same within the same area. On the other hand, local marginal price 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 see [25, 10].

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, 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,

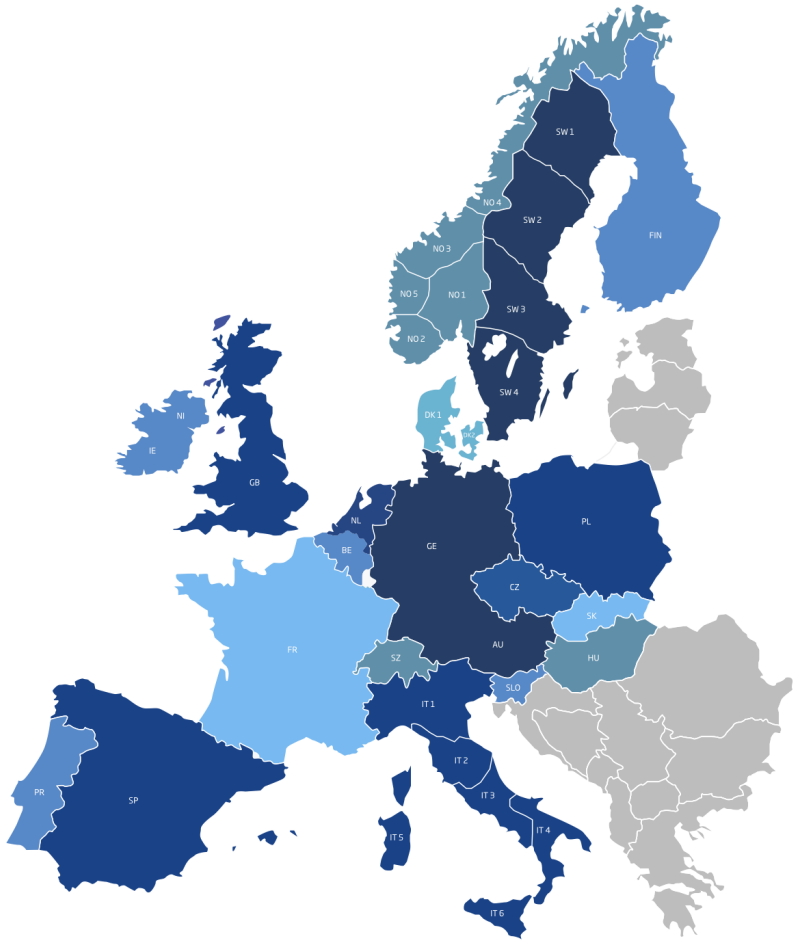


Figure 5.3: Zonal pricing EU [1]

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.

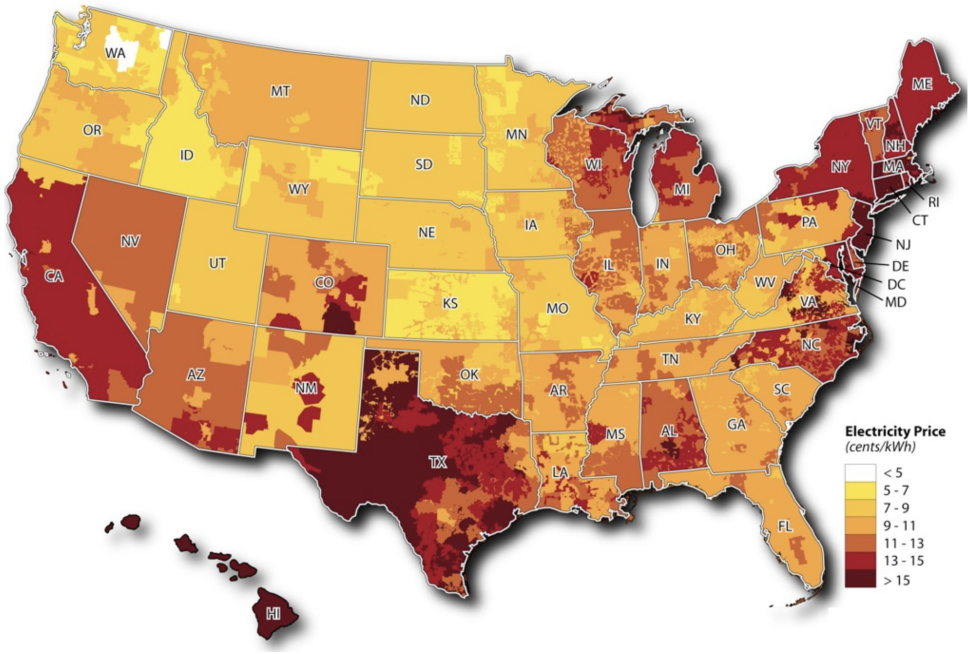


Figure 5.4: Local pricing US [4]

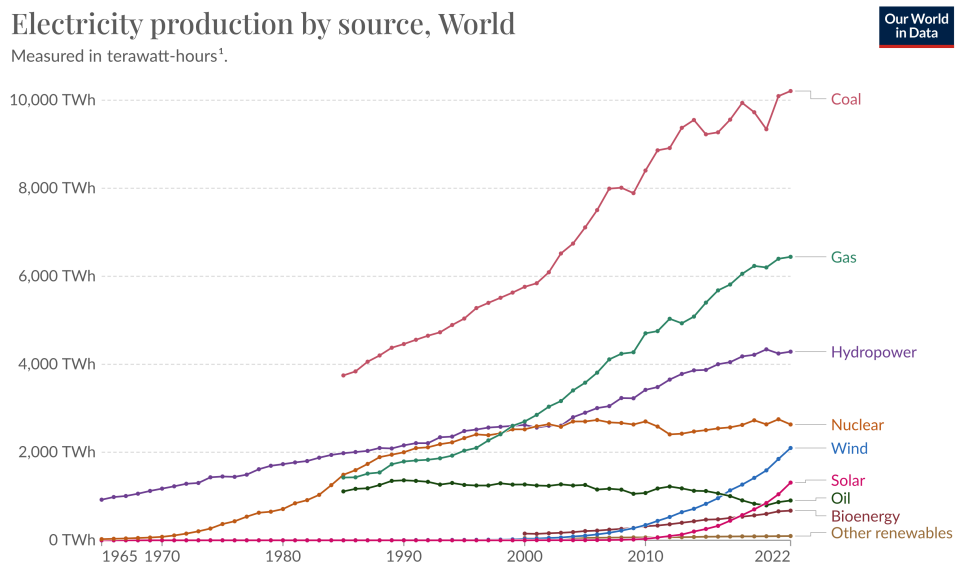


Figure 5.5: Electricity production by source [58]

Chapter 6

Point forecasting

This chapter covers the theory of the most widely used methods for point prediction in the electricity forecasting 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 electricity point forecasting literature. Notice, they are not used per se but usually combined with other more advanced models.

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

6.2 Autoregressive models

Autoregressive models are a standard approach for modelling time series data. Before introducing this popular model in electricity point forecasting, 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. In a nutshell, stationarity means that the distribution of any subsequence of random variables of the stochastic 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 can 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

$$\hat{y}_n = \sum_{i=1}^{p} \psi_i y_{n-i}^{(d)} + \sum_{j=1}^{q} \theta_j \varepsilon_{n-j} + \varepsilon_n \tag{6.2}$$

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

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

where B is the backward shift operator, $B^h y_t = y_{t-h}$ and $\psi(B)$ is shorthand for $\psi(B) = 1 - \psi_1 B - \cdots - \psi_p B^p$. 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 [15]. 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 (ACF) 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

$$\hat{y}_n = \sum_{k=1}^{h} \mu_i X_{n-k} + \sum_{i=1}^{p} \psi_i y_{n-i}^{(d)} + \sum_{j=1}^{q} \theta_j \varepsilon_{n-j} + \varepsilon_n \tag{6.4}$$

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 pattern, 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

$$\psi(B)\Psi(B^P)\nabla^d \nabla_m^D y_t = \theta(B)\Theta(B^Q)\varepsilon_t \tag{6.5}$$

where Ψ and Θ account for the P and Q of the seasonal pattern m respectively.

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 [43]. Approximations for f_i are obtained through an iterative procedure, the backfitting algorithm.

Particularly successful in time series forecasting is the Prophet model [95], developed by Meta. This additive model is capable of handling non linear trends, holiday effects and yearly weekly and daily seasonalities. Thus, the reason why 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 [69]. 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. [23] [101], 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 support vector regression (SVR) we have

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

Where $\mathcal{X}$ denotes the space of the input data 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.

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

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. [91] point out that considering the dual formulation makes our optimisation problem easier to solve. Doing so we have the equivalent optimisation problem

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

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

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

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

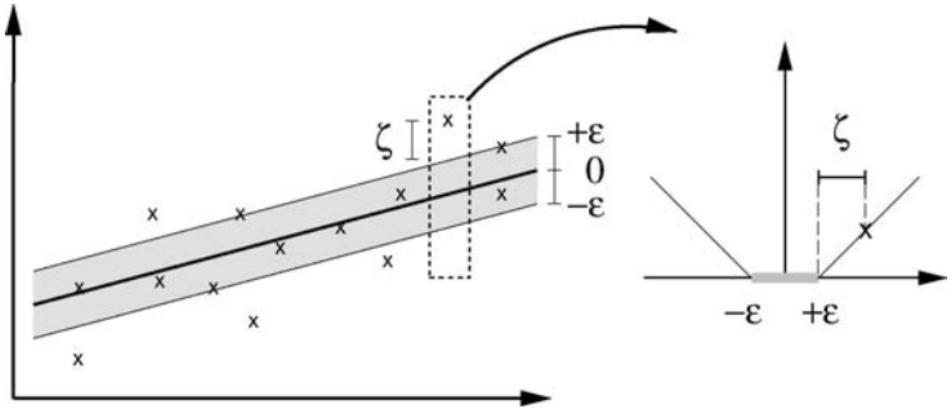


Figure 6.1: Support vector regression [90]

is independent of the feature space dimensionality, but depends only on the number of support vectors; that is those point i for which $(\lambda_i - \lambda_i^*) \neq 0$ Moreover, we do not needd to compute w explicitly in order to evaluate $f(x)$. That is because, equation 6.12 implies

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

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. Consequently for any of the data points with associated $\lambda_i, \lambda_i^* \in (0, C)$, the equality $b = y_i - \langle w, x_i - \varepsilon \rangle$ has to hold.
Furthermore, this implies that the Lagrange multipliers λ_i, λ_i^* may be nonzero only for the samples inside the epsilon insensitive tube, that is only those points may be support vectors.
To get an idea of how the algorithm 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 [83].

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

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

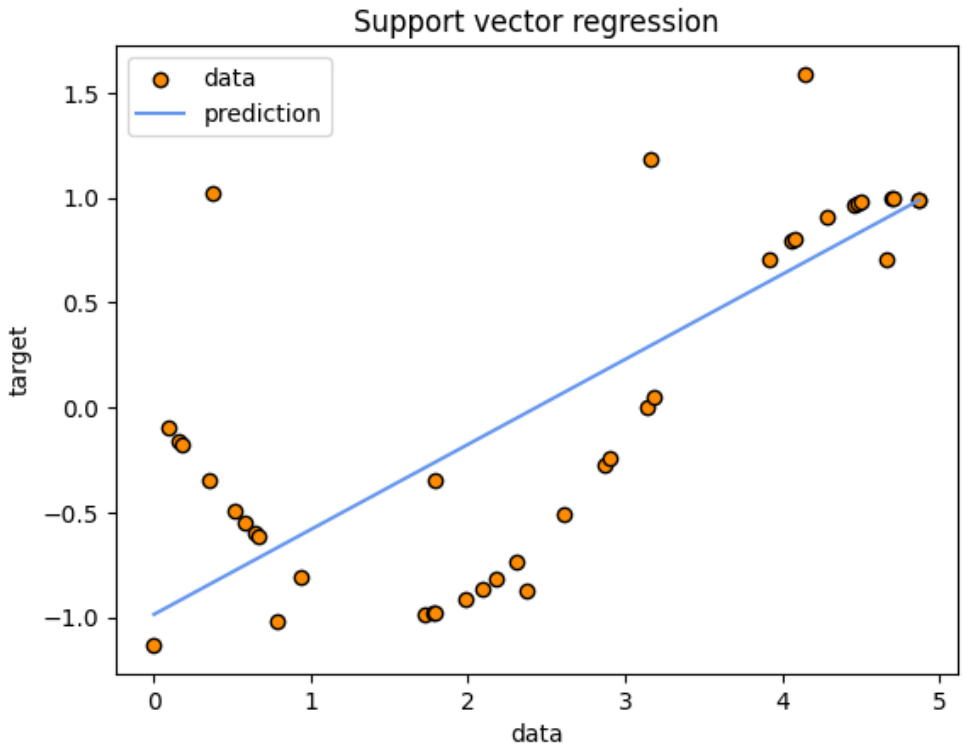


Figure 6.2: Linear support vector regression

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.

6.6.2 Long short term memory

Long short term memory network (LSTM) extends MLP by introducing a cell state in its layers. In this way, the current state of a cell depends 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 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}.

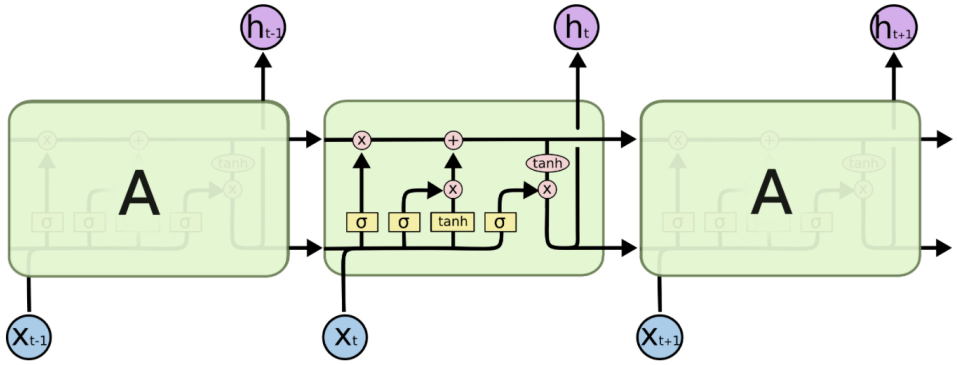


Figure 6.3: Long short term memory cell

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

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

Before proceeding, we need to introduce the representer theorem.

Theorem 6.1 *Let $k : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ on a non empty set $\mathcal{X}$ with corresponding RKHS $\mathcal{H}$, a strictly increasing real valued function $g : [0, \infty) \to \mathbb{R}$, and the loss function $E : (\mathcal{X} \times \mathbb{R}^2)^n \to \mathbb{R} \cup \{\infty\}$. Consider the regularized risk functional $Risk[f] : f \mapsto E((x_1, y_1, f(x_1)), \ldots, (x_n, y_n, f(x_n))) + g(\|f\|)$. 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 a 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

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

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

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

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 of equation 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

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

Setting the gradient to zero

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

We have then, that $\hat{\alpha} = (K + \tau I)^{-1} y$. It follows that 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 (SVR) can be kernelized by swapping the $\mathbb{R}^n$ euclidean dot product of data x with the dot product in the higher feature space $\mathcal{H}$. Doing so, the optimisation problem can be restated as

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

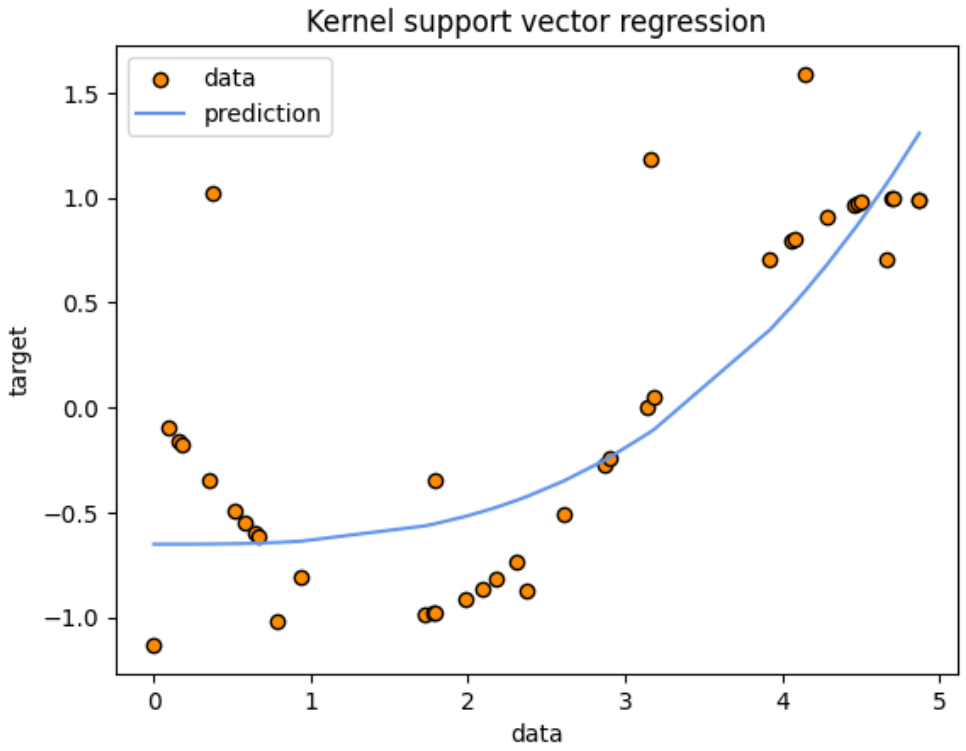


Figure 6.4: Polynomial support vector regression

Letting $\alpha_i = \lambda_i - \lambda_i^*$, our regressor f is then given by

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

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

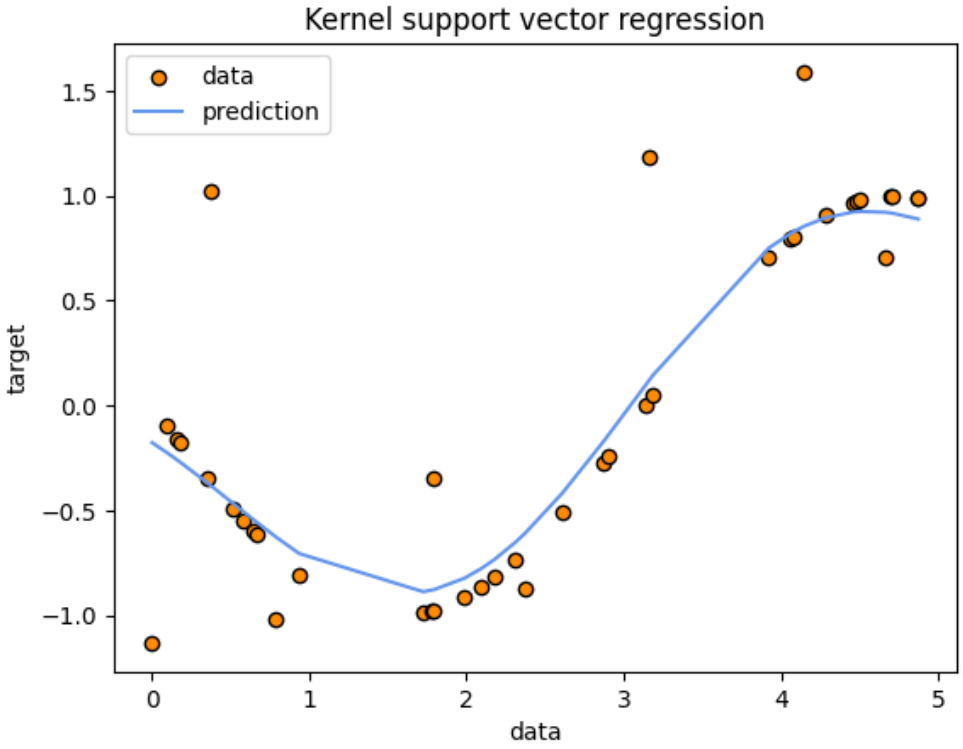


Figure 6.5: Gaussian RBF support vector regression

Chapter 7

Probabilistic forecasting

We saw in the previous chapter how point forecasting relates to predicting the expected value of the target variable. Conversely, in probabilistic forecasting we are interested in modelling a probability distribution over the target variable. This approach enables us to account for prediction uncertainty and to make more informed decisions.

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 the following optimization problems.
For any $0 < q < 1$ consider the pinball loss function from section 4.7, $\rho_q(u) = u(q - \mathbb{I}_{\{u<0\}})$. 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 $\hat{Q}_q = x_i\beta(q)$, can be estimated by solving

$$\hat{\beta}(q) = \arg\min_{\beta} \sum_{i=1}^{n} \rho_q(y_i - x_i\beta) \tag{7.2}$$

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 [16].
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 [72] extends the idea of random forest [17] 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 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 a visualisation consider the pictures 7.1 and 7.2 taken from [42], where the decision tree algorithm is visualised for a regression problem with two independent variables X_2 and X_1.

Suppose now 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

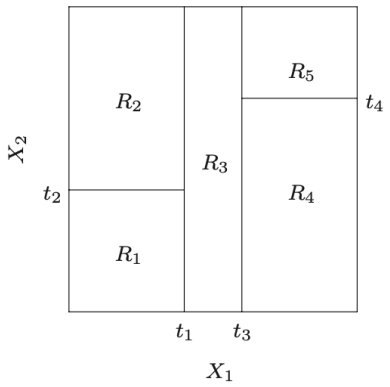


Figure 7.1: two-dimensional feature space partitioned by recursive binary splitting [42]

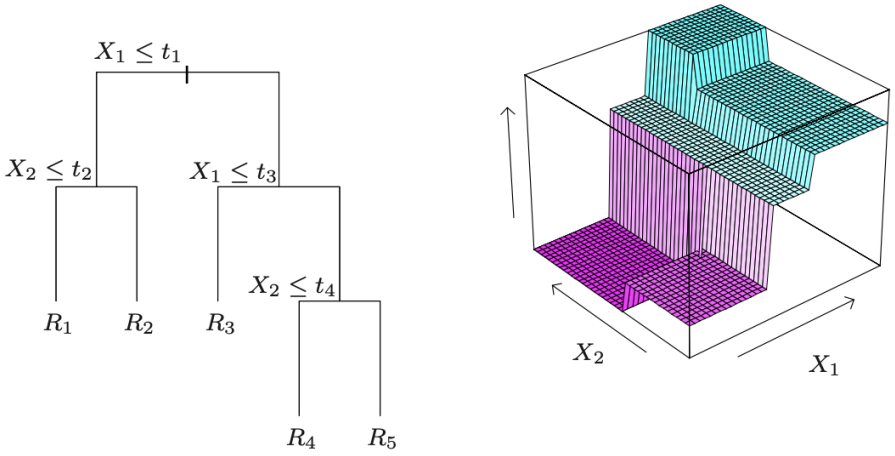


Figure 7.2: partition tree and regression model [42]

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

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

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

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

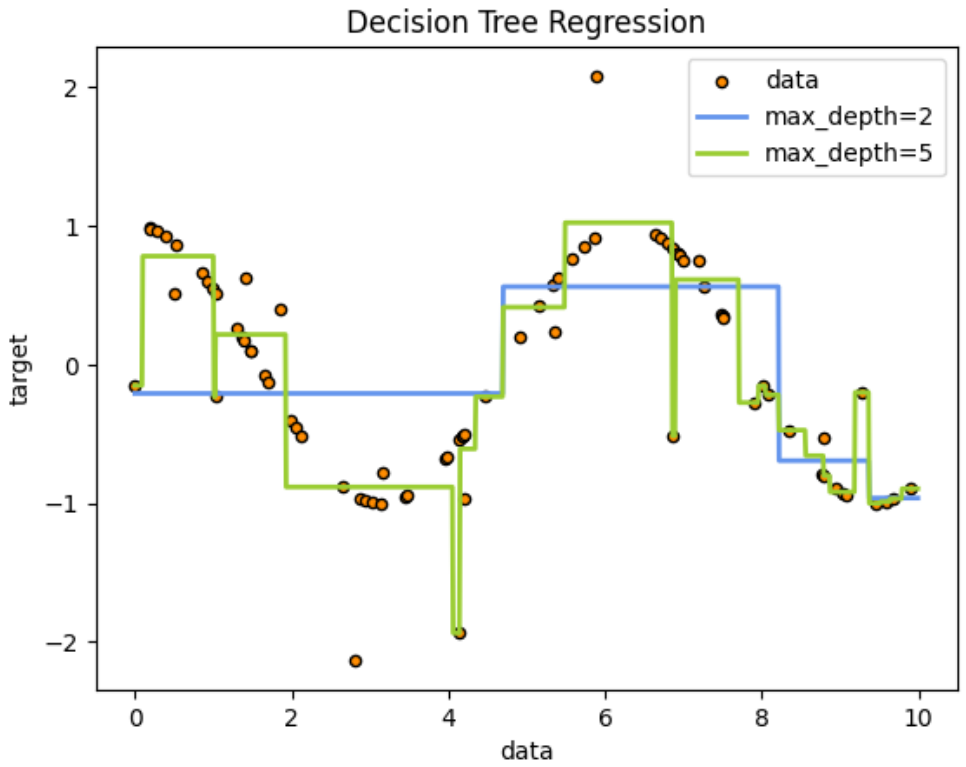


Figure 7.3: decision tree regression

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 depth of tree, threshold on the mean squared error (MSE), minimum number of observations 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 an example where decision trees with different hyperparameters are fitted to a sine wave plus noise. The most popular among these model is the CART [18] 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 or of a prediction but also to improve their estimates.

7.2.3 Bagging

Bagging stands for boostrap aggregation. The bagging estimate is defined 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 (bias-variance tradeoff). 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 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.

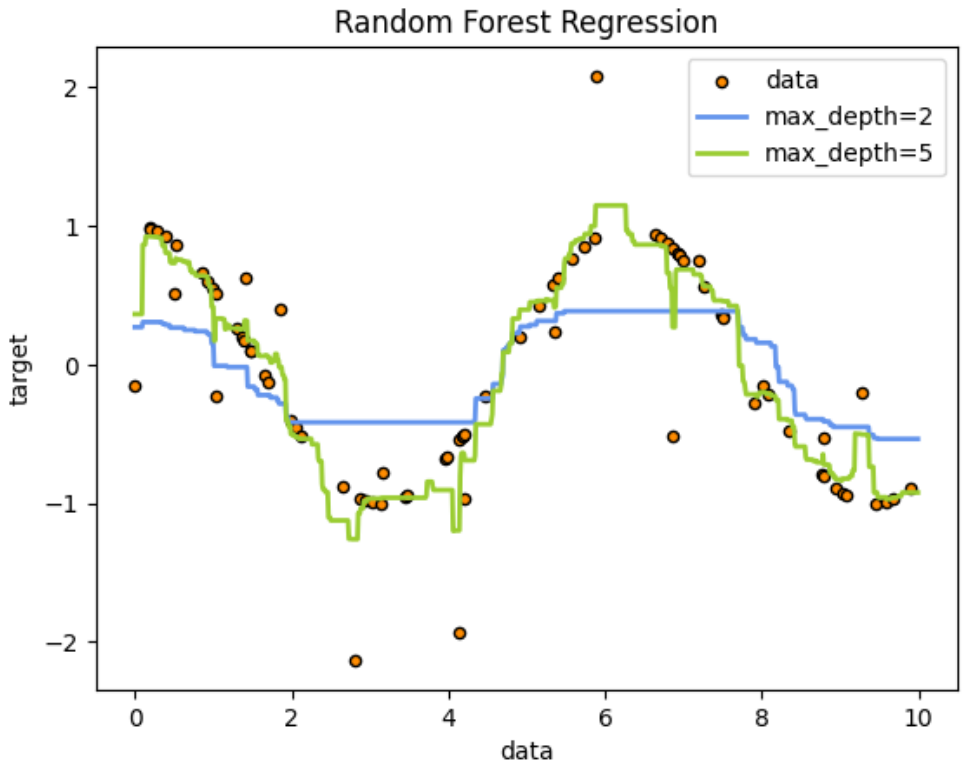


Figure 7.4: random forest regression

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 L 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

Combining several trees $T(\cdot, \Theta_m)$, we obtain 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

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

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

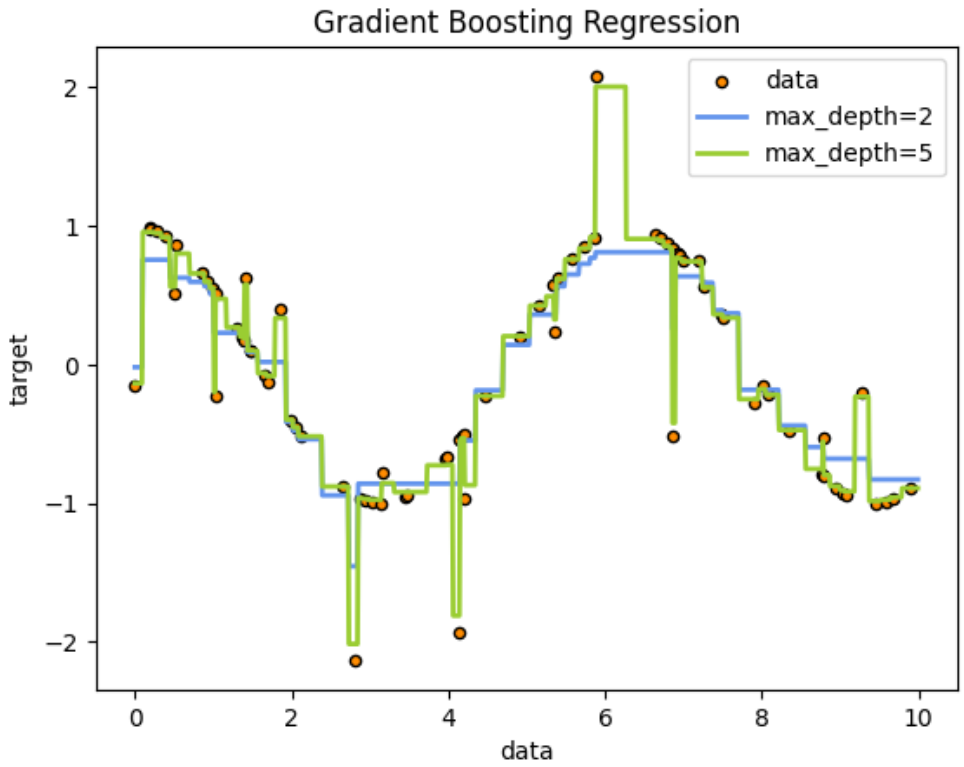


Figure 7.5: gradient boosting regression

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 quantile regression

The idea of quantile regression has been extended to kernel methods by Takeuchi et al. [93]. There, they minimize a risk functional plus regularizer defined as follows.

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

where $f = w + b$, $w \in \mathcal{H}$ and $b \in \mathbb{R}$. Using the link between RKHS and feature spaces, we can rewrite $f(x) = \langle w, \varphi(x)\rangle_{\mathcal{H}} + b$. Doing so we obtain a

minimization problem equivalent to minimizing equation 7.19.

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

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

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

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 $\mathcal{L}$ associated to 7.22

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

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 $\mathcal{L}(x,\lambda,\nu)$ is given by $g(\lambda,\nu) = \inf_x \mathcal{L}(x,\lambda,\nu)$*

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

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

where p^* is the optimal value of your optimization problem. In other words $g(\lambda,\nu)$ is a lower bound for the optimal p^*. Consider now a simple lagrangian

$\mathcal{L}(x, \lambda, \nu) = f(x) + \sum_{i=1}^{n} \lambda_i r_i(x) + \sum_{i=1}^{n} \nu_i h_i(x)$, 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 constraint $\lambda \succeq 0$, 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.

Proposition 7.3 *Slater's condition reads as*

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

Where relint D is the relative interior of $D := \bigcap_{i=0}^{m} \mathrm{dom}(r_i)$

Slater's theorem naturally follows.

Theorem 7.4 *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 an x such that $r_i(x) < 0$. For convexity, a sufficient condition is the positive definiteness of Q in the quadratic programming problem

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

This condition is easily checked by the fact that kernel matrices are by definition positive semidefinite. 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 $0 \leq \xi \leq \mu$ and $0 \leq \xi^* \leq \mu$.
Next, let us consider the other two inequalities and make the following ansatz.

$$w = \Phi^\intercal(\Phi\Phi^\intercal)^{-1}(y - b) \tag{7.27}$$

We then have for any $\xi > 0$ and $\xi^* > 0$ that

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

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, which we will solve in order to get the quantiles prediction of our quantile kernel algorithm.
First, derive the dual function of 7.23.

$$g(\lambda, \lambda^*, \nu, \nu^*) = \inf_{\xi, \xi^*, w, b} \mathcal{L}(w, b, \xi, \xi^*, \lambda, \lambda^*, \nu, \nu^*) \tag{7.29}$$

Setting its derivates to zero

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

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

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

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

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

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

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

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

The kernel quantile regression estimator is then given by

$$f(x) = \sum_{i=1}^{n} \alpha_i k(x_i, x) + b \tag{7.34}$$

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 (KKT), see section 5.5.3 [16]. Thanks to the KKT conditions on the primal optimisation problem we have that $f(x_i) = y_i$ for $\alpha_i \notin \{C(q-1), Cq\}$. To see this, we have to consider the KKT conditions.

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

In our setting we have

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

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

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

Now, let us break into cases

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

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

$$b = y_i - \sum_{i=1}^{n} \alpha_i k(x_i, x) \tag{7.39}$$

7.4.1 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 [57]. 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 [56], 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. Notice, the kernel considered here is the Gaussian RBF. Hyperparameters have been tuned through cross validation, see A.3.

Table 7.1: Pinball loss Melbourne data

	Linear qr	Gbm qr	Quantile forest	KQR
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 conclusions as the one of above, to know more see A.2.

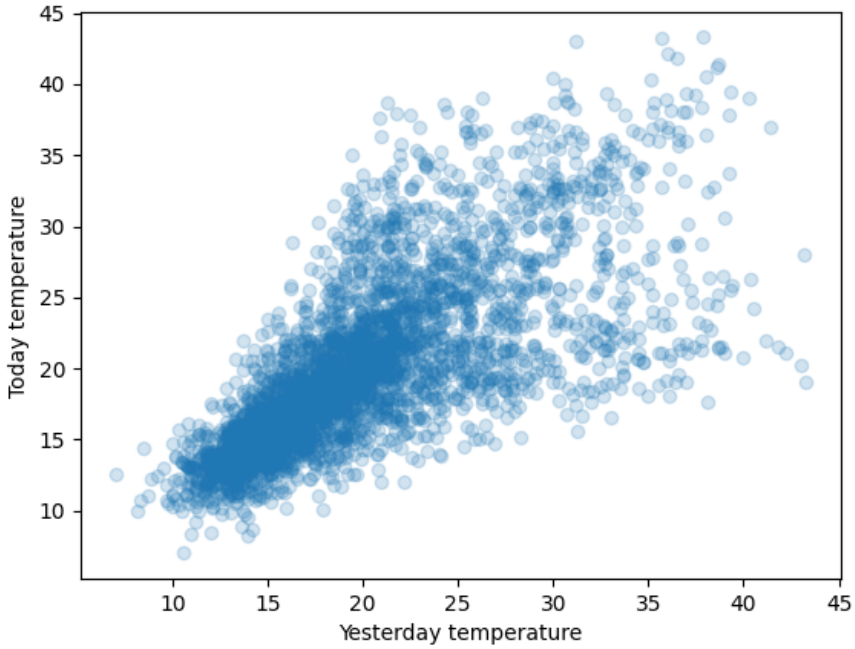


Figure 7.6: Melbourne temperatures dataset

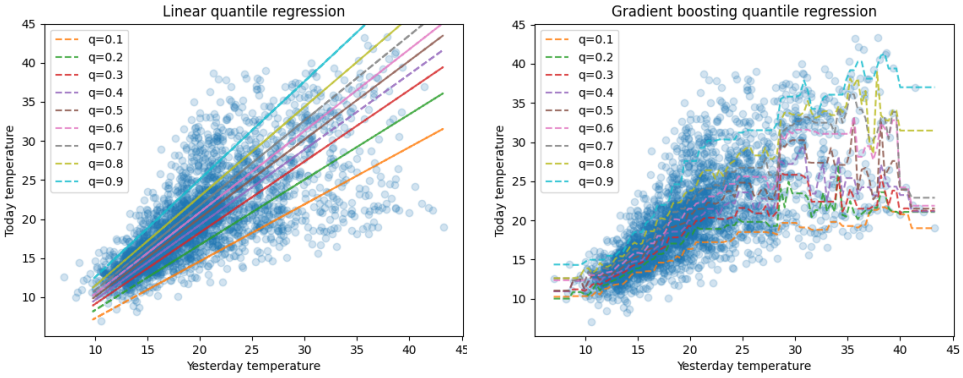
Table 7.2: Pinball loss quantile-wise Melbourne data

Quantiles	Linear qr	Gbm qr	Quantile forest	KQR
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	KQR
MAE	3.253882	3.134805	3.095041	**3.024336**

We conclude this chapter with reporting the same kind of tables 7.4, 7.5, 7.6 and figure 7.8 comparing various kernel functions.



(a) Linear quantile regression

(b) Gradient boosting quantile regression

(c) Quantile forest

(d) Kernel quantile regression, Gaussian RBF

Figure 7.7: Quantile regressors comparison Melbourne weather data

Table 7.4: Kernels comparison pinball loss Melbourne data

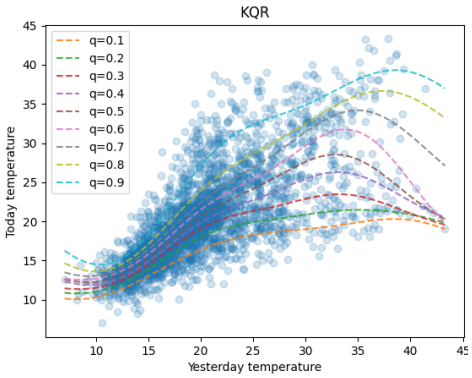
Kernel	Pinball loss
Gaussian RBF	10.031708
Absolute Laplacian	10.056884
Gauss RBFx Absolute Laplacian	10.150826
Cosine	16.253973
Linear	10.463867
Polynomial	10.089147
Sigmoid	16.253973
Chi squared	**10.023732**
Matern 0.5	10.056883
Matern 1.5	10.040288
Matern 2.5	10.024495
Periodic	15.946272

Table 7.5: Kernels comparison pinball loss quantile-wise Melbourne data

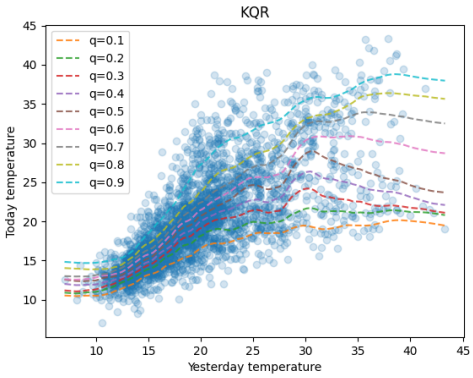
Kernel/Quantiles	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Gauss RBF	**0.540145**	0.903193	1.173627	1.368022	1.456053	1.447470	1.331375	1.096649	0.715013
Laplacian	0.542499	0.906517	1.177309	1.371198	1.462111	1.453119	1.340193	1.095991	0.707947
Gauss RBFxLaplacian	0.545787	0.915559	1.189965	1.387109	1.479901	1.475541	1.350803	1.101151	**0.704922**
Cosine	0.755479	1.343973	1.802603	2.123425	2.307123	2.367123	2.262603	1.971644	1.320411
Linear	0.565663	0.947889	1.250009	1.437863	1.506723	1.478379	1.358865	1.135702	0.782660
Polynomial	0.542573	0.908681	1.178357	1.362351	1.450249	1.455228	1.339822	1.107768	0.744118
Sigmoid	0.755479	1.343973	1.802603	2.123425	2.307123	2.367123	2.262603	1.971644	1.320411
Chi squared	0.541430	0.906352	1.173164	**1.358862**	**1.444130**	**1.440915**	1.332140	1.100262	0.726625
Matern 0.5	0.542499	0.906517	1.177309	1.371198	1.462111	1.453119	1.340193	1.095991	0.707947
Matern 1.5	0.541410	0.902901	1.173943	1.371218	1.462212	1.449296	1.333655	**1.095092**	0.710560
Matern 2.5	0.541233	**0.902231**	**1.172001**	1.366449	1.456505	1.444124	**1.331251**	1.095674	0.715029
Periodic	0.759452	1.352747	1.809921	2.125765	2.314718	2.379657	2.274711	1.976768	1.328630

Table 7.6: Kernels comparison mean absolute error Melbourne data

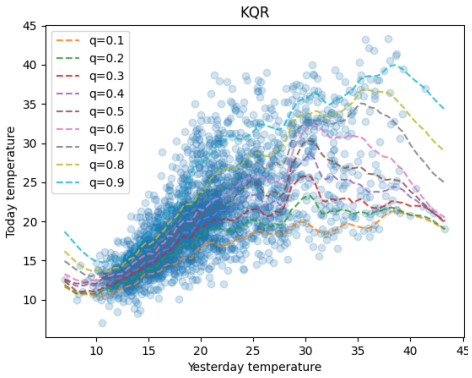
Kernel	MAE
Gauss RBF	3.024336
Laplacian	3.038633
Gauss RBFxLaplacian	3.070812
Cosine	4.849589
Linear	3.142352
Polynomial	3.064472
Sigmoid	4.849589
Chi squared	3.022778
Matern 0.5	3.038633
Matern 1.5	3.023060
Matern 2.5	**3.021949**
Periodic	4.880673



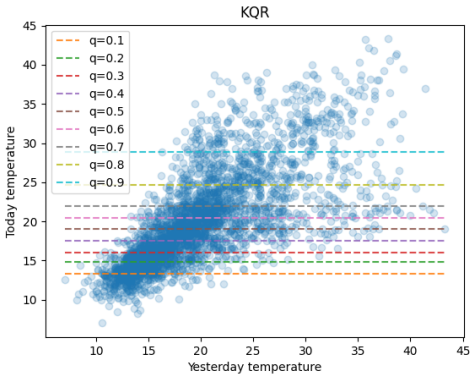
(a) Gaussian RBF



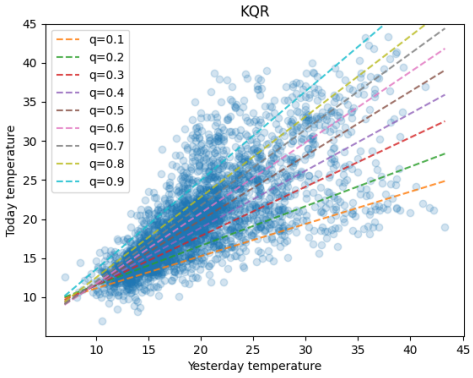
(b) Laplacian



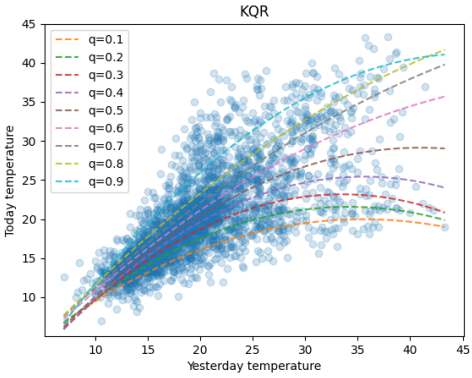
(c) Gauss RBFxLaplacian



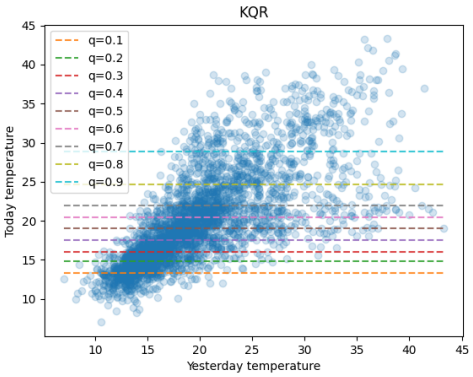
(d) Cosine



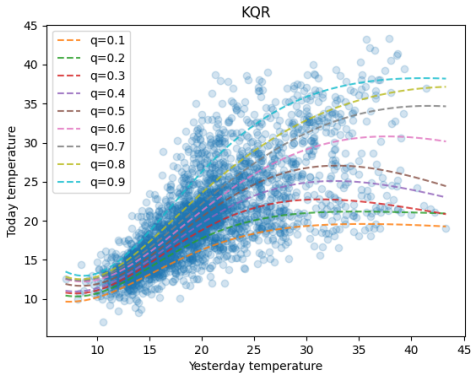
(e) Linear



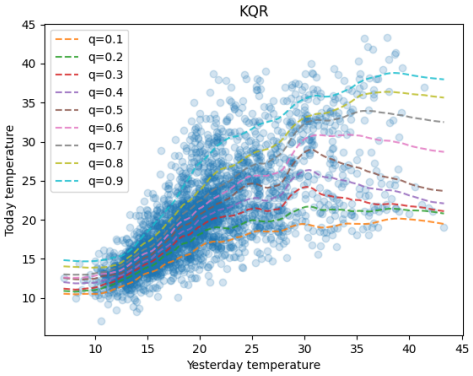
(f) Polynomial



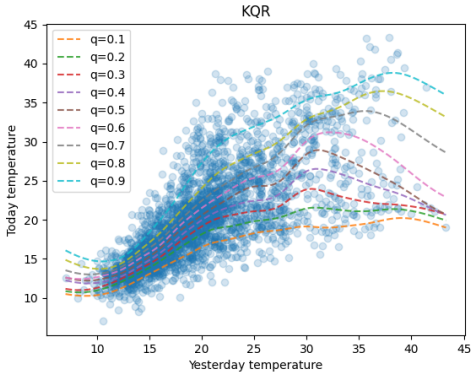
(g) Sigmoid



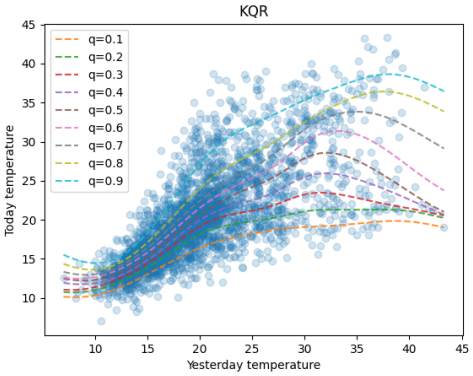
(h) Chi squared



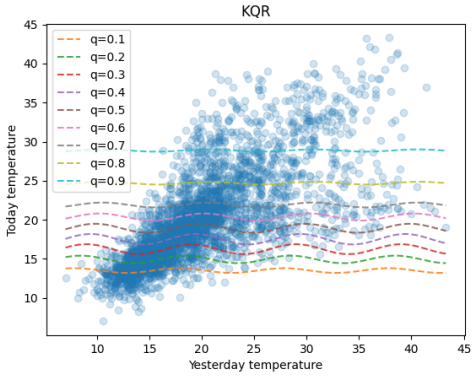
(i) Matern 0.5



(j) Matern 1.5



(k) Matern 2.5



(l) Periodic

Figure 7.8: Kernel quantile regressors comparison Melbourne data

Chapter 8

Exploratory analysis and data extract transform load pipeline

This section covers the datasets used in our experiments, in addition, attributes and features will be thoroughly described. Next, we will explain the extract transform load pipeline (ETL) that we set up in order to ease the workflow of our studies. Finally, in order to to better understand patterns within the data, we carry out an exploratory data analysis.

8.1 GEFCom2014

The first class of datasets we identified were the ones from the GEFCom 2014 competition. The data is freely hosted on Dr. Hong blog [52]. The main reason is that, these datasets are considered an 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 focused on providing forecasts for the load and price quantities.

8.1.1 GEFCom2014 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.

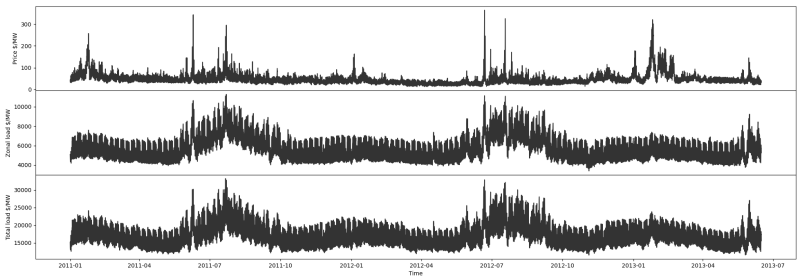


Figure 8.1: Price track GEFCom2014

EDA

In exploring this data we started off by plotting the time series of the zonal prices alongside with the total load and zonal load, see figure 8.1. After that, we plotted the scatter plot for inspecting the correlation between the predictors and the dependent variables, figure 8.2 and figure 8.3. From these visualisations, we can see their goodness for predicting zonal price since there exists a clear correlation between the candidate predictors and the target variables.

8.1.2 GEFCom2014 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 have a txt file with the competition instructions and 15 subfolders 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. Any data from the various task folders are shifted by one month between eachother.
In order to compare our model performance with the winning entries of the GEFCom2014 load competition, we will refer to the Provisional_Leaderboard_V2.csv

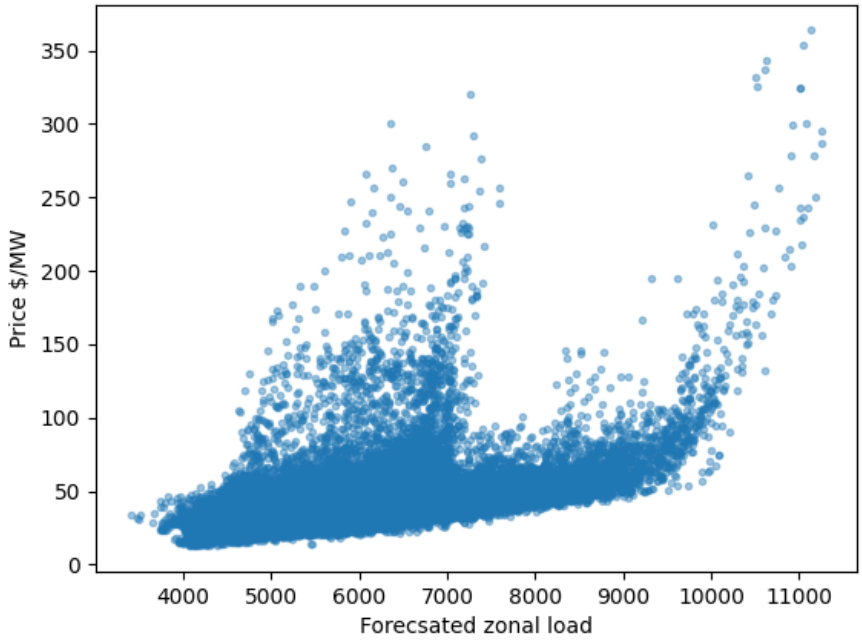


Figure 8.2: Zonal price versus zonal load GEFCom2014

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.

EDA

We applied the same kind of EDA to the GEFCom load data. Figure 8.4 depicts the hisorical load while figure 8.5 reports the historical weather temperature. Finally, figure 8.6 shows the scatter plot between the the load and the weather. These plots motivate the selection of weather temperature as one of the predictors for the load.

GEFCom ETL

The provided data is raw, hence the need for a standardized extract transform and load pipeline. Firstly, we drop all Nans and zeros. Notice, the provided data for the challenge has a timestamp column not coherent to any time format. Thus, we first pass the data through a method extracting time information: hour, day, month and year. This info is then combined into a datetime object, we selected the ISO format. Next, since the data does not provide the location for the target utility, the best we can do is selecting the

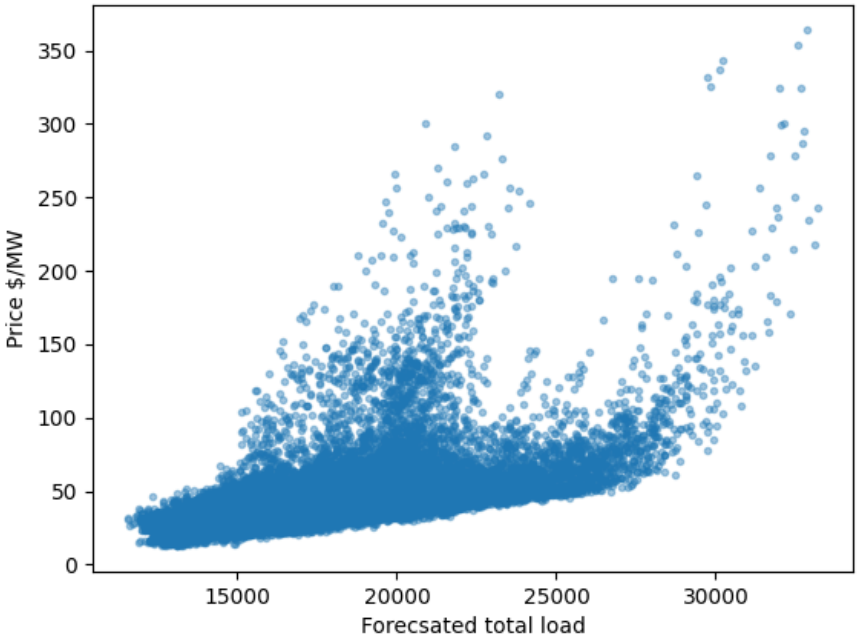


Figure 8.3: Zonal price versus total load GEFCom2014

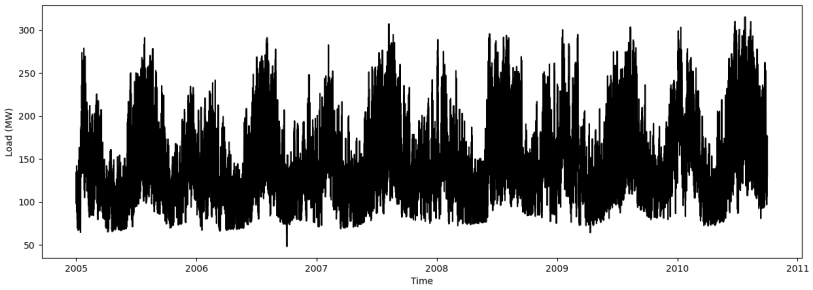


Figure 8.4: Historical load GEFCom2014

average of weather temperatures as predictor variables. This independent variable is named w_avg. The same thing applies in the cleaning of the test data with the addition of an if clause for handling the different naming convention and formatting used in the last track of the challenge. The python etl pipeline has been then automated to loop every task folder through bash scripting.

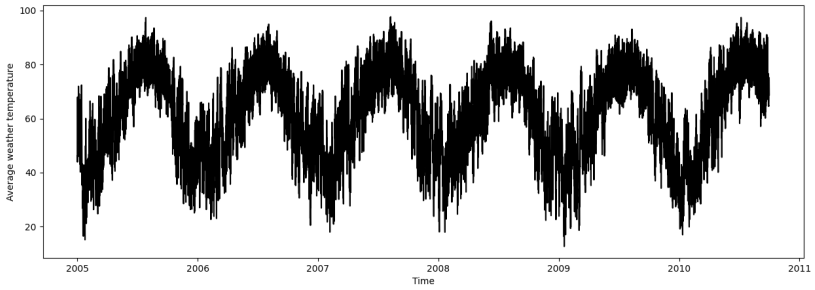


Figure 8.5: Historical weather GEFCom2014

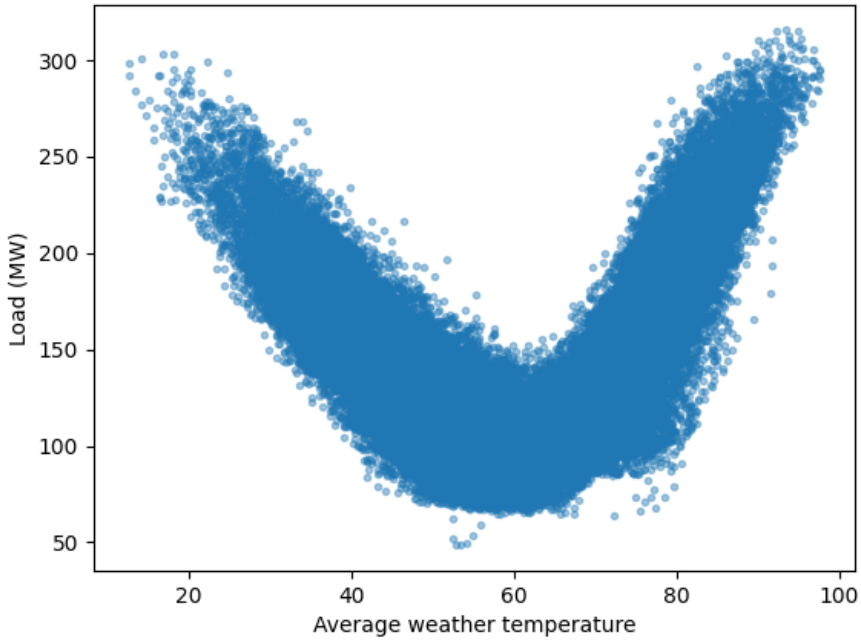


Figure 8.6: Load versus weather GEFCom2014

8.2 Germany Switzerland Austria data

Next, we identified various data sources for Germany, Switzerland and Austria in order to further investigate the techniques studied in this thesis work. Taking inspiration from the GEFCom2014 format, we firstly created a load track for Germany and Switzerland. The source of data has been the energy charts website. It provides data for electricity power, electricity potential capacity, energy generation, market prices, environmental measurements and also scenario simulations. Notice, this data is retrieved from the European

network of transmission system operators for electricity ENTSOE and made available thanks to the Fraunhofer Institute for Solar Energy Systems ISE. In this experiment, the goal is comparing kernel quantile regression to the other state of the art quantile regressors, see section 7. The target dependent variable is the national load while as predictors we took the weather temperature, the wind speed, the date expressed in terms of the three attributes day, month and hour, a categorical variable for the day of the week and a binary categorical variable accounting for the holiday effect.

In the second experiment we compared various kernels in the context of load foreacsting. For this study we considered the data of SECURES-Met. This data consists of various metereological attributes relevant for modelling electricity demand [30]. The dataset is made up of historical data until end of 2020, while from 2021 onwards the data consists of forecasts made by a metereological model developed by the European Centre for Medium-Range Weather Forecasts (ECMWF). Notice, the forecasted data is splitted in two files according to the scenarios RCP4.5 and RCP8.5. In our experiment we considered the former. In this setting the independent variables available are direct irradiation, global radiation, power from reservoir plants, power from run of the river plants (ROR), weather temperature and wind potential.

ETL

From energy charts we downloaded the historical load along with the weather temperatures and wind speeds forecasts. Notice, the data request to the environmental variables at hourly frequency for a whole year is broken. Therefore, we downloaded each month separately and then transformed them into a single csv for the whole year.

For what concers the data from SECURES-Met we first identified the data interesting to us. Next, we formatted all the csv to a coherent format, gave coloumns meaningful names, sliced for the years 2020 (train) and 2021 (test) and finally we merged all single csv_s together.

All the manipulations involved in the ETL pipeline have been carried out by combining python and bash scripting.

EDA

Following are reported the EDA plots alongside with insights that can be extracted from them. We choose to focus the following analysis on the Swiss data only, given that the plots for the German data are pretty similar. To get started we visualized the historical load data in figure 8.7. As expected, electricity load exhibits a seasonal patter with highs in the winter and lows during the summer season. Next we built scatter plots of the independent continous variables against the load as means of visualizing correlation

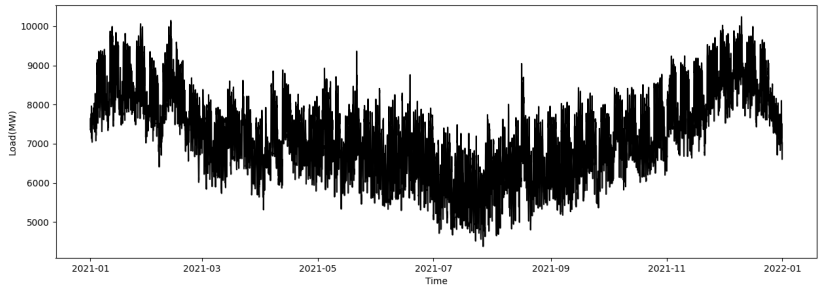


Figure 8.7: Switzerland historical load (2021)

between them. Figure 8.8 plots load against temperatures while figure 8.9 depicts load against wind speed. From these plots we can clearly see there exist a relationship that ties wind speeds and temperatures to load.

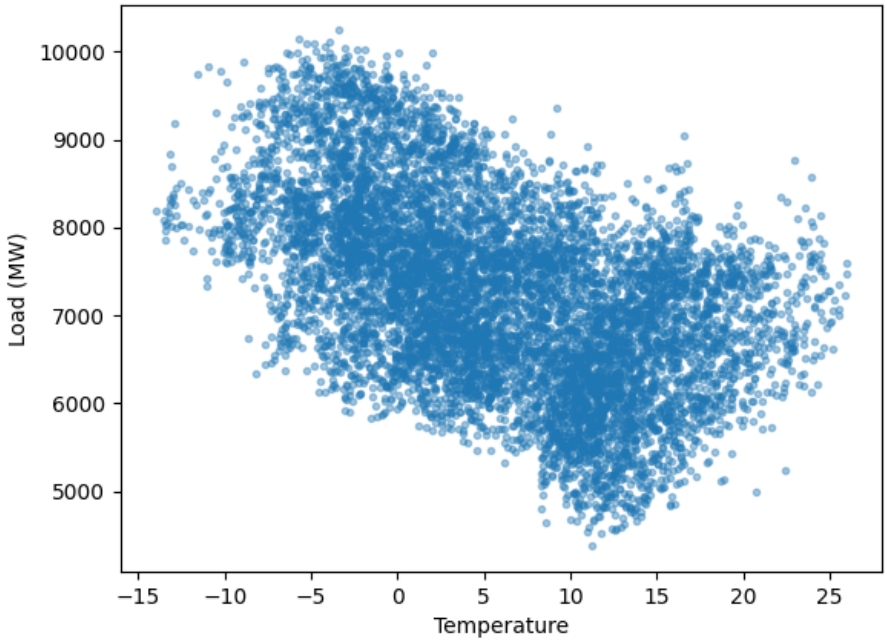


Figure 8.8: Load vs temperature Switzerland (2021)

Following, we used box plots in order to visualize the relations between categorical independent variables and the dependent one. Such visualisation are reported in figure 8.10 and figure 8.11. We can see that weekdays and workdays involve a higher load and posses a wider range.

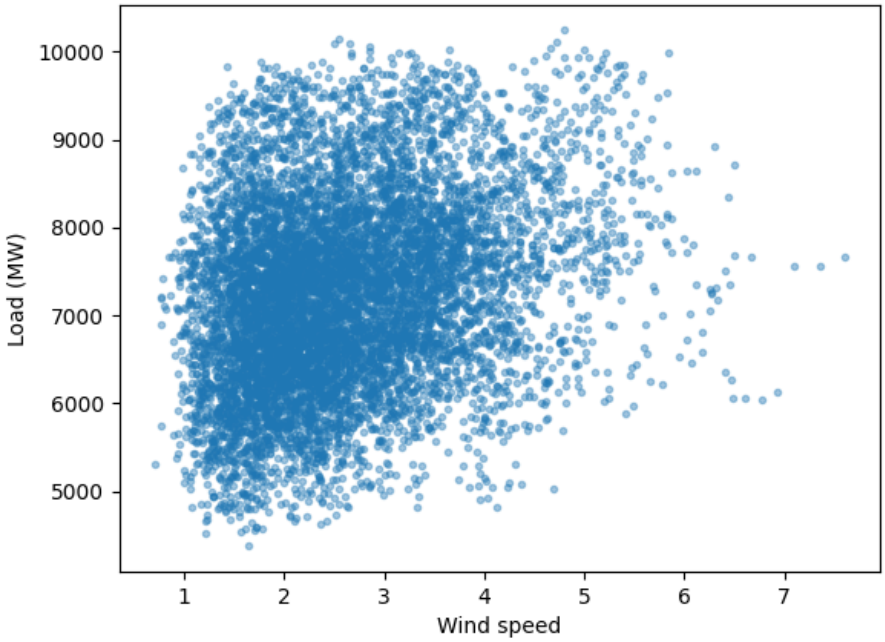


Figure 8.9: Load vs wind speed Switzerland (2021)

We conclude this EDA by reporting the heatmap of independent variables alongside with the load in figure 8.12. Temperature, wind speed and day of week show a meaningful correlation. This is a clear indication of their suitability as predictors for forecasting the load.

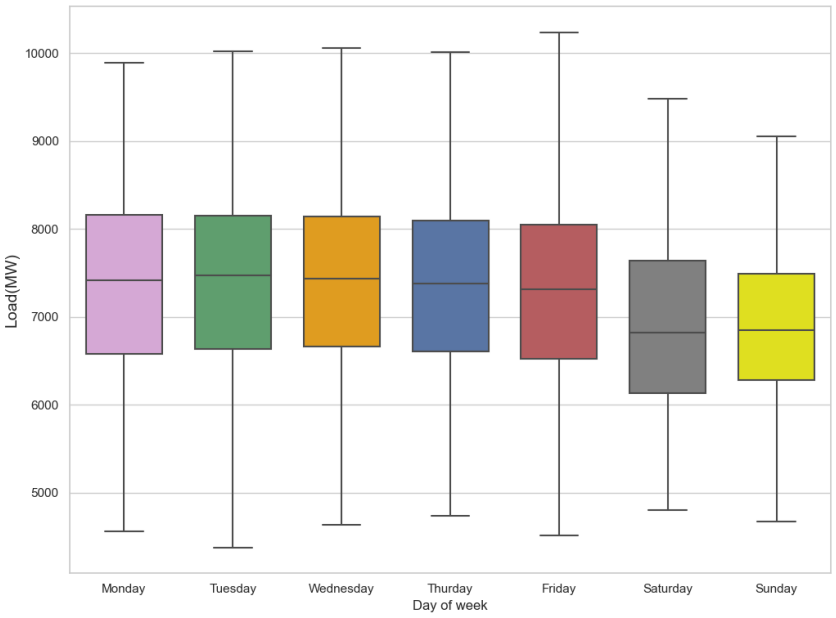


Figure 8.10: Load versus day of week boxplot Switzerland (2021)

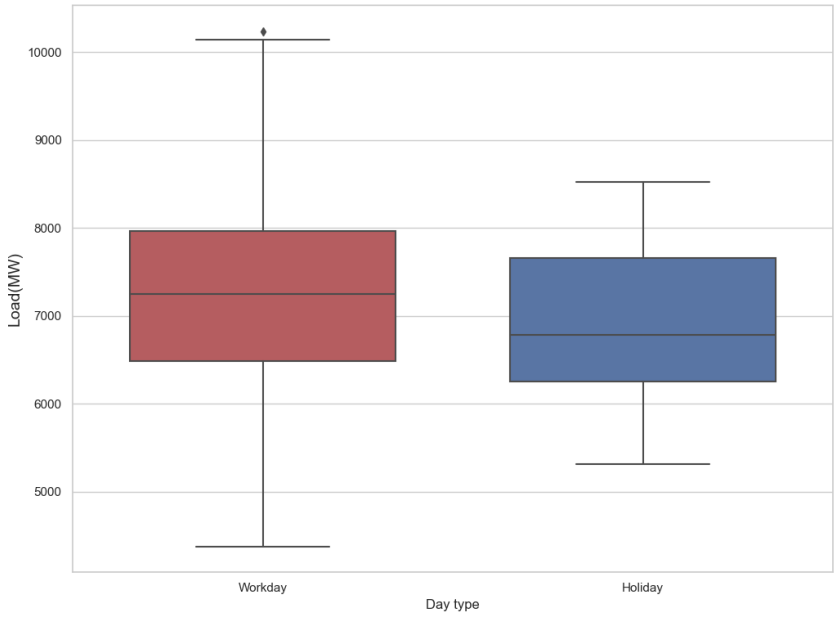


Figure 8.11: Load versus day type boxplot Switzerland (2021)

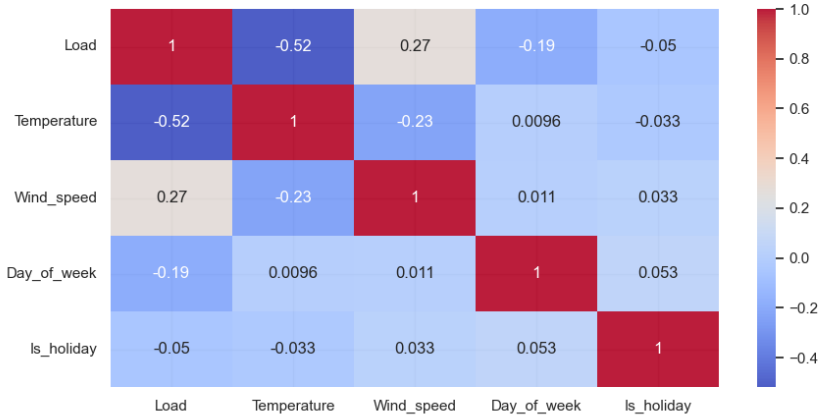


Figure 8.12: Switzerland load heatmap (2021)

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 on https://github.com/luca-pernigo/ThesisKernelMethods. All experiments have been carried out on a 3.2 GHz 16GB Apple M1 Pro.

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

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 model has been applied by employing the python API provided by Meta.

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 Long short term memory

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.

9.2 Probabilistic forecasting

9.2.1 Linear quantile regression

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

9.2.2 Quantile gradient boosting machine

The implementation of GradientBoostingRegressor from the sklearn.ensemble submodule has been used.

9.2.3 Quantile forest

The implementation of quantile_forest.RandomForestQuantileRegressor has been used. This estimator is compatible with scikit-learn API [60].

9.2.4 Kernel quantile regression

Kernel quantile regression had no previously implemented python 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, the kernel family and its corresponding hyperparameters.
In the fit method, we set up and solve the convex optimisation problem through the interior point method. This algorithm is taken from the cvxopt

library, see its offical manual [98] for a reference. 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 and provide them 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. Moreover, this code is compatible with sklearn in built kernel functions sklearn.metrics.pairwise and sklearn.gaussian_process.kernels, functions supported by our kernel quantile regression include: gaussian rbf, absolute laplacian, matern 3/2, matern 5/2, linear, cosine, sigmoid, periodic, polynomial and custom composition of kernels.

Chapter 10

Experiments analysis

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

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: a binary variable for holidays, where holiday= 1 and working day= 0
- 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 figure 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.

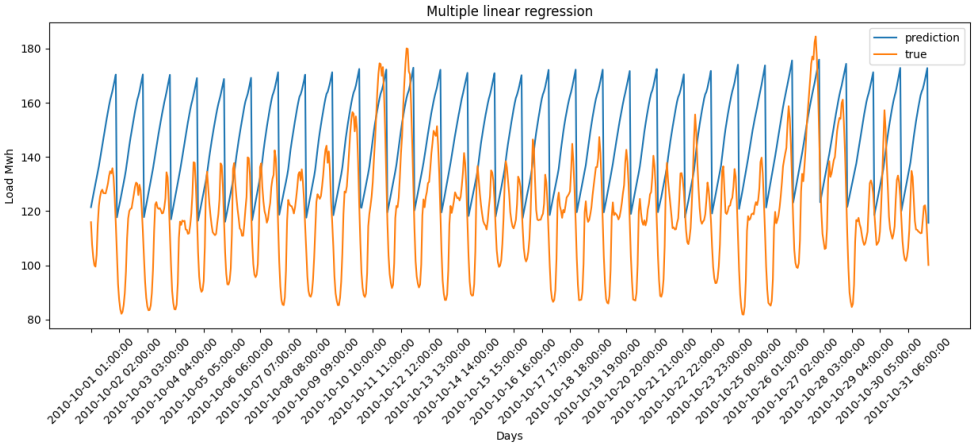


Figure 10.1: Multiple linear regression prediction

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 [27] 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 dependent variable. Doing so, RMSE went down to 10.29. Forecast is visualized in 10.4.

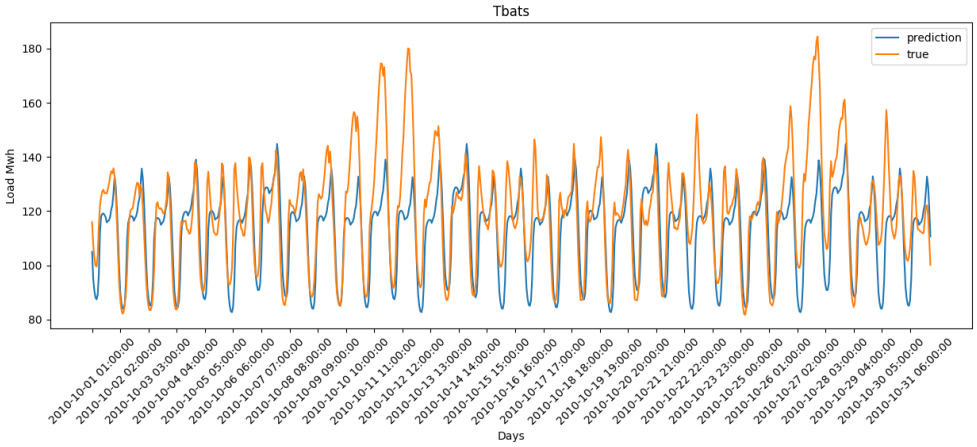


Figure 10.2: Tbats prediction

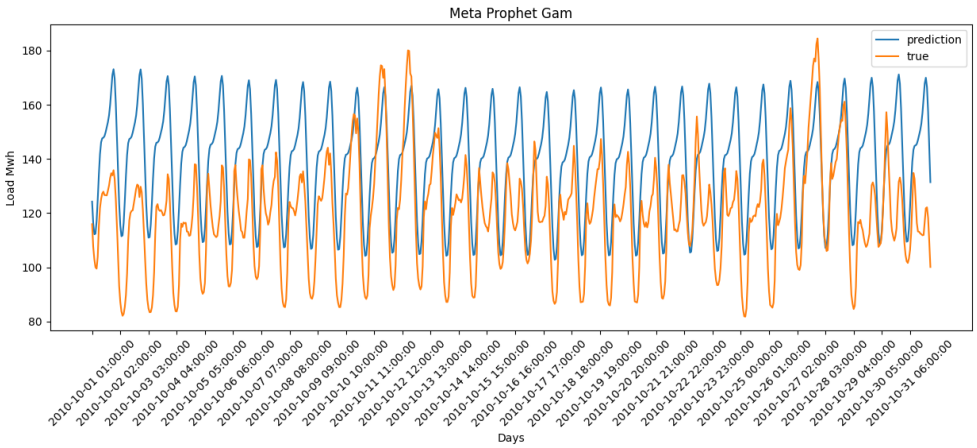


Figure 10.3: Prophet prediction

Prophet alongside with the right features is a good model in the context of electricity load 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 10.5, RMSE is 12.54. What it can be said is that, K-nearest neighbour is capable of predicting prices well by averaging past data.

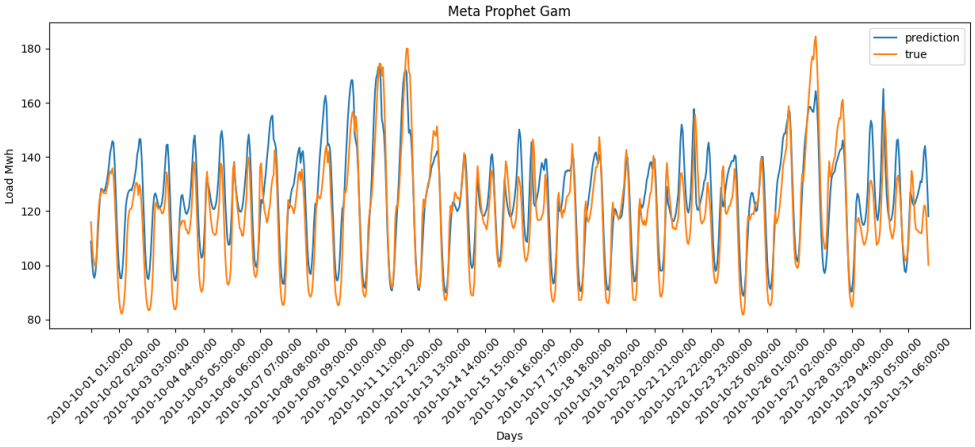


Figure 10.4: Prophet model_2 prediction

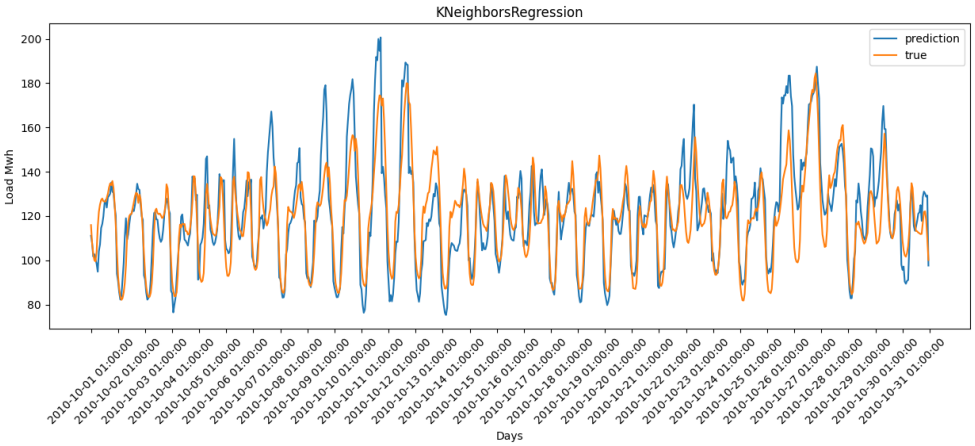


Figure 10.5: K-nearest neighbour regression

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 10.6. 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 the tuning phase. The final

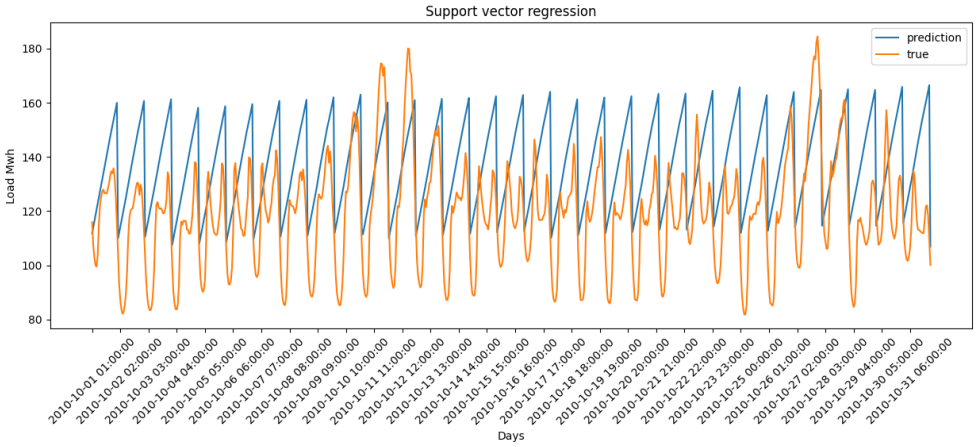


Figure 10.6: Support vector regression prediction

architecture we set on follows

- hidden_size= 64
- num_layers= 2
- output_size= 1
- num_epochs= 30
- learning_rate= 0.001
- batch_size= 32
- window_size= 24

LSTM forecast is reported in figure 10.7, it achieves a RMSE of 9.1782

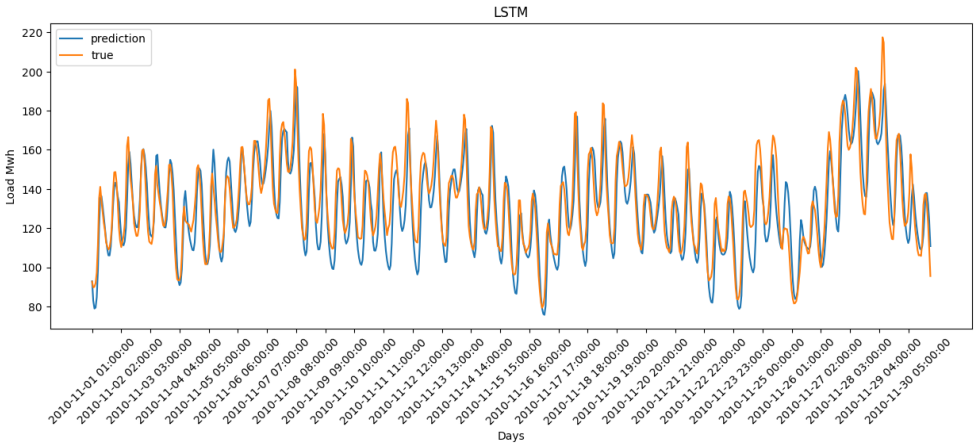


Figure 10.7: Long short term memory prediction

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 the RBF kernel bandwith and the regularization constant. The RMSE achieved is 9.86, figure 10.8 reports the prediction. We can observe that,

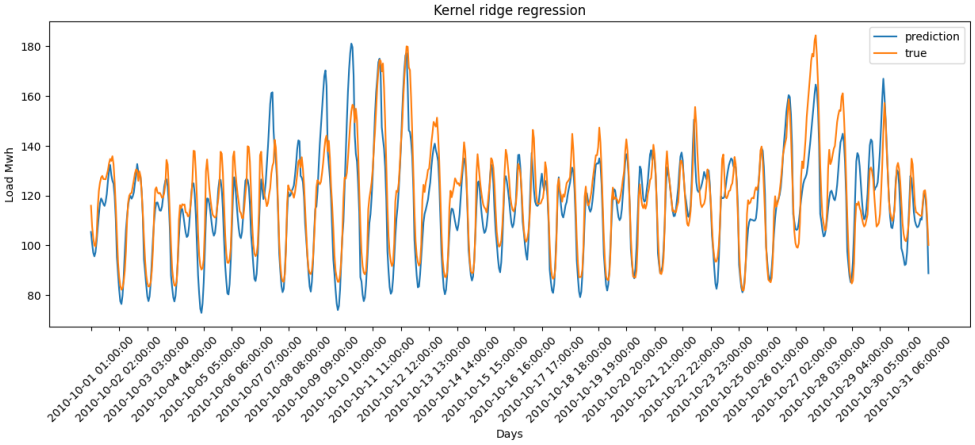


Figure 10.8: Kernel ridge prediction

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 10.9.

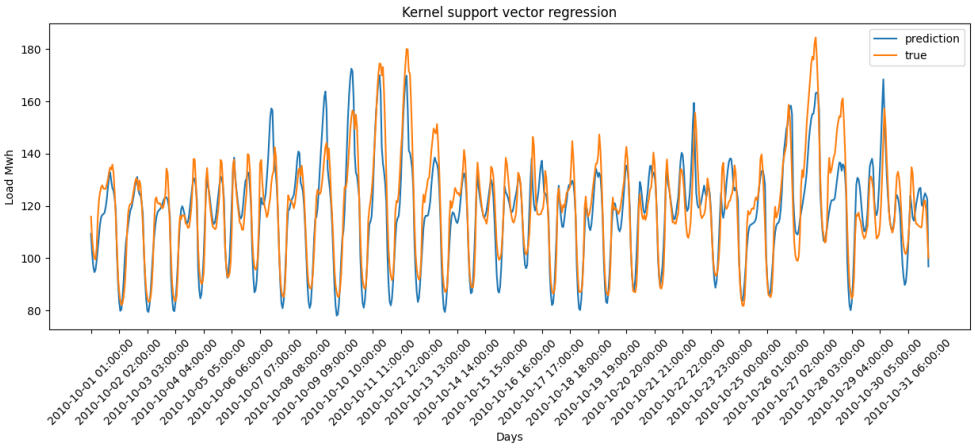


Figure 10.9: Kernel support vector prediction

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

10.1.9 Results

This section reports the tables 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.

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**

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; with KrnSVR being the top one.

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**

Finally table 10.3 reports the MAPE score for each method. Its results are in line with the other tables, hence suggesting the goodness of kernel methods.

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

We continue our experimental analysis by considering the probabilistic framework. The goal of this section is comparing between different quantile regressors and study the choice of the kernel function.

10.2.1 Quantile regressors comparison

Using the load data from energy charts for Germany and Switzerland, we proceeded to compare between quantile regression algorithms. In this study, we trained our models on the entire 2021 data and then we tested them and measured their score on the 2022 data. Notice, for kernel quantile regression (KQR), the kernel of choice in this setting is the absolute Laplacian. Additionally notice, the loss function utilized in this section is the normalized pinball loss. Here, we built two models. In the first, the national load is forecasted by taking as predictor the following variables

- Weather temperature
- Wind speed

Results for Germany are reported in table 10.4 and figure 10.10 while results for Switzerland can be found in table 10.5 and figure 10.11.

Table 10.4: Pinball loss for load in Germany (2022)

Quantile	LQR	GBMQR	QF	KQR
0.1	0.04517	0.02890	0.02891	**0.02767**
0.2	0.08086	0.04745	0.04906	**0.04670**
0.3	0.10820	0.06130	0.06254	**0.05981**
0.4	0.12722	0.07031	0.07136	**0.06834**
0.5	0.13886	0.07371	0.07428	**0.07247**
0.6	0.14182	0.07129	0.07381	**0.07054**
0.7	0.13591	0.06227	0.06237	**0.06215**
0.8	0.11869	0.04741	**0.04723**	0.04730
0.9	0.08458	0.02687	**0.02668**	0.02698
CRPS	0.10904	0.05439	0.05514	**0.05355**

Next, we extended the models to take into account the impact of two additional categorical variables

- Is holiday: a binary variable for holidays, where holiday= 1 and working day= 0
- Day of week: an ordinal categorical variable corresponding to the day of the week, e.i. Monday= 0,. . . , Sunday= 6

Table 10.5: Pinball loss for load in Switzerland (2022)

Quantile	LQR	GBMQR	QF	KQR
0.1	0.04258	0.01909	0.01899	**0.01896**
0.2	0.07460	0.03076	0.03046	**0.03041**
0.3	0.10012	0.03838	0.03903	**0.03814**
0.4	0.11910	0.04311	0.04378	**0.04294**
0.5	0.13095	0.04499	0.04489	**0.04466**
0.6	0.13426	0.04387	0.04397	**0.04347**
0.7	0.12855	0.03928	0.03948	**0.03894**
0.8	0.11201	0.03083	0.03166	**0.03066**
0.9	0.07884	0.01878	0.01878	**0.01841**
CRPS	0.10233	0.03434	0.03456	**0.03407**

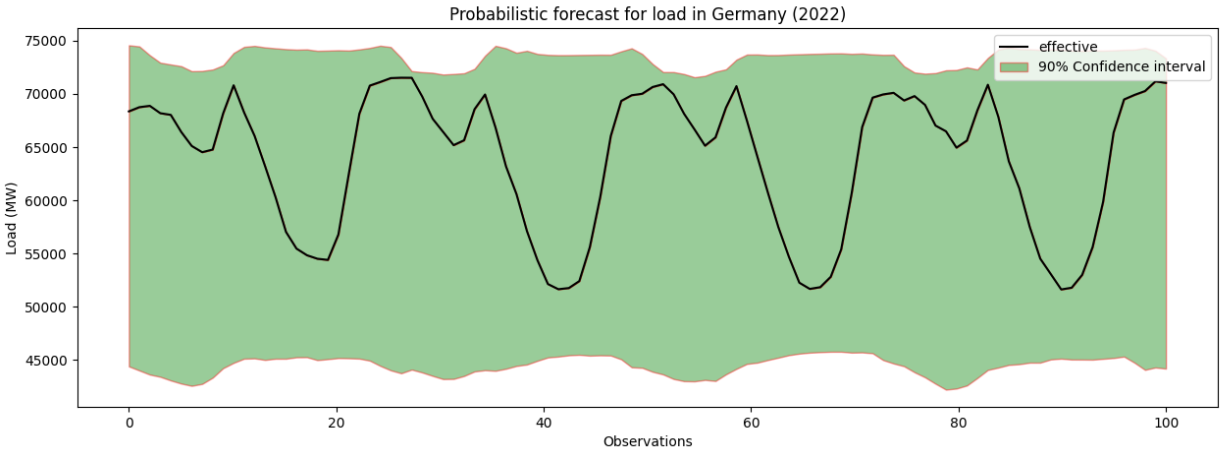


Figure 10.10: Load 90% confidence interval for Germany Energy charts (2022) data using KQR Laplacian. The black line is the observed path for the load. The 90% confidence interval bands are plotted in green. Lower and upper red lines denote the 95% and 5% quantile forecast respectively.

Table 10.6 reports the model scores on the German data while table 10.7 is the one corresponding to the Suisse data.

What can be concluded from this first study is that kernel quantile regression equipped with the absolute Laplacian kernel outperformed almost always its contenders.

10.2.2 Kernel function choice

This section analyzes which kernel function best fits to the characteristics of the load data. The kernel functions considered are: Gaussian RBF, absolute Laplacian, Matern 1.5, Matern 2.5, Linear, Periodic, Polynomial, Sigmoid, and

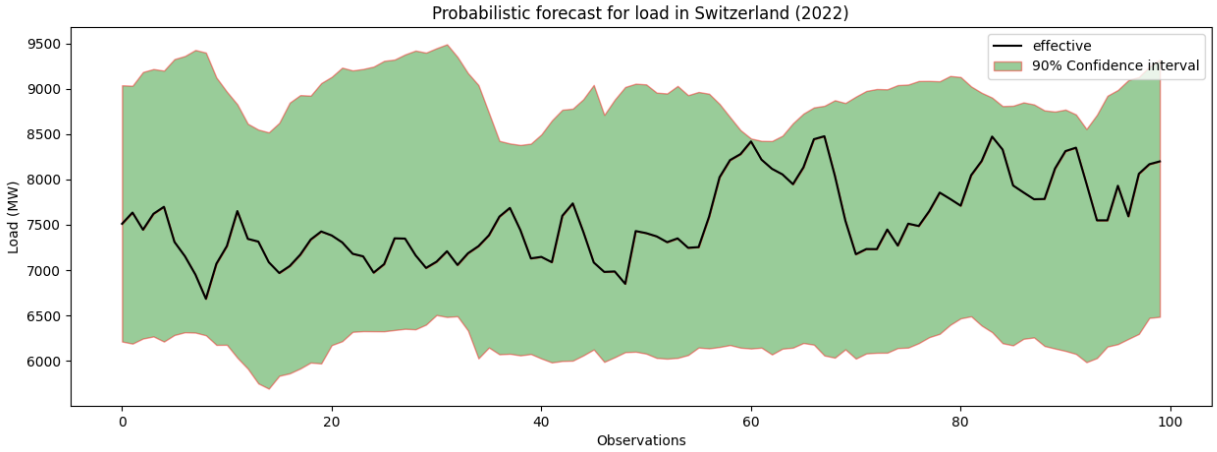


Figure 10.11: Load 90% confidence interval for Switzerland Energy charts data (2022) using KQR Laplacian. The black line is the observed path for the load. The 90% confidence interval bands are plotted in green. Lower and upper red lines denote the 95% and 5% quantile forecast respectively.

Table 10.6: Pinball loss for load in Germany (2022) model_2

Quantile	LQR	GBMQR	QF	KQR
0.1	0.04522	0.02580	0.02796	**0.02491**
0.2	0.08048	0.04326	0.04625	**0.04209**
0.3	0.10677	0.05570	0.05706	**0.05424**
0.4	0.12610	0.06182	0.06346	**0.06071**
0.5	0.13767	0.06214	0.06544	**0.06150**
0.6	0.14077	0.05752	0.06001	**0.05726**
0.7	0.13409	0.04893	0.05122	**0.04869**
0.8	0.11628	0.03687	0.03822	**0.03665**
0.9	0.08204	0.02114	0.02324	**0.02108**
CRPS	0.10771	0.04591	0.04810	**0.04524**

Cosine. Details concerning the hyperparameter selection for the different kernels can be found in sklearn.metrics.pairwise and sklearn.gaussian_process. The pool of data for this study has been create by combining SECURES-Met data (predictors) [30] and the load data from energy charts. SECURES-Met data consists of historical data up to the end of 2020 while from 2021 onward, the data consists of forecasts modeled by the the European Centre for Medium-Range Weather Forecasts (ECMWF). Therefore, in this experimental study, we used the whole data of 2020 as the training set and then tested our kernels on the 2021 data. Note that there are two kinds of predictions for data from 2021 onward, one for each of the emission scenarios RCP4.5 and

Table 10.7: Pinball loss for load in Switzerland (2022) model_2

Quantile	LQR	GBMQR	QF	KQR
0.1	0.04118	0.01881	0.01909	**0.01876**
0.2	0.07215	0.03004	0.03040	**0.02979**
0.3	0.09603	0.03749	0.03819	**0.03693**
0.4	0.11329	0.04230	0.04311	**0.04165**
0.5	0.12403	0.04404	0.04465	**0.04358**
0.6	0.12748	0.04282	0.04363	**0.04249**
0.7	0.12223	0.03789	0.03891	**0.03779**
0.8	0.10639	**0.02961**	0.03097	0.02972
0.9	0.07531	**0.01847**	0.01957	0.01862
CRPS	0.09756	0.03350	0.03428	**0.03326**

RCP8.5. In this study, we considered the more conservative RCP4.5 scenario data.

The predictors making up the datasets follow:

- Direct irradiation: direct normal irradiation
- Global radiation: mean global radiation
- Hydro reservoir: daily mean power from reservoir plants in MW
- Hydro river: daily mean power from run of river plants in MW
- Temperature: air temperature
- Wind potential: potential wind power production

Table 10.9 shows results for Germany, table 10.8 is the one for Switzerland and table 10.10 corresponds to Austria.

Table 10.8: Pinball loss kernel comparison for load in Switzerland (2021)

Quantile	Gaussian RBF	Laplacian	Matern 1.5	Matern 2.5	Linear	Periodic	Polynomial	Sigmoid	Cosine
0.1	0.018861	**0.018798**	0.019285	0.019510	0.019128	0.024474	0.021105	0.024299	0.018996
0.2	0.030924	**0.030430**	0.031038	0.031438	0.031223	0.040274	0.033892	0.039774	0.031008
0.3	0.039107	**0.038510**	0.039295	0.039751	0.039637	0.051023	0.042683	0.050242	0.039363
0.4	0.044285	**0.043718**	0.044763	0.045143	0.045130	0.057785	0.048232	0.056815	0.044896
0.5	0.046840	**0.046261**	0.047502	0.047957	0.047739	0.060671	0.051024	0.059697	0.047302
0.6	0.046138	**0.045582**	0.047239	0.047686	0.047416	0.059627	0.050456	0.058824	0.046834
0.7	0.041910	**0.041615**	0.043404	0.043964	0.043735	0.054688	0.046437	0.054037	0.042897
0.8	0.034032	**0.033852**	0.035332	0.035997	0.035518	0.044591	0.037918	0.044112	0.034557
0.9	**0.021817**	0.021923	0.022712	0.023082	0.022721	0.028070	0.024161	0.027920	0.022606
CRPS	0.035990	**0.035632**	0.036730	0.037170	0.036916	0.046800	0.039545	0.046191	0.036495

From our tables, it can be concluded that the absolute Laplacian kernel is the one fitting best to the characteristics of loda data. Additionally, we can conclude that our results are in line with the ones published in [44]. In that

Table 10.9: Pinball loss kernel comparison for load in Germany (2021)

Quantile	Gaussian RBF	Laplacian	Matern 1.5	Matern 2.5	Linear	Periodic	Polynomial	Sigmoid	Cosine
0.1	0.028057	**0.027483**	0.028236	0.028257	0.028261	0.028277	0.028276	0.028146	0.028004
0.2	0.047118	**0.046051**	0.047531	0.047544	0.047569	0.047595	0.047596	0.047353	0.047025
0.3	0.060366	**0.058739**	0.060892	0.060940	0.060962	0.060988	0.060984	0.060654	0.060209
0.4	0.068344	**0.066347**	0.068952	0.069022	0.069054	0.069104	0.069101	0.068634	0.068202
0.5	0.071450	**0.069118**	0.072188	0.072250	0.072285	0.072331	0.072330	0.071807	0.071359
0.6	0.068821	**0.067098**	0.069454	0.069526	0.069571	0.069615	0.069613	0.069157	0.068780
0.7	0.061150	**0.059972**	0.061382	0.061410	0.061427	0.061442	0.061442	0.061281	0.060990
0.8	0.047147	**0.046457**	0.047325	0.047331	0.047340	0.047348	0.047349	0.047247	0.047056
0.9	0.026646	0.026343	0.026749	0.026757	0.026763	0.026772	0.026772	**0.026282**	0.026628
CRPS	0.053233	**0.051956**	0.053634	0.053671	0.053693	0.053719	0.053718	0.053395	0.053139

Table 10.10: Pinball loss kernel comparison for load in Austria (2021)

Quantile	Gaussian RBF	Laplacian	Matern 1.5	Matern 2.5	Linear	Periodic	Polynomial	Sigmoid	Cosine
0.1	0.024309	**0.023882**	0.024316	0.024611	0.029367	0.029462	0.027319	0.029459	0.027816
0.2	0.041028	**0.040318**	0.040987	0.041487	0.048510	0.048716	0.046164	0.048714	0.045879
0.3	0.052597	**0.051643**	0.052548	0.052972	0.062242	0.062561	0.059133	0.062548	0.058454
0.4	0.058803	**0.057676**	0.058841	0.059078	0.070210	0.070577	0.067914	0.070567	0.066335
0.5	0.060846	**0.059441**	0.060912	0.061541	0.073399	0.073782	0.071616	0.073764	0.069574
0.6	0.059336	**0.057820**	0.059237	0.059930	0.070817	0.071157	0.070249	0.071134	0.066970
0.7	0.053937	**0.052402**	0.053949	0.054372	0.062453	0.062688	0.062970	0.062675	0.059419
0.8	0.043366	**0.042427**	0.043364	0.043510	0.049024	0.049150	0.049737	0.049147	0.046662
0.9	0.026727	**0.026227**	0.026645	0.026817	0.029516	0.029601	0.030097	0.029598	0.027986
CRPS	0.046772	**0.045760**	0.046755	0.047146	0.055060	0.055299	0.053911	0.055290	0.052122

study, similarly to what we did in this subsection, the authors carried out a comparitive study concerning the choice of kernels in KQR. Nevertheless notice, that the set of kernel we considered in our study is larger than the one presented in [44]. Their paper showed evidence of the superiority of the Gaussian RBF among their set of kernel functions. Analogously, we can see from tables 10.9, 10.8 and 10.10 that the Gaussian RBF is the second best in terms of pinball loss.

10.2.3 GEFCom2014

The follwing subsection compares our kernel quantile regression against the price and load track entries of the GEFCom2014 competiton. Error measure of the competition was the pinball loss, see section 4, averaged over the 99 quantiles, $q \in \{i/100\}_{i=1}^{n=99}$. Backed by the conclusion from the previous subsection we selected the absolute Laplacian and the Gaussian RBF as kernels in our application of KQR.

GEFCom2014 price track

In this setting, the predictors fed to our KQR models are

- Day
- Hour
- Forecasted total load

- Forecasted zonal load

Since target days vary all over the year, we trained a model for any month associated to them.

Our results for this track are reported in table 10.11. In this track, the top entries came from the teams: Tololo, Team Poland, GMD, and C3 Green Team; for a breakdown of each method's attributes, see [52, Table 8]. Finally, the Gaussian RBF probabilistic prediction for the 13th July 2013 zonal price at the 90% confidence interval is visualized in Figure 10.12.

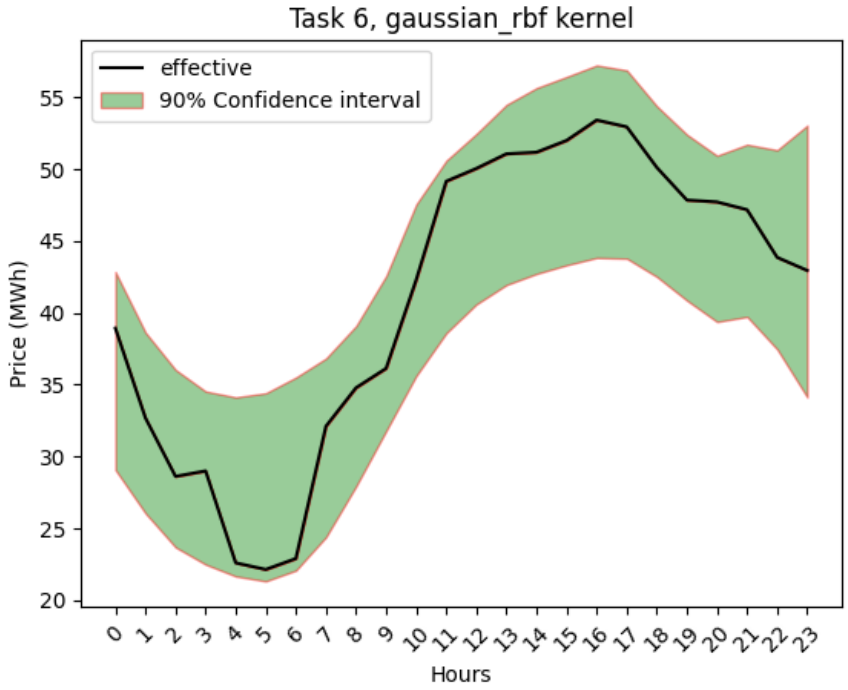


Figure 10.12: Price 90% confidence interval task 6. Electricity price probabilistic forecast for the 13^{th} July 2013. The black line is the observed path for the price. The 90% confidence interval bands are plotted in green. Lower and upper red lines denote the 95% and 5% quantile forecast respectively.

What can be concluded from table 10.11 is that KQR performed consistently among the top algorithms. Furthermore, It ranked first in two out of the thirteen tasks of the price track.

GEFCom2014 load track

The GEFCom2014 load track constitutes a good setting for analyzing the performance of KQR in the context of medium term load forecasting (MTLF). In this track we were faced with the challenge of predicting the load for the next

Table 10.11: Pinball loss GEFCom2014 price data

Team name\Task number	1	2	3	4	5	6	7	8	9	10	11	12
KQR laplacian	**1.3249**	2.9497	0.8868	7.3780	5.4314	8.2166	4.9091	1.5245	1.9601	2.7324	2.4733	17.5371
KQR gaussian rbf	2.0614	3.2247	**0.7050**	6.1407	6.1134	8.5969	2.9360	1.5731	1.8830	2.6419	2.4814	16.1921
Arkadiy Strelnikov	1,92899	3,46853	4,35373	7,16274	7,42959	6,32601	5,03883	2,02658	**0,66422**	2,35183	2,08614	7,23447
Benchmark - Price	4,02875	7,97208	5,70395	12,15104	38,33541	44,22979	18,22395	31,56729	42,94958	2,85583	3,20395	22,38333
C3 Green Team	1,85897	3,27786	1,2593	5,08886	6,87674	6,1505	4,42379	1,32639	1,25915	3,08224	1,55811	6,58123
E.S. Mangalova	2,05693	7,97208	0,87971	7,04219	11,0464	6,57565	6,02388	**0,69721**	2,77446	2,78586	2,14229	7,35955
EPSteam	4,02875	7,97208	5,70395	3,46235	29,5226	27,73186	17,0324	2,49354	1,25639	**1,79504**	1,63482	10,357
Florencio Gonzalez	4,02875	4,32597	4,35373	9,72491	10,51628	44,17405	4,00812	4,6924	7,36395	1,79865	2,1575	4,42008
GMD	3,7271	1,783	0,92191	5,08886	6,21331	**3,82599**	4,9342	1,47858	1,65933	2,06134	2,1235	6,84571
Manuel Oviedo de la Fuente	1,62605	3,27786	13,73529	6,68756	23,55608	10,01475	6,61296	2,2259	1,48028	3,65169	3,20396	4,7237
NimNid	3,44735	10,59679	2,76634	24,32776	23,55608	7,96082	3,33627	1,8191	1,60593	2,5711	2,32578	8,41167
San/Saini	2,47163	1,95533	0,84071	5,31786	9,61338	8,28547	3,0475	2,86903	3,60395	4,37704	1,82957	16,81896
Team Poland	1,97477	1,81898	1,19162	2,82318	7,55914	4,20773	2,59715	1,04693	1,24193	4,06012	**1,08458**	**3,06512**
Tololo	1,70734	**1,45173**	1,10384	**2,01694**	9,15596	4,6821	**1,59517**	0,75352	2,45935	2,9614	1,34614	3,55819
Xiaorong (Iris) Sun	1,45661	1,59212	0,98985	3,0349	4,73309	4,52459	3,63208	2,30481	0,90781	5,00935	1,18223	4,5302
Yanghai Cong	1,69815	5,84	4,83261	3,17957	10,21729	6,44717	5,55374	3,80812	4,38485	1,45195	1,51106	14,61798
dmlab	2,30162	1,92804	1,2593	2,58646	14,09957	7,5589	4,13365	0,80748	1,5149	3,71509	3,43097	10,22129
pat1	2,36615	1,98567	1,07248	2,79465	**4,23269**	4,70614	8,40506	1,25376	2,23991	3,67952	1,06139	6,27517

month without the availability of weather temperature forecasts. Therefore, the primary task was to first accurately predict the weather temperatures, and then model the load accordingly. Since there were no attributes available for humidity or wind speed, we chose to predict weather temperatures by aggregating historical temperature data across different dimensions such as day, month, and hour. Then we proceeded with building KQR models for the load; we chose the following predictors

- Day: the number of the day
- Hour
- Day of week: an ordinal categorical variable for the day of the week
- Is holiday: a binary variable for holidays, where holiday= 1 and working day= 0
- w avg: average of weather temperatures across all the 25 stations

We built 12 models, one for each month, with each task model trained on the historical data of the month associated with it.

Table 10.12 reports our results for the load track. The top teams for the load forecasting track were Tololo, Adada, Jingrui(Rain) Xie, OxMAth, E.S. Managalova, Ziel Florian, and Bidong Liu; for a breakdown of the attributes of each method, see [52, Table 6]. Finally, the 90% confidence interval forecast by our model equipped with the absolute Laplacian kernel for task number 9, that is the prediction for June, is reported in Figure 10.14. Similarly to the price track, we have that KQR was among the best algorithms also in the load forecasting track.

10.3 Conclusions

This section concludes this thesis work with some final remarks. This thesis was concerned with electricity forecsating, in particular we focused on the

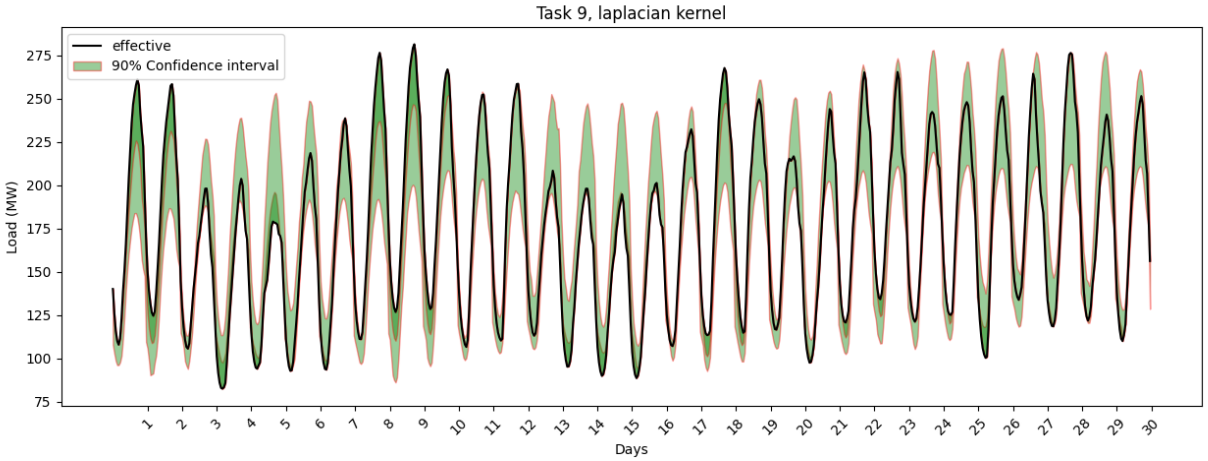


Figure 10.13: Load 90% confidence interval task 9 using KQR laplacian. The black line is the observed path for the load. The 90% confidence interval bands are plotted in green. Lower and upper red lines denote the 95% and 5% quantile forecast respectively.

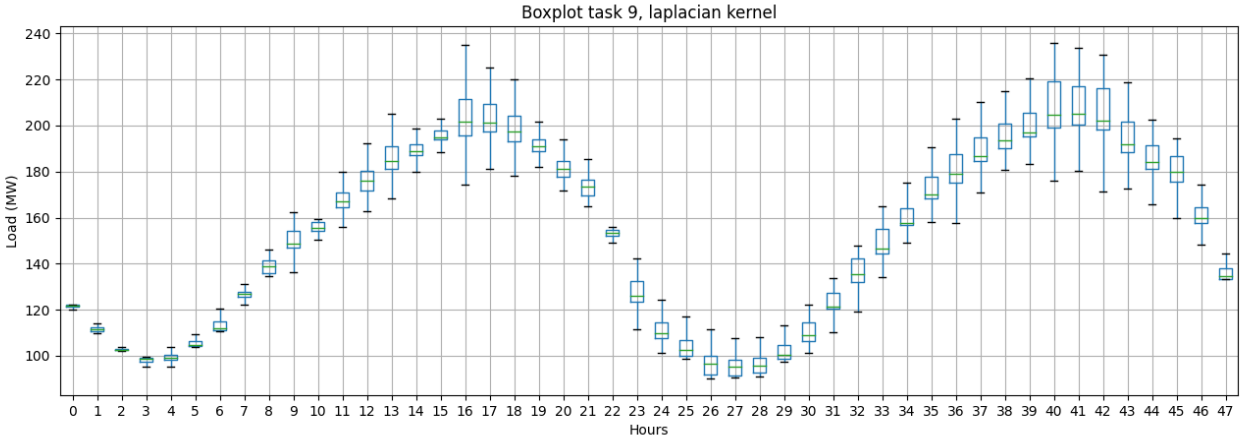


Figure 10.14: Boxplot for first two days of probabilistic forecast task 9 using KQR laplacian

probabilistic framework. Probabilistic forecasting is still a relatively new research area, lately research efforts are shifting towards this setting. This is due to the fact that probabilistic forecasts are more informative than point ones and better suited for the current and future electricity landscape; think for example about the higher uncertainty resulting from market liberalization or renewables integration requirements.

Among the contributions of this thesis work we can list

- Study of kernel methods in electricity forecasting both in the point and probabilistic framework.

Table 10.12: Pinball loss GEFCom2014 load data

Team name\Task number	1	2	3	4	5	6	7	8	9	10	11	12
KQR laplacian	12.0552	12.0573	8.8213	5.1721	6.9133	7.8080	11.5559	11.8250	6.8941	3.9680	7.4931	10.8869
KQR gaussian rbf	12.4665	12.1780	9.3326	5.1713	6.8703	6.7482	11.02473	11.93156	6.6019	4.3111	7.2207	10.8840
ACE	12.1330	14.4846	**7.3933**	4.8213	6.8048	7.0566	9.5921	11.6316	5.9859	5.0730	**5.6028**	8.9699
Adada	10.5093	10.0801	7.6238	4.7289	**5.3936**	6.6242	8.0144	11.1366	5.7779	3.6379	7.0096	8.9109
Alastair Muir	13.6118	16.6750	10.0759	7.7078	9.5386	8.3615	9.8152	13.1363	8.9715	5.4082	8.5881	17.5325
Andrew Landgraf	14.3650	10.1090	8.4180	6.2522	7.2248	11.1638	9.9403	11.0204	5.6920	6.1176	11.0677	13.3985
Benchmark - Load	18.7384	22.7585	13.2163	8.3626	10.9162	16.9937	13.4038	17.3151	13.8374	6.4237	10.9380	34.0685
Bidong Liu	16.4215	11.8655	9.3733	5.6212	7.7387	6.5536	9.1406	11.3485	6.5096	4.8031	6.9697	10.8935
C3 Green Team	18.7384	19.2208	7.9637	4.6370	6.4543	8.3799	10.5546	10.6609	5.8867	4.4866	5.9396	10.3917
Christopher Benfield	18.7384	11.1324	9.4377	4.9097	7.4184	19.7325	9.2215	11.4385	6.6395	4.5966	6.5002	10.8633
Dao Vu	33.3711	13.3340	10.4300	6.0815	9.0706	8.7098	11.2808	18.4869	7.4065	5.7414	10.3431	23.5659
E.S. Mangalova	18.7384	13.3340	7.8025	**4.4096**	6.6330	6.2306	10.1511	10.9294	6.2224	4.2382	6.5464	8.8080
Jingrui (Rain) Xie	11.8700	10.9250	8.4938	4.9611	7.4442	6.9921	9.0523	11.2600	5.4864	**3.3602**	5.9011	9.7316
Manuel Oviedo de la Fuente	12.5502	21.4591	10.1593	5.4647	9.3072	7.6102	8.8361	12.6340	12.3969	15.5225	6.5221	18.7754
OxMath	14.4091	**8.9136**	7.6059	4.4548	7.2944	7.4551	7.9527	**10.2444**	5.4551	4.2111	6.4054	9.5520
Rasmus Paivarinta	18.7384	11.7474	9.7230	7.7078	8.3074	7.1793	8.4391	10.9357	6.3855	4.3796	6.3794	11.7871
SAOR	18.7384	21.4591	10.4300	6.8512	8.6899	6.7653	12.1427	11.6261	6.7310	6.2850	5.9699	21.2711
Sniper	18.7384	10.3056	8.3493	5.6639	8.3074	6.9599	10.8848	13.4937	7.8877	5.3514	6.4770	11.1951
Tololo	**10.4369**	12.5232	8.2695	4.4220	5.8976	**6.1878**	**7.3182**	10.8032	**5.4469**	3.9613	6.3173	**8.4787**
Trevor Maynor	18.6934	19.4422	12.6071	5.5631	10.9298	13.9594	10.8657	14.4647	7.2859	6.2850	8.5881	17.4703
Xiaorong (Iris) Sun	13.8779	12.3979	10.3672	8.3626	8.1979	20.5364	11.2808	12.6316	6.6821	4.4907	6.2363	10.1624
dmlab	14.4091	14.0059	9.2128	5.9099	6.6675	6.9046	10.0130	11.0201	7.8508	3.8128	6.6474	26.6655
nikolina	18.7384	19.2208	10.6214	5.8257	9.5386	7.1234	10.0948	12.5375	5.8139	4.8060	8.4956	21.2711
pat1	19.3898	10.5097	9.1477	9.7023	7.0428	7.8794	8.8610	12.8179	5.5443	5.0730	6.5846	11.2142

- Provided Python open source implementation for kernel quantile regression compatible with the sklearn API. The code has been packaged and uploaded to the Python Package Index (PyPI) with the name kernel-quantile-regression. The github repo hosting the source code includes also the script implementing the experiments along with the cleaned datasets; this contribution is intended to forster reproducibility in research.
- Achieved superior performance of kernel quantile regression compared to standard quantile regressor algorithms.
- Created and made available datasets suitable for algorithms benchmarking considering data from the DACH region (Germany, Switzerland and Austria). The format of these data takes inspiration from the popular GEFCom competitions.
- Kernel quantile regression extensive comparison between kernel function types.
- Compared kernel quantile regression against state of the art probabilistic algorithms in the literature by means of the GEFCom2014 competition.

To finish, we proceed with summarizing the conclusions we drew from our experimental analysis. Kernel methods showed good performance both in the context of point and probabilistic forecasting. Specifically, kernel quantile regression resulted almost always in superior performance compared to standard quantile regressors. The absolute Laplacian kernel turned out to be the most suitable kernel according to the characteristic of electricity forecasting. Finally, we proved the validity of our kernel quantile regression

implementation. We achieved this by showing how it compared favourably to the top entries of the GEFCom2014 competition. In particular, up to now research work focused on the short term horizon capabilities of kernel quantile regression. In this thesis work, we were also able to show its suitability on the medium term horizon.

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
CDF	Cumulative density function
DDNN	Distributional neural network
ECMWF	European Centre for Medium-Range Weather Forecasts
EDA	Exploratory data analysis
EF	Electricity forecasting
ELF	Electricity load forecasting
EMD	Empirical mode decomposition
ENTSOE	European network of transmission system operators for electricity
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

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
ROR	Run of river
PACF	Partial autocorrelation function
PX	Power exchange
PI	Prediction interval
PF	Price Forecasting
PPF	Probabilistic Price Forecasting
PDF	Probability density function
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
$\oplus$	direct sum
d^*	dual optimum
$\mathbb{E}$	expectation
$\Phi(x)$	feature matrix
$\mathcal{F}$	feature space
$\varphi(x)$	feature vector

$g(v)$	dual function
$\odot$	Hadamard product
$\mathbb{I}$	Indicator function
K	kernel matrix
$k(x, x')$	kernel function
$\mathcal{L}$	lagrangian
ρ_q	pinball loss
p^*	primal optimum
relint	relative interior
$\mathcal{H}$	reproducing kernel hilbert space
$\otimes$	Tensor product
$\mathbb{1}$	vector of ones

Appendix A

Appendix

A.1 Feature map normalization

Lemma A.1 *Associated to any RKHS $\mathcal{H}$, there exists a feature vector $\varphi(x)$ such that $\|\varphi(x)\|_{\mathcal{H}} = R \quad \forall x \in \mathcal{X}$. That is the Hilbert space norm of the feature vector is equal to a constant R for all states in the set $\mathcal{X}$.*

Proof Taking the new feature vector as $\varphi^{new}(x) := \frac{\varphi(x)}{\|\varphi(x)\|_{\mathcal{H}}}$ we have

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

□

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

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**

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.
- 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.3: Mean absolute error Boston data

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

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

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.6: Mean absolute error Abalone data

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

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**

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 [72] and gradient boosting machine quantile regression [31] in terms of the pinball loss as well as in terms of the mean absolute error.

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.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 carrying out 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 $\hat{y}_{N+m}$ we use as inputs $y_1, y_2, \ldots, \hat{y}_N, \hat{y}_{N+1}, \ldots, \hat{y}_{N+m-1}$; where $\hat{y}_N, \hat{y}_{N+1}, \ldots, \hat{y}_{N+m-1}$ are one step ahead forecasts.

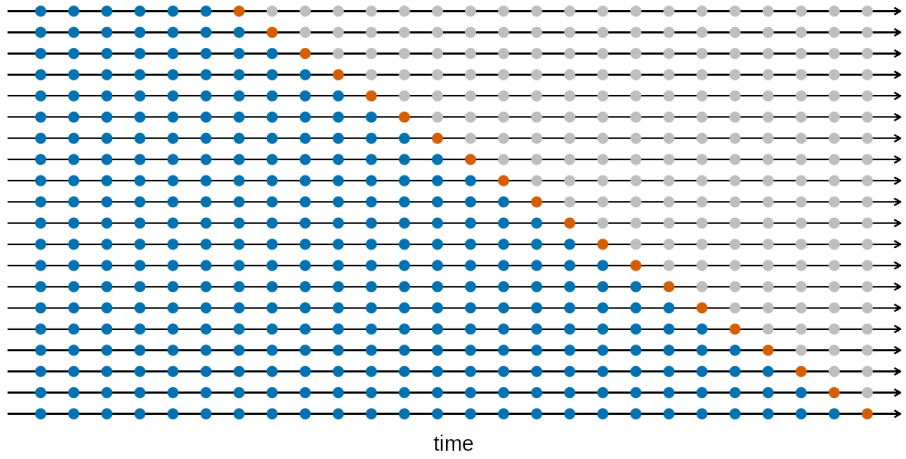


Figure A.1: Cross validation for one step ahead timeseries data [55]

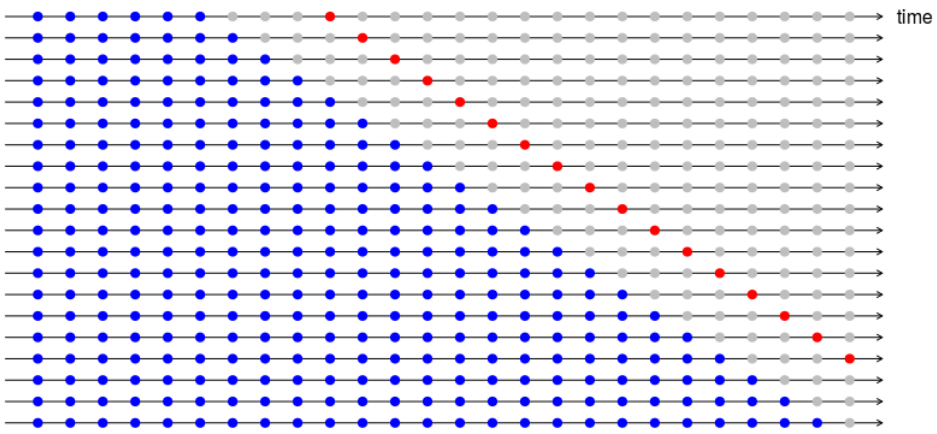


Figure A.2: Cross validation for m step ahead timeseries data [55]

A.4 Kernel methods best practices

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

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

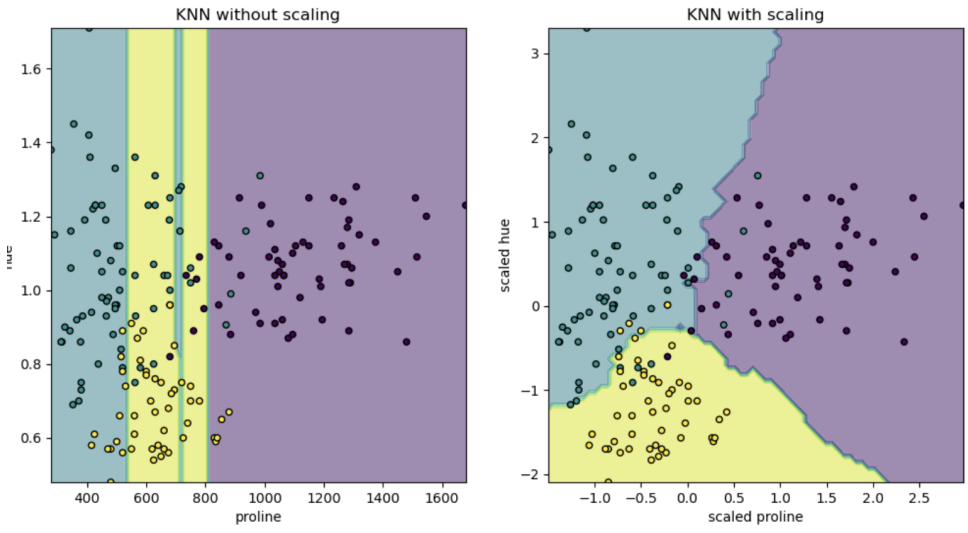


Figure A.3: importance of feature scaling

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 contribute 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 costs are reduced to $O(nm)$ and $O(nm^2)$ respectively. Nyström works by selecting $m < n$ points, callled representative points; it approximates K as

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

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.
- thesis/: folder containing tex files for thesis.
- beamer/: folder containing tex files for pptx thesis defence.
- experiments: folder containing train, test script for each of the experiments carried out in the thesis alongside with the binary files of the pickled models.
- experiments/src/kernel_quantile_regression/kqr.py: file implementing our custom kernel quantile regression.
- experiments/Data: cleaned data used in our experimental studies.
- experiment/plots/: folder storing plots resulting from experiments.
- experiment/eda/: scripts for exploratory data analysis.
- experiments/point/: scripts for experiments point framework.
- experiments/train_test/: scripts for training, testing and generating tables for experiments probabilistic framework.
- experiments/utils/: utility functions used in extracting, transforming and cleaning raw data.
- extradata/: folder containing data for experiments in section A.2.
- requirements.txt: pip freeze of python packages used.

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] Nikoleta Andreadou, Miguel Olariaga Guardiola, and Gianluca Fulli. Telecommunication technologies for smart grid projects with focus on smart metering applications. *Energies*, 9(5):375, 2016.

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

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

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

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

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

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

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

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

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

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

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

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

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

[24] 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.

[25] PAOLA Antonio DE, Nikoleta ANDREADOU, Evangelos KOTSAKIS, et al. Clean energy technology observatory: Smart grids in the european union-2023 status report on technology development trends, value chains and markets.

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

[27] 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.

[28] 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.

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

[30] Herbert Formayer, Imran Nadeem, David Leidinger, Philipp Maier, Franziska Schöniger, Demet Suna, Gustav Resch, Gerhard Totschnig, and Fabian Lehner. Secures-met: A european meteorological data set suitable for electricity modelling applications. *Scientific Data*, 10(1), 2023.

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

[32] 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.

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

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

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

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

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

[38] 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.

[39] 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.

[40] 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.

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

[42] 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.

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

[44] Yaoyao He, Rui Liu, Haiyan Li, Shuo Wang, and Xiaofen Lu. Short-term power load probability density forecasting method using kernel-based support vector quantile regression and copula theory. *Applied energy*, 185:254–266, 2017.

[45] 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.

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

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

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

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

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

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

[52] 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.

[53] 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.

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

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

[56] 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.

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

[58] Energy Institute. Statistical review of world energy, 2023.

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

[60] Reid A. Johnson. quantile-forest: A python package for quantile regression forests. *Journal of Open Source Software*, 9(93):5976, 2024.

[61] 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.

[62] 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.

[63] 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.

[64] 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.

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

[66] 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.

[67] 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.

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

[69] 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.

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

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

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

[73] 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.

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

[75] 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.

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

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

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

[79] 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.

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

[81] 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.

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

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

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

[85] 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.

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

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

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

[89] John Shawe-Taylor and Nello Cristianini. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

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

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

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

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

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

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

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

[97] 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.

[98] Lieven Vandenberghe. The cvxopt linear and quadratic cone program solvers, 2010.

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

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

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

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

[103] 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.

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

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

[106] 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.

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

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

[109] 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.

[110] 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.

[111] 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.

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

[113] 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.

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

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