Royal Holloway University of London

Centre for Reliable Machine Learning

Department of Computer Science

Machine Learning for Probabilistic Prediction

Valery Manokhin

Supervisors:

Prof. Vladimir Vovk

Prof. Alessio Sancetta

June 24, 2022

A Thesis submitted for the Degree of Doctor of Philosophy in Machine Learning

Declaration of Authorship

I hereby declare that this thesis and the work presented in it is entirely my own. Where I have consulted the work of others, this is always clearly stated.

Valery Manokhin June 24, 2022

"Prophesy is a good line of business, but it is full of risks."

Mark Twain in "Following the Equator."

Abstract

Prediction is the key objective of many machine learning applications. Accurate, reliable and robust predictions are essential for optimal and fair decisions by downstream components of artificial intelligence systems, especially in high-stakes applications, such as personalised health, self-driving cars, finance, new drug development, forecasting of election outcomes and pandemics.

Many modern machine learning algorithms output overconfident predictions, resulting in incorrect decisions and technology acceptance issues. Classical calibration methods rely on artificial assumptions and often result in overfitting, whilst modern calibration methods attempt to solve calibration issues by modifying components of black-box deep-learning systems. While this provides a partial solution, such modifications do not provide mathematical guarantees of predictions validity, are intrusive, complex, and costly to implement.

This thesis introduces novel methods for producing well-calibrated probabilistic predictions for machine learning classification and regression problems. A new method for multi-class classification problems is developed and compared to traditional calibration approaches. In the regression setting, the thesis develops novel methods for probabilistic regression to derive predictive distribution functions that are valid under a non-parametric IID assumption in terms of guaranteed coverage and contain more information when compared to classical conformal prediction methods whilst improving computational efficiency. Experimental studies of the methods introduced in this thesis demonstrate advantages with regard to state-of-the-art. The main advantage of split conformal predictive systems is their guaranteed validity, whilst cross-conformal predictive systems enjoy higher predictive efficiency and empirical validity in the absence of excess randomisation.

Acknowledgements

I would like to thank Prof. Vladimir Vovk and Prof. Alessio Sancetta for their invaluable support and guidance during the course of my PhD. My research has been supported by the Leverhulme Magna Carta Doctoral Centre.

Contents

List of Figures

List of Tables

List of Abbreviations

CP	**C**onformal **P**rediction
CPD	**C**onformal **P**redictive **D**istribution
CPS	**C**onformal **P**redictive **S**ystem
CCPS	**C**ross **C**onformal **P**redictive **S**ystem
CVAP	**C**ross **V**enn-**A**bers **P**redictors
DL	**D**eep **L**earning
DH	**D**empster-**H**ill predictive distribution
IVAP	**I**nductive **V**enn-**A**bers **P**redictors
IR	**I**sotonic **R**egression
KRRPM	**K**ernel **R**idge **R**egression **P**rediction **M**achine
LSPM	**L**east **S**quared **P**rediction **M**achine
LR	**L**ogistic **R**egression
NB	**N**aïve **B**ayes
NN	**N**eural **N**etwork
RPD	**R**andomized **P**redictive **D**istribution
RPS	**R**andomized **P**redictive **S**ystem
RRPM	**R**idge **R**egression **P**rediction **M**achine
SCPS	**S**plit **C**onformal **P**redictive **S**ystem
SVM	**S**upport **V**ector **M**achines
VA	**V**enn-**A**bers
VP	**V**enn **P**rediction

Chapter 1

Introduction

1.1 Machine learning

Machine learning is concerned with the development of data-driven algorithms that are able to make predictions about something in the world. Tom M. Mitchell defined a machine learning algorithm as follows: "A computer program is said to learn from experience **E** with respect to some class of tasks **T** and performance measure **P** if its performance at tasks in **T**, as measured by **P**, improves with experience **E**" [92].

Machine learning applications range from identifying handwritten digits in postal codes for efficient sorting of mail [78] to filtering of spam and predicting property prices. Recent progress in deep learning (a branch of machine learning based on artifical neural networks) resulted in human or near-human performance in computer vision, speech recognition and machine translation. This progress is the result of application of new algorithms for training advanced deep neural network architectures using abundant data and high-performance Graphics Processing Units (GPUs) that are able to run parallel computations at high speed. In computer vision, although convolutional neural networks (CNNs) have been around for decades, faster GPU-based implementations of backpropagation algorithm popularised by Hinton *et al.* [114] for training of deep neural networks allowed to achieve near-human performance in visual pattern recognition contests such as `ImageNet` competition [63].

1.1.1 Terminology of machine learning

Machine learning consists of several branches that include supervised learning, unsupervised learning and reinforcement learning. In supervised learning, an intelligent system is presented with examples of inputs (objects) and outputs (labels), and the goal is to learn a function that maps inputs to outputs. In unsupervised learning no labels are available — the goal of an unsupervised learning algorithm is to learn the structure of the data. The output of an unsupervised learning algorithm is either the assignment of objects into clusters, the density estimation or the projection of data into a lower dimensional space. Reinforcement learning is concerned with the behaviour of intelligent agents maximising their reward function by taking actions in an environment. This thesis is concerned with probabilistic prediction for supervised machine learning problems only.

In supervised machine learning problems, an algorithm is presented with a number of examples $\mathbf{z}$ from the training set of examples $\mathbf{Z}$. The examples contain objects $\mathbf{x}$ from the set $\mathbf{X}$ and labels $\mathbf{y}$ from the set $\mathbf{Y}$. The Cartesian product $\mathbf{Z} := \mathbf{X} \times \mathbf{Y}$ represents the set of all possible examples. Supervised machine learning consists of two main classes of problems: classification and regression. In a classification problem, the labels constitute a discrete set that can be as small as the set of numbers from 0 to 1 in a standard binary classification problem, several classes in a multi-class classification problem (as in `MNIST` dataset [78] containing examples of objects 0 to 9 representing hadwritten digits) or as large as tens of thousands classes (as in `ImageNet` [63] competition that contains over twenty thousand categories of images). In a machine learning regression problem, the labels are continuous — the objective of the machine learning regression algorithm is to learn the function that maps objects to continuous labels. Examples of machine learning regression tasks include making predictions about product demand, property prices, drug efficacy or toxicity and the lengths of taxi journeys.

1.1.2 History of supervised learning

Even though artificial intelligence and machine learning have seen significant developments since their start in the 1950s and 1960s, these fields use mathematical tools such as Bayes' theorem and Least Squares that date back to the 18th and 19th century. In 1950,

Alan Turing proposed a learning machine [130] that could become artificially intelligent via the learning process. In 1951, Marvin Minsky and Dean Edmonds built the first neural network machine, known as SNARC [115]. In 1957, Frank Rosenblatt caused a lot of excitement in the artificial intelligence community by building the perceptron [113]. It quickly turned out, however, that the perceptron was not able to recognise many patterns including a rather simple XOR function as demonstrated in 1969 by Minsky and Papert [91]. In 1967, Cover published a paper describing the Nearest neighbour algorithm [23]. The Nearest neighbour algorithm provided computers with the ability to perform basic pattern recognition tasks. In 1986, David Rumelhart, Geoff Hinton and Ronald J. Williams popularised the method of backpropagation for training of neural networks [114] (continuous backpropagation was initially derived by Henry J. Kelley [59] in the context of control theory). A number of more powerful supervised machine learning algorithms including random forest [9] and support vector machines (SVMs) [22] were published in the 1990s (the original linear support vector machines were invented and published by Vladimir Vapnik and Alexey Chervonenkis in the early 1960s [133]. In 1997, Sepp Hochreiter and Jürgen Schmidhuber published long short-term memory (LSTM, [52]) — an artificial recurrent neural network architecture that used feedback connections to solve the problem of vanishing gradient — the issue that impeded the effective use of the LSTM predecessor — traditional recurrent neural networks (RNNs).

1.2 Probabilistic prediction

Prediction is about forming forecasts about the future, whilst probabilistic prediction is about quantifying the uncertainty in prediction [40]. Forecast uncertainty quantification goes back to year 1906 when Australian meteorologist W.E. Cooke attempted to formulate weather forecasts in terms of probabilities, however the concept of odds was used in weather forecasts for more than 200 years [96]. In 1793, the British forecaster J. Dalton issued forecasts that included statements such as "the probability of a fair day to that of a wet one is as 10:1" and in France J.B. Lamarck produced forecasts using the term probable that implied a probability of an event greater than 0.5 [96].

Probabilistic forecasts are an important component of an optimal decision-making process [40] and are used to quantify uncertainty in such diverse domains as forecasting weather and election results, demographic projections and financial risk management. Central banks worldwide adopted probabilistic forecasting following the Bank of England's use of probabilistic models for interest rates for almost three decades [40].

More recently, estimating predictive uncertainty has become vitally important in many applications, such as healthcare, drug development and self-driving cars (see, e.g., [4], [105], [129], [60]).

More recently Stanford statisticians and Washington Post data scientists built more honest election prediction models using conformal prediction. According to Stanford University Prof. Emmanuel Candes: "Predictive models are used to make decisions that can have enormous consequences for people's lives. It's extremely important to understand the uncertainty about these predictions, so people don't make decisions based on false beliefs" [131].

The Stanford-Washington Post model is the "first real-world application of an existing statistical technique developed at Stanford by Candès, former postdoctoral scholar Yaniv Romano and former graduate student Evan Patterson. The technique is applicable to a variety of problems and, as in the Post's predication model, could help elevate the importance of honest uncertainty in forecasting. While the Post continues to fine-tune their model for future elections, Candès is applying the underlying technique elsewhere, including to data about COVID-19" (see [131], [112]).

Probabilistic prediction is rapidly gaining momentum in both academic research and practical applications — the recent `M4` [82] and `M5` forecasting competitions [83] required estimating prediction intervals (in addition to producing point forecasts) for a large number of time-series. Large tech companies such as Amazon [39] and Uber [71] are actively involved in probabilistic forecasting research and have incorporated probabilistic forecasting into the core of their machine learning systems. Probabilistic forecasting is used in production systems to provide large-scale retail demand predictions at Amazon and capacity forecasting at Amazon Web Services (AWS), whilst Uber uses probabilistic forecasting for marketplace forecasting of demand and resource allocation. Companies like

Walmart have embraced machine learning and predictive uncertainty estimation and co-organised the most recent M5 forecasting competition [83] that included two tracks to estimate both accuracy and uncertainty in the forecasts of future Walmart product sales.

The estimation of predictive uncertainty becomes particularly important when it is necessary to determine a case for manual intervention, especially in safety-critical applications such as healthcare [105] and self-driving cars [34] .

1.2.1 Terminology of probabilistic prediction

We will define the terminology of probabilistic prediction following Gneiting and Katzfuss [40]. The etymology of words "predict" and "forecast" is different. Whilst the word "prediction" comes from the Latin word "praedicatus" (relating to foretelling or prophesy), the meaning of the word "forecast" comes from the Middle English word for "forethought" or "prudence". When compared to prediction, forecast, therefore, implies more control over the planning for the future. However, we will use the terms "probabilistic prediction" and "probabilistic forecasting" interchangeably.

The general objective of probabilistic forecasting is to "maximise the sharpness of a predictive distribution subject to calibration" of the probabilistic forecast [40]. The sharpness is defined by the concentration of the probabilistic forecast, whilst calibration describes the relationship between forecast probabilities and the relative frequency of observed events. An example of a probabilistic prediction is a statement that "rain will occur with a probability of 70%" and if the relative frequency of rain that actually occurs during the days when rain is predicted to occur turns out to be 70%, we say that the probabilistic forecast is well calibrated (see, e.g., [26]). We introduce a number of formal definitions (for more details about proper scoring rules and terminology of probabilistic forecasting in general see Gneiting and Katzfuss [40]). The definitions below follow [40].

Definition 1.2.1 (Probability integral transform)**.** Given the observation Y, the probability integral transform (PIT) is the random variable $Z_F = F(Y)$, where F is the cumulative distribution function (CDF) for Y.

Definition 1.2.2 (Forecast calibration)**.** Let F and G be CDF-valued random quantities with PITs Z_F and Z_G:

- The forecast F is marginally calibrated if $\mathbb{E}(F(y)) = \mathsf{P}(Y \leq y)$ for all $y \in \mathbb{R}$.
- The forecast F is probabilistically calibrated if its PIT Z_F has a standard uniform distribution.

From the properties of standard uniform distribution, it follows that if F is probabilistically calibrated then $\mathrm{var}(Z_F) = \frac{1}{12}$ and F is well dispersed [40]. The most commonly used proper scoring loss is the logarithmic loss:

Definition 1.2.3 (Logarithmic loss)**.**

$$\mathrm{LL}(f, y) = -\log f(y)$$

In addition to the logarithmic loss, for machine learning classification problem, we will also use Brier loss (see [10], [6]).

Definition 1.2.4 (Brier loss)**.**

$$\mathrm{BL}(f, y) = \sum_{y' \in \mathbf{Y}} \left(1_{\{y'=y\}} - P\left(\{y'\}\right)\right)^2$$

Both the logarithmic loss and the Brier loss are examples of proper loss functions for classification problems. The continuous ranked probability score (CRPS) [40] is an example of proper loss function for the regression problems.

Definition 1.2.5 (Continuous ranked probability score)**.**

$$\mathrm{CRPS}(F, y) = \int_{-\infty}^{\infty} (F(x) - 1\{y \leq x\})^2 \mathrm{d}x$$

The CRPS generalizes the absolute error and in the case of point forecast is the same as the absolute error. The CRPS can be used to compare probabilistic forecasts [40].

1.2.2 Conformal prediction

In a supervised machine learning problem, the objective is to predict the label based on the features of an object. However, given a point prediction $\hat{y}$, how can one evaluate prediction quality (i.e., how likely is it that $\hat{y} = y$)? In practice, such evaluation can be done on

the basis of how such predictions performed in comparison with the actual results based on past experience. The central idea of conformal prediction is to use such past experience to determine precise levels of confidence in the individual predictions [120]. Conformal prediction is the statistical learning framework that is able to provide measures of confidence for each individual prediction, thus effectively "hedging" the predictions [137].

The prediction region Γ^{ε} is constructed using the significance level $\epsilon \in (0, 1)$ with a specified degree of confidence. To construct prediction region, the conformal predictor estimates degree of similarity of a new example to the examples in the training set using specified conformity measure, this conformity measure is then used to produce prediction region Γ^{ε} for each significance level ε [120].

The ideas of conformal prediction go back to the 1960s as Kolmogorov's thinking about the meaning of randomness [2] and later to the 1970s and the 1980s as on-line compression models [120] that were studied using the ideas from the statistical mechanics [72]. In both fields of study the models use summary statistics that contain all the useful information to predict future examples [120]. When a new example is observed, summary statistics is updated and "the probabilistic content of the structure is expressed by Markov kernels that give probabilities for summarized examples conditional on the summaries" [120].

Specifically, given a set of n examples $x_1, y_1, \ldots x_{n-1}, y_{n-1}, x_n$ and a significance level ϵ, the conformal predictor outputs the prediction set

$$\Gamma^{\epsilon}\left(x_1, y_1, \ldots x_{n-1}, y_{n-1}, x_n\right).$$

The prediction set Γ^{ϵ} must satisfy the property $\Gamma^{\epsilon_1}\left(x_1, y_1, \ldots x_{n-1}, y_{n-1}, x_n\right) \subseteq \Gamma^{\epsilon_2}\left(x_1, y_1, \ldots x_{n-1}, y_{n-1}, x_n\right)$ whenever $\epsilon_1 \geqslant \epsilon_2$, i.e., the larger subset of possible labels results in a more confident prediction.

1.3 Main contributions

The goal of the research described in this thesis is to develop new methods for probabilistic prediction for machine learning classification and regression tasks.

- Chapter 2 describes calibration methods, including recently developed computationally efficient algorithms IVAP (inductive Venn-Abers predictor) and CVAP (cross Venn-Abers predictor). The remainder of this chapter will outline new methods for probabilistic classification and regression. These new methods further develop current state-of-the art in probabilistic prediction.

- Chapter 3 describes a novel method of application of IVAPs and CVAPs (probabilistic algorithms for binary classification) to multi-class classification and studies its performance empirically in comparison with traditional calibration methods such as Platt's scaling and isotonic regression. This work was presented at "The 6th Symposium on Conformal and Probabilistic Prediction with Applications" (COPA 2017, Stockholm). The conference paper "Multi-class probabilistic classification using inductive and cross Venn–Abers predictors" by Manokhin [85] introducing a novel method of application of IVAPs and CVAPs to the problem of multi-class classification and studying the performance of the new algorithm in comparison with traditional calibration methods such as Platt's scaling and isotonic regression has been published in the "Proceedings of Machine Learning Research." The code has been released as open source [86].

- Chapter 4 focuses on the study of probabilistic regression and derives predictive distribution functions that are valid under a non-parametric assumption. A new prediction algorithm, called the Least Squared Prediction Machine (LSPM), is introduced to generalize the classical Dempster-Hill predictive distribution to regression problems. This work was presented at "The 6th Symposium on Conformal and Probabilistic Prediction with Applications" (COPA 2017, Stockholm). A conference version of the paper [145] has been published in the "Proceedings of Machine Learning Research.". The journal paper "Nonparametric predictive distributions based on conformal prediction" by Vovk, Shen, Manokhin, Xie [146] describing a novel method of deriving predictive distributions that are valid under a nonparametric assumption has been published by "Machine Learning" (Springer).

- Chapter 5 studies theoretically and empirically conformal predictive distributions

with kernel methods. Kernel Ridge Regression Prediction Machine (KRRPM) is introduced and its properties are studied. The main advantage of the KRRPM is its flexibility: for a suitable kernel, it gets the location and shape of the predictive distribution right in the case of homoscedasticity. This work resulted in publication as a chapter "Conformal predictive distributions with kernels" in the book "Braverman Readings in Machine Learning. Key Ideas from Inception to Current State" (Springer) by Vovk, Nouretdinov, Manokhin and Gammerman [139] with preprint version published on arXiv [139].

- Chapter 6 introduces two novel computationally efficient versions of conformal predictive distributions: split and cross conformal predictive distributions, and discusses their advantages and limitations. This work has been presented at "The 7th Symposium on Conformal and Probabilistic Prediction with Applications" (COPA 2018, Maastricht). The conference paper "Cross-conformal predictive distributions" [140] has been published in the "Proceedings of Machine Learning Research." The journal version "Computationally efficient versions of conformal predictive distributions" by Vovk, Petej, Nouretdinov, Manokhin and Gammerman has been published in "Neurocomputing" [143] (pre-print version published on arXiv [142]). This journal version of the conference paper [140] contains more detailed comparison of SCPS and CCPS, a detailed discussion of Venn-Abers predictive systems, and the analysis of universality of various predictive systems. An important finding in the paper is that SCPS and CCPS are universal, whereas Venn–Abers predictive systems are not (Venn-Abers predictive systems introduced in Nouretdinov *et al.*[101] are not classical Venn predictors in a classification setting, but are rather the application of Venn Predictors to any regression method to obtain calibrated predictive distributions).

- Created "Awesome Conformal Prediction" [84] – the most comprehensive professionally curated resource on conformal prediction. "Awesome Conformal Prediction" has been cited in one of the most prominent books on Machine Learning "Probabilistic Machine Learning: An Introduction" [97] by the leading research scientist at Google Kevin Murphy (over 80K Google Scholar citations). The resource has

proven very popular with both the academic and the tech community and accumulated hundreeds of stars on Github in just a few months raising awareness about conformal prediction globally. "Awesome Conformal Prediction" is the official conformal prediction for the ICML2022 "Workshop on Distribution-Free Uncertainty Quantification" and has been featured in the most popular tutorial on conformal prediction "A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification" [1].

Chapter 2

Literature review

This chapter introduces probabilistic machine learning and describes some of the calibration methods, both classical and modern, including recently developed computationally efficient algorithms IVAP (inductive Venn-Abers predictor) and CVAP (cross Venn-Abers predictor). The aim is to summarise the existing approaches, both parametric and non-parametric, which will serve as the basis of comparison for the methods developed in Chapter 3.

2.1 Introduction

The objective of a supervised machine learning classification task is to predict labels for new and unseen objects from the test set. A desirable property of a machine learning algorithm is its ability not only to classify a new object into the correct class, but also to generate accurate, reliable and robust class probabilities. By assigning class probabilities to the new object, the machine learning algorithm is able to effectively "hedge" its predictions by complementing class prediction with measures of prediction uncertainty / prediction quality. This becomes especially critical in many real-world applications such as healthcare [105] and self-driving cars [94], where obtaining accurate class probabilities significantly affects critical elements of a decision-making process, such as whether to stop a self-driving car when the algorithm is unsure about its prediction of whether there is a pedestrian or obstacle on the road or not [94].

Predictive classification models are most commonly evaluated by looking at their ability to separate objects into correct classes, this is often evaluated using receiver operating

characteristic (ROC-AUC) that are able to provide more comprehensive measure of performance when compared to accuracy and error rate [36]. Calibration is another important criteria of performance that has received less attention in both academic literature and practice, but is becoming increasingly important due to the need to provide predictions in an accurate, fair and transparent manner. A model is said to be well calibrated if its output reflects true probabilities of the events — for every 100 objects assigned a probability score of s, close to s will belong to class with label 1 [14].

It is often assumed, that traditional neural networks are well-calibrated (see, e.g., Caruana and Mizil [17], however, there is a growing body of research demonstrating that not only modern deep neural networks are not well-calibrated (see, e.g., [44], [89], [94], [105]), but also that traditional neural networks are often mis-calibrated as well [57]. As shown in [44], whilst very large and deep computer vision models such as ResNet [45] are much more accurate than previous classical architectures such as LeNet [77], modern deep neural networks become significantly miscalibrated even as classification accuracy improves. Guo *et al.* [44] attribute this behaviour to recent advances in deep learning such as model capacity, batch normalization, weight decay that all have strong negative effects on network calibration.

Messoudi *et al.* [89] considered Bayesian approach to estimation of confidence of the predictions of machine learning models, however concluded that such methods have major limitations. In order to accurately estimate posterior distributions, Bayesian methods need correct prior distributions, however in practice such prior distributions are often chosen arbitrarily or simply not available. If an incorrect prior is chosen, posterior probability will not result in a valid forecast (see, e.g, [88]) and predictive interval will not contain actual labels ("ground truth") with specified probability (e.g., 95% predictive interval will not contain 95% of the actual labels), resulting in predictive intervals being too wide or, most likely, too narrow (overconfident predictions) — as has been demonstrated in multiple forecasting competitions (see, e.g, [43]). In addition, in novel situations such as new drug development it is not possible even for an expert to arrive at correct priors — a situation well-known to regulators who discourage using subjective opinions not backed by actual data, especially in regulated industries such as pharmaceuticals, financial services

and healthcare.

Melluish *et al.* [88] demonstrated that Bayesian methods give misleading predictions and incorrect prediction intervals when the assumptions of the Bayesian model are incorrect. The authors concluded that "when correct priors are known, Bayesian algorithms give optimal decisions, and accurate confidence values for predictions can be obtained. However, if the prior is incorrect, these confidence values have no theoretical base — even though the algorithm's predictive performance may be good. Bayesian frameworks are often applied to these algorithms in order to obtain such values, however they can rely on unjustified priors."

Mukhoti *et al.* [94] analysed miscalibration of deep neural networks and concluded that many current multi-class classification networks are poorly calibrated, resulting in the mismtach between the model's confidence and accuracy. The authors concluded that the high capacity of these networks leaves them vulnerable to overfitting on the negative log-likelihood (NLL) loss they conventionally use during the training and replaced the cross-entropy with focal-loss to prevent the model from becoming overconfident.

Outside of the deep learning domain — in forecasting — the recent "M4" forecasting competition [82] is a typical example of the majority of submitted models outputting uncalibrated predictions. According to [43] "prediction intervals became overconfident and increasingly so over the long horizons." The author concluded [43] that "many of the models submitted performed poorly (except, notably, for the top two or three submissions), in that they often performed worse than the benchmarks provided by the competition organisers. A possible explanation for this poor performance might be a mix of overconfidence and overfitting."

Messoudi *et al.* [89] used conformal prediction to develop more reliabe and cautious deep learning models and concluded that "the conformal model not only adds reliability and robustness to the deep model by detecting ambiguous examples, but also keeps or even improves the performance of the basic deep model when it predicts only one class. We also illustrated the ability of conformal prediction to handle noisy and outlier examples for all three types of data. These experiments show that the conformal method can give more robustness and reliability to predictions on several types of data and basic deep

architectures."

The application of conformal prediction to neural networks is not a new idea, Papadopoulos *et al.* [103] used Venn-Abers predictors to produce well-calibrated probabilistic predictions using neural networks in multiclass setting, by combining Venn predictors [137] combined with neural networks applied to binary classifications problems [104].

Perreira *et al.* [105] evaluated various uncertainty quantification methods (described in more detail in 2.2) to produce individual probabilistic predictions for clinical decisions for patients with Alzheimer decease. Both Platt's scaling 2.2.2 and isotonic regression 2.2.3 were evaluated and compared to Conformal Prediction and Venn-Abers prediction. The study concluded that Platt's scaling is only adequate to calibrate the output from the SVM, the finding confirmed in several other papers covered in more detail in 2.2 . Whilst isotonic regression performed better than Platt's scaling, combining isotonic regression with decision trees produced poor results. Both Conformal Prediction and Venn-Abers prediction produced superior results by reducing both false negative and false positives. The study concluded that "Conformal Prediciton and Venn-Abers were the preferable methods to complement predictions with measures of uncertainty and conformal prediction based methods produced predictions with a low error rate for high credibility thresholds."

To summarise, many machine learning algorithms do not produce class membership probabilities and the ones that do often generate classification scores that do not correspond to class probabilities. In such cases the scores need to be transformed into well-calibrated probabilities that can be combined with utility scores for effective decision-making.

2.2 Calibration methods

To transform class scores into class probabilities, several methods have been invented. Such methods include both parametric and non-parametric methods and are summarised below.

In a binary classification problem, the objective of the calibration process is to convert class scores from an underlying machine learning classifier $s = f(x)$ into calibrated probabilities using the calibration map μ. A perfectly calibrated classifier $s = f(x)$ is defined as follows (see, e.g., [44]):

Definition 2.2.1 (Perfectly calibrated classifier)**.** A scoring classifier is perfectly calibrated if: $\mathbb{P}(\hat{Y} = Y \mid \hat{P} = p) = p, \quad \forall p \in [0, 1]$, where $\hat{Y}$ is class prediction and $\hat{P}$ is confidence associated with class prediction [44].

Intuitively this means that we would like for model output to represent true probability of classes.

2.2.1 Histogram binning

Histogram binning is an approach proposed by Zadrozny and Elkan in [152]. In this approach, the scores are sorted by value and the set of scores is divided into a number of bins. The value of each bin is then selected by solving an optimisation problem in which squared error (e.g., the Brier loss) is minimized by replacing each object's score with with the value of the bin assigned to that object. The disadvantages of the histogram binning method include arbitrary choice of the number of bins and the fixed size of the bins [99]. In addition, the mapping function is not a continuous function and there is no guarantee that the mapping function is non-decreasing function of the object scores.

2.2.2 Platt's method

Platt's method, also known as "Platt's scaling" or "Platt calibration" was designed to learn the function mapping classification scores produces by the SVM classifier to probability distribution over classes. The method (described by Platt in [109]) was invented to address specific disadvantage of a very popular classification algorithm — support vector machines [22] — that produced accurate assignment of the objects to classes, but required further calibration of the class scores to align them with true class probabilities. Support vector machines (SVM) do not produce class probabilities direct, but instead output uncalibrated classification scores that reflect distance to the maximum-margin hyperplane.

Platt observed that the relationship between classification scores produced by the SVM and empirical probabilities tended to be of the sigmoid nature that can be desribed by the parametrized sigmoid function:

$$P(y = 1|x) = \frac{1}{1 + \exp(Ax + B)}. \tag{2.1}$$

This is equivalent to the assumption that classification scores from the SVM are proportional to the log-odds based on the probability of an object belonging to the positive class [109]:

$$g(x) = \log \frac{P(y = 1 \mid x)}{P(y = 0|x)}. \tag{2.2}$$

The probability of the positive class can then be expressed as:

$$P(y = 1|x) = \frac{1}{1 + e^{-g(x)}}. \tag{2.3}$$

Platt's suggestion was to estimate parameters A and B by fitting maximum likelihood estimator on a new training set (x_i, t_i) [109] constructed as follows:

$$t_i = \frac{y_i + 1}{2}.$$

The parameters A and B can then be estimated by minimizing negative log-likelihood (LL) on the new training set:

$$LL = -\sum_i t_i \log(p_i) + (1 - t_i) \log(1 - p_i)$$

Where p_i is probability of positive class described by (2.3). As a regularized alternative method designed to address potential overfitring by the sigmoid function, Platt [109] suggested to modify t_i as follows:

$$t_+ = \frac{N_+ + 1}{N_+ + 2}$$

$$t_- = \frac{1}{N_- + 2}.$$

Platt [109] demonstrated that that this method produces estimates of probabilities that are as accurate as in the earlier method suggested by Vapnik [134] that has mapped the output of the SVM to class probabilities on the basis of feature space decomposition.

Zhang [154] proposed piecewise logistic regression as an alternative to Platt's [109] method. The method extends logistic regression by replacing linear function in Platt's sigmoid (2.1) with a piecewise linear function. Such an approach results in the log-odds (2.2) also becoming piecewise linear, thus providing additional flexibility in comparison with the original linear function $Ax + b$ used by Platt. The use of piecewise linear function allows for better separation of objects into the three classes — two classes where class ownership is clear and the borderline cluster of objects where there is less certainty as to which class an object might belong. The authors showed [154] that the piecewise logistic regression performed significantly better than Platt's method across a number of text categorization datasets using logistic loss as an error metric.

Whilst Platt's method is often used in practice, its mapping function is tailored to the very specific case of SVM as underlying machine learning classifier and as shown in [105] is unsuitable to calibrate output from classifiers other than SVM. As shown in more recent research [65] (whilst not explicitely stated in Platt's original paper [109]), Platt's scaler's assumptions are very restrictive and are not reasonable for many probabilistic classifiers that output scores in the range [0,1]. In such situations the use of Platt's scaler results in model mismatch and can result in classifiers becoming less calibrated even in comparison with class scores produced by an underlying machine learning classifier [65] (this in turn is due to the fact that Platt's mapping does not contain the identity function).

2.2.3 Isotonic regression

To address some of the above mentioned issues with Platt's scaler, Zadrozny and Elkan [153] proposed to use a non-parametric calibration method based on the isotonic regression [7] and applied it to a range of datasets using Naïve Bayes and support vector machines as underlying classifiers. Zadrozny and Elkan [153] demonstrate that for the Naïve Bayes the sigmoid assumption used by Platt does not generally fit the shape of class scores produced by classifiers other than SVM.

To correct this issue, Zadrozny and Elkan proposed a method that is an intermediate approach between Platt's sigmoid and histogram binning. Assuming that the underlying classifier ranks objects correctly (as will be shown in this section, this is a rather limiting assumption which results in the major drawback of the isotonic regression as a calibration method), the function mapping class scores to probabilities should be non-decreasing. In statistics such an approach is known as isotonic regression [7]. For fitting isotonic regression, Zadrozny and Elkan use pair-adjacent-violators (PAV) algorithm [5], this non-parametric algorithm solves fitting problem in linear time $O(N)$ by computing stepwise-constant isotonic function using general mean-squared error as an error metric [153].

Isotonic regression allocates class scores into bins using the convex hull of the ROC curve [65]. The slope of each segment is related to the empirical likelihood ratio. Calibrated posterior probabilities can thus be derived by using computed slopes of each segment. The monotonic mapping of isotonic regression relies on the underlying classification algorithm (whether machine learning or statistical) producing correct ranking of classification scores — this in turn is equivalent to having ROC AUC score of 1 which is almost never possible to obtain on a test set using any underlying classifier unless one uses a toy dataset.

2.2.4 Smooth isotonic regression

The mapping function obtained by applying isotonic regression is non-continuous, similar to the case of histogram binning. Jiang *et al.* [56] developed a smoother computationally effective method by interpolating isotonic regression between representative values using monotonic splines called "Piecewise Cubic Hermite Interpolating Polynomial" (PCHIP). In a range of experiments involving both synthetic and real datasets, the calibration results from using smooth isotonic regression were compared with results obtained using Platt's scaler and isotonic regression when applied to the output from the Logistic Regression. According to the experiments, the smoothing method significantly improved calibration of the Logistics Regression classification scores compared to the isotonic regression [56].

2.2.5 Nested Dichotomies

Nested Dichotomies (NDs) [76] are a method of transforming a multiclass classification problem into a series of binary classification problems. In this method the set of classes is recursively split into subsets using a tree structure, at each node a binary classification model is used to separate objects into two subsets. Leathart *et al.* [76] show that nested dichotomies exhibit poor calibration, even when the underlying classifiers are well calibrated. The problem of poor calibration is exarcebated in cases when the underlying binary classifiers are not well calibrated. The authors demonstrate [76] that both accuracy and calibration (as measured by the Log Loss) can be improved by calibrating both the underlying binary classification models and the full ND structure. By performing experiments on a range of datasets using Naïve Bayes and boosted trees as base classifiers, the authors demonstrate that predictive performance of NDs can be substantially improved by applying calibration techniques [76].

2.2.6 Beta calibration

Beta calibration is a method developed in Kull *et al.* [65]. The authors demonstrate that parametric assumptions in Platt's scaler are equivalent to assuming that the scores outputted by the underlying classifier are normally distributed and have the same variance (σ^2) for each class. In addition, experiments in [65] show that using Platt's scaling to calibrate output of many classifiers including AdaBoost and Naïve Bayes can result in probability estimates that are worse than original scores when score distributions have heavy tails. Using Platt's scaling (also known as logistic calibration) as a mapping function for well calibrated cases can result in distortion as logistic function does not include the identity function. Whist Platt's scaling is a parametric method that requires less data than isotonic regression (as non-parametric method isotonic regression can overfit on small datasets), logistic calibration can produce bad results when the data model is misspecified. The authors demonstrate [65] that "model mismatch is a real danger for a range of widely used machine learning models and can make classifiers less calibrated."

The normality and homoscedasticity assumptions in Platt's scaler are not reasonable

for many probabilistic classifiers that output scores in the range [0,1] (whilst normal distributions have infinite support). To address this, the authors propose to use beta distribution that allows for modeling of rich class of mapping functions.

In the experiments Kull *et al.* [65] apply logistic (Platt's) scaler, isotonic regression and beta calibration to the class scores produced by a range of underlying classifiers, including Naïve Bayes, Adaboost, logistic regression, support vector machines, random forest and multi-layer perceptron on 41 datasets from the UCI data repository [33]. The results showed that Beta-calibration performed well on a range of dataset sizes and a variety of classifiers with different characteristics. In terms of both the Log loss and the Brier loss, beta calibration was often the best calibrator when compared to uncalibrated scores, calibrated scores produced by Platt's scaler and isotonic regresison. It was also never the worst calibrator, making it a good choice when compared to isotonic regression on smaller datasets and in general often outperforming Platt's scaler.

2.2.7 Scaling-binning

Scaling-binning calibrator [66] is a combination approach that uses both histogram binning and parametric methods. As traditional neural networks [57] and most modern deep learning architectures do not output calibrated probabilities out of the box and are miscalibrated (see, e.g,, [44], [64], [94]), researchers and machine learning practitioners use easy scaling approaches such as Platt's scaling [109], isotonic regression [152] or temperature scaling [44]. As shown in [66] such approaches are in practice less calibrated than reported in the literature, in addition such approaches "have fundamental limitation as their true calibration error can not be measured using a finite number of bins" [66].

In the scaling-binning approach a simple function g is first fit to the scores s outputted by the underlying machine learning classifier. The input space is then binned with equal number of inputs ending up in the same bin. As a final step, the values of g are averaged across each bin. The resulting function can be computed more efficiently than in binning methods and as shown in [66] has lower variance than labels y.

In multiclass calibration experiments run on `CIFAR-10` [62] and `Image-Net` [63] computer vision datasets, scaling-binning calibrator achieved a lower calibration error (35%

lower on CIFAR-10 and 5x lower on Image-Net) than histogram binning [66].

2.2.8 Probability Calibration Trees

Probability calibration trees (PCT) [75] is a local probability calibration approach that identifies regions in the input space to learn local calibration models in order to improve performance.

The most commonly used methods such as Platt's scaling and isotonic regression calibrate classification scores globally across the object space. Leathart *et al.* [75] hypothesise that this approach can be improved using a more fine-grained calibration model and propose a novel approach based on logistic model trees [69]. The PCT approach follows [69] by combining decision trees with logistic regression. Such an approach results in an adaptive model where for simple datasets a linear model might give the best performance, whilst for more complicated datasets require a complex tree to be built [75].

In the experiments run on 32 UCI datasets [33] probability calibration trees either performed in line with or outpeformed both Platt's scaling and isotonic regressions when Naïve-Bayes was used as the base classifier. For other base classifiers such as boosted stumps, boosted decision trees and SVM, probability calibrations trees performed better than Platt's scaling and better than isonotic regression for all but 3 datasets.

2.2.9 Bayesian Binning into Quantiles

Naeni *et al.* [100] developed a non-parametric calibration method called Bayesian Binning into Quantiles (BBQ) using an ensemble of near isotonic regressions with Bayesian information criterion (BIC). BBQ extends histogram binning by combining outputs of multiple binning models. The models differ in the number of bins, the objects are allocated into bins based on equal frequency. Using Expected Calibration Error and Maximum Calibration Error defined below, the authors evaluate performace of the BBQ in comparison with histogram binning, Platt's scaling and isotonic regression using support vector machines as the base classifier.

Definition 2.2.2 (Expected Calibration Error)**.** Expected Calibration Error is computed as:

$$ECE = \sum_{i=1}^{K} P(i) \cdot |o_i - e_i| \,,$$

where o_i is the fraction of positive labels in bin i, e_i is the mean of probabilities (either before or after calibration) in the same bin i, and $P(i)$ is the empirical frequency of all objects assigned to bin i.

Definition 2.2.3 (Maximum Calibration Error)**.** Maximum Calibration Error is computed as:

$$MCE = \max_{i=1}^{K} (|o_i - e_i|) \,.$$

In the experiments on synthetic and 30 real datasets from [33], BBQ performed competitively in terms of allocating objects to the correct class and often performed better than Platt's scaling and isotonic regression in terms of calibration using a range of base classifiers including logistic regression, support vector machines and Naïve-Bayes [100].

2.2.10 Ensemble of Near Isotonic Regression (ENIR)

Naeni and Cooper [98] extended the approach in Bayesian Binning into Quantiles (BBQ) by addressing the monotonicity assumption of the isotonic regresssion. Whilst isotonic regression assumes that ranking of classification scores produced by the underlying classifier is correct, in practice this is equivalent to assuming that the area under the ROC curve (AUC) is equal to 1, which rarely happens in practice [99]. In comparison BBQ does not make such assumption, whilst the further key idea in ENIR is to balance the approache taken in isotonic regression to that one in Bayesian Binning into Quantiles [98]. The assumption in ENIR is that the mapping function converting uncalibrated scores into calibrated probablities is nearly isotonic by allowing score rank violations by the underlying algorithm and later penalizing them via the use or regularization term [98].

In an extensive set of experiments this assumption proved to be realistic and unlike the assumption in the isotonic regression not biased [98]. In two sets of experiments on 40 randomly selected datasets from the UCI [33] using logistic regression, support vector machines and Naïve Bayes as base classifiers significant calibration improvement was

observed using RMSE, ECE and MCE as calibration metrics. In addition ENIR outperformed both the isotonic regression and the BBQ method (Platt's scaling was not used due to its simplicity and also due to the fact that the BBQ method was already shown to outperform Platt's scaling [100].

2.2.11 Temperature Scaling

Guo *et al.* [44] demonstrated that deep convolutional neural networks (CNNs) are miscalibrated, with the probabilistic error and miscalibration becoming worse even as classification error is reduced during the model training. Guo *et al.* [44] proposed that modern advances in deep neural network architectures such as model complexity, batch-normalization and early-stopping strongly contributed to miscalibration — the deep neural networks become increasingly overconfident even as calibration worsens. To address this issue, Guo *et al.* [44] propose temperature scaling — a simple extension of Platt's scaling. Temperature scaling was originally used in statistical physics [54] and was later popularized by Hinton *et al.* [51] in the domain of knowledge distillation in deep neural networks.

Given the logit vector s_i, the new calibrated prediction is $\hat{q}_i = \max_k \sigma\left(s_i/T\right)^{(k)}$, where k is the class index and i is the index of an object x_i. The parameter T is called the *temperature* as it "raises the output entropy" [44]. As $T \to \infty$, the probability $\hat{q}_i \to 1/K$ resulting in maximum uncertainty. At $T = 1$ the original scores s_i are recovered and at $T \to 0$ the probability becomes that of a point mass [44].

Despite its simplicity and popularity, the temperature scaling method has a number of drawbacks, including dampening of overconfidence in both correct and incorrect predictions — as well as making calibration worse under covariance shift [102].

2.2.12 Entropy penalty

Pereyra *et al.* [106] systematically explore regularisation of neural networks. Compared to earlier techniques that have been used to reduce overfitting by acting on the activations or weights of a neural network (e.g., early stopping, dropout [124], batch normalization [53]) the proposed method follows entropy penalty approach by penalizing low entropy

output distributions — the same approach that has been successfully used in reinforcement learning to encourage exploration [151].

Overconfident predictions in deep neural networks are associated with output distributions that have low entropy, as shown by Szegedy *et al.* [127]. By penalizing low entropy via regularisation term overconfident (peaked) distributions can be mitigated leading to better generalization [106]. Pereyra *et al.* [106] illustrate the effects of various techniques such as dropout and confidence penalty on the softmax probabilities on the `MNIST` validation set. Whilst dropout leads to overconfident predictions that are either 0 or 1 (the effects also noted in Guo *et al.* [44]), both label smoothing and the confidence penalty via entropy regularizer lead to smoother output distributions and better regularization [106].

Definition 2.2.4 (Entropy)**.**

$$H\left(p_\theta(\boldsymbol{y} \mid \boldsymbol{x})\right) = -\sum_i p_\theta\left(\boldsymbol{y}_i \mid \boldsymbol{x}\right) \log\left(p_\theta\left(\boldsymbol{y}_i \mid \boldsymbol{x}\right)\right)$$

To construct the loss function Pereyra *et al.* [106] penalize overconfident distributions by adding negative entropy to the negative log-likelihood (NLL) as follows:

$$\mathcal{L}(\theta) = -\beta H\left(p_\theta(\boldsymbol{y} \mid \boldsymbol{x})\right) - \sum \log p_\theta(\boldsymbol{y} \mid \boldsymbol{x})$$

In experimental evaluations of the proposed confidence penalty on common benchmark datasets the authors find that confidence penalty improves the performance without modifying hyperparameters [106].

2.2.13 Maximum Mean Calibration Error (MMCE)

Methods like temperature scaling [44] and entropy penalty [106] improve calibration by clamping confidence [67], but in the process also penalize correct confident predictions. Kumar *et al.* [67] propose a principled approach to minimize calibration error using Maximum Mean Calibration Error (MMCE) — a kernel based measure that is trained in parallel with minimizing the NLL (negative likelihood loss). The MMCE is minimized at perfect calibration and enjoys the properties of consistent and fast convergence [67].

In experiments on datasets spanning images, NLP and time-series data and several network architectures Kumar *et al.* find that MMCE minimised calibration metrics whist preserving high confidence predictions [67].

2.2.14 Label smoothing

To address the issue of overconfidence in deep neural networks, Mueller *et al.* [95] propose label smoothing approach. Neural network training is sensitive to the loss function that is being minimized [95], whilst initially Rumelhart [114] derived backpropagation calculations using quadratic loss function, later on it was discovered that using cross entropy results in better performance and faster convergence (see, e.g, [8]). More recently Szegedy *et al.* [127] introduced label smoothing — an approach that computes cross entropy by using a weighted combination of target labels with the uniform distribution. Whilst label smoothing has long been used as a practical trick to improve performance of neural networks in image classification and neural language processing (NLP) tasks, it was generally not known when and why label smoothing works [95]. Mueller *et al.* demonstrate that in addition to improving deep neural network performance label smoothing implicitly calibrates predictions, making prediction confidence better aligned with accuracy maximisation [95].

In a neural network trained with label smoothing, instead of using hard labels y_i, the smoothed version of the labels $(1 - \alpha)y_i + \alpha/K$ is used to minimise cross entropy [95]. Using label-smoothing approach, the actual label y is weighted-averaged with one of the K class labels drawn randomly from uniform distribution.

Pereyra *et al.* have shown [106] that label smoothing is equivalent to confidence penalty if the uniform distribution and model outputs in the KL divergence is reversed. In a range of experiments using computer vision and NLP datasets, label smoothing improved performance and resulted in better generalization and improved calibration of deep neural networks [95]. Moreover, in on NLP tasks label smoothing improved translation quality (as measured by BLEU score) despite resulting in worse NLL [95].

2.2.15 Focal loss

Focal loss is a method that was originally developed to address the class imbalance problem encountered in dense object detectors by adding a modulating factor $(1 - p_i)^\gamma$ to the cross entropy loss [80].

Definition 2.2.5 (Cross Entropy (CE) loss)**.** Cross entropy loss is defined as:

$$\mathrm{CE}(p, y) = \mathrm{CE}\left(p_{\mathrm{t}}\right) = -\log\left(p_{\mathrm{t}}\right)$$

$$p_{\mathrm{t}} = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{if } y = 0. \end{cases}$$

Definition 2.2.6 (Focal loss)**.** Focal loss is then defined as:

$$FL = -\left(1 - p_t\right)^\gamma \log p_t.$$

When an object is misclassified and p_i is small, the modulating factor is close to 1 and does not affect cross entropy. As the model becomes more confident and $p_i \to 1$, the factor tends to zero and the entropy is down-weighted by dampening the effect of overconfident predictions. The parameter γ controls the effect of dampening overconfidence with $\gamma = 0$ corresponding to unmodulated cross entropy and by increasing γ the effect of modulation is increased.

Mukhoti *et al.* [94] extended the idea of using focal loss to improving model calibration. Whilst temperature scaling described in Guo *et al.* [44] is a popular modern variant of Platt's scaling (see 2.2.11 for more details) that can improve calibration of a deep neural network without affecting its accuracy, it has certain drawbacks, including reducing confidence in correct predictions at the same time as dampening overconfidence in incorrect predictions. Mukhoti *et al.* [94] proposed replacing cross-entropy with focal loss and have found that this is equivalent to minimising a regularised KL divergence between the predicted softmax distribution and the target label distribution. The focal loss is directing neural network's attention to correctly classified objects for which it is predicting low probabilities. At the same time, when compared to using cross-entropy loss, focal loss

indirectly regularises network weights during training by reducing gradient norms for confident samples [94]. Mukhoti *et al.* [94] outline a number of interesing findings:

- curse of misslassified samples — the increase in test NLL during training (indicating overfitting) is only caused by incorrrectly classifed samples, the miscalibration observed during training is linked to overfitting on textNLL in line with findings in [44].
- peak at the wrong place — the entropies for all objects (including both correctly and incorrectly classified) decline during the training, "this indicates that the network is becoming more and more confident about its incorrect predictions." This finding is similar to that observed in Pereyra *et al.* [106] described in more detail in 2.2.12.
- weight magnification — the overconfidence by neural network increasing the norm of its weights, cross-entropy induces such weight magnification during network training.

Another interesting finding is that whilst minimising cross entropy results in minimisation of the KL divergence between the softmax and the target distribution, the use of focal loss results in the minimisation of regularised KL divergence [94]:

$$FL \geq \mathrm{KL}(q\|\hat{p}) - \gamma H[\hat{p}].$$

Where q is the target distribution and $\hat{p}$ is the predicted distribution over the labels. Replacing cross-entropy with the focal loss therefore has the effect similar to that of adding entropy regulariser in [106] (described in 2.2.12), with the effect of both minimising KL divergence whilst simultaneuosly trying to increase the entropy of the prediction distribution $\hat{p}$. By dampening overconfident distributions, the focal loss improves calibration.

In experiments on a range of computer vision and document classification datasets, the focal loss approach was compared to the following baselines using the Brier loss (see 1.2.4, [10]) — the MMCE 2.2.13 and label smoothing 2.2.14. As shown in [29], the Brier loss can be decomposed into calibration and refinement terms and also imposes penalty on incorrect class probabilities [94]. In almost all experiments, using focal loss resulted in deep neural networks that were better calibrated than networks trained with baselines.

2.2.16 Venn predictors

As described in Chapter 1, Venn predictors (introduced in [144]) and described in [137, Chapter 5], it is not possible to estimate the true conditional probabilities of the class labels. Venn predictors (VP), while not as perfect as the true probabilities, are well-calibrated probabilistic predictors that, given a class label, output multi-probability distribution for each class label of the test object. As shown in [137, Chapter 6] Venn predictors achieve the calibration objective in a very strong non-asymptotic case. In general, one could expect Venn predictors to be well calibrated as long as one accepts the assumption of randomness [137].

Venn predictors are constructed by dividing the training set into categories. A test object is then assigned into one of the categories. Since at the time of such assignment the test object's label is unknown, one can try each label that an object might have and then compute the frequency of the labels in each category that an object has been assigned to depending on the chosen label. Obtained set of frequencies for the test object's unknown label can then be interpreted as a probability distribution over the test object's labels ([137]. The procedure obtains several probabilities even in a binary classification case, in practice if the number of examples in each category is large enough, such binary probabilities will be almost identical. Venn predictors output probability distributions that are guaranteed to contain well-calibrated probabilities under the assumption of randomness (see, e.g., [137], Theorem 6.6).

The main desiderata for Venn predictors are validity, predictive and computational efficiency. Venn predictors have been successfully combined with several underlying machine learning algorithms such as support vector machines [155], artificial neural networks [103] and K-nearest neighbours algorithm [25].

2.2.17 Venn-Abers predictors

A natural extension of Venn predictors are Venn-Abers predictors (VAP) — with the Abers part of the name formed by combining the surnames of the authors in Ayer *et al.* [5] that has described the underlying algorithm. Venn-Abers predictors can be built on

top of a wide range of machine learning classification algorithms that produce classification scores, such scores can then be converted into probabilities using the Venn-Abers method. As shown in [68], the method of isotonic regression for probabilistic calibration introduced by Zadrozny and Elkan [153] can often lead to mis-calibrated predictions (a similar conclusion was reached in [56]) and overfit on small datasets [76]. Another limitation of the isotonic regression is its reliance on the assumption that classification scores produced by the underlying classifier are ranked correctly — in practice this is equivalent to assuming that the AUC score is equal to 1 on the test set which is unrealistic to expect in practice [98].

Venn-Abers predictors are a modification of the isotonic regression [153], and as they are a variant of Venn predictors (that have an automatic property of validity) they are also perfectly calibrated. The property of automatic calibration comes with a cost, however, as Venn-Abers predictors are multi-probability predictors.

Chapter 3

Probabilistic classification

This chapter introduces a new method of probabilistic prediction in multi-class classification setting. The method coverts two computationally efficient calibration algorithms IVAP and CVAP into multi-class probabilistic predictors. The proposed multi-class predictors improve calibration for most classifiers and depending on the data set are often more experimentally accurate than classical calibration methods such as Platt's scaler and isotonic regression.

3.1 Introduction

Multi-class classification is the problem of classifying objects into one of the more than two classes. The goal of classification is to construct a classifier which, given a new test object, predicts the class label from the set of possible k classes. In an "ordinary" classification problem, the objective is to minimize the loss function by predicting correct labels on the test set. In contrast, a probabilistic classifier outputs, for a given test object, the probability distribution over the set of k classes. Probabilistic classifiers allow to express a degree of confidence about the classification of the test object. This is useful in a range of applications and industries where uncertainty around predictions needs to be quantified, including in life sciences, pharmaceutical R&D, self-driving cars, finance, robotics, election forecasting [131] and forecasting of pandemics such as COVID-19.

A number of techniques are available to solve binary classification problems. In logistic regression, the dependent variable is a categorical variable indicating one of the two possible classes. Logistic regression model is used to estimate the probability of a binary response based on a number of independent variables (features). In a large empirical

comparison of supervised learning algorithms by Caruana and Mizil [16], the logistic regression, whilst not competitive with the best methods, was the best model for some performance metrics for specific problems. Another technique that can be used for both binary and multi-class classification problem is the artificial neural network (ANN). Artificial neural network is a computational model based on a large collection of connected units called artificial neurons. Depending on the problem setting, an ANN could be used either to predict one of the k classes directly (by having k neurons in the final layer of the neural network) or indirectly by building separate networks for each of the k classes. ANNs are also able to estimate multi-class probabilities directly by using softmax function in the final layer of a neural-based classifier.

In a probabilistic classification setting where the loss function uses exact class probabilities, calibrating classification scores produced by the underlying classifier (whether machine learning or statistical) improves overall model performance. As shown in Caruana and Mizil [17], maximum margin methods such as support vector machines and boosted trees result in sigmoid-shaped distortion of the predicted probabilities. According to Caruana and Mizil [17], other methods such as neural networks and logistic regression do not suffer from these biases and result in better-calibrated probabilities, however more recent studies (see, e.g., [57]) have shown that traditional neural networks are often miscalibrated.

Platt's scaling [109] (also known as logistic scaling or logistic calibration and described in more detail in Section 2.2.2 of Chapter 2 [109]) is effective when the underlying machine learning algorithm produces sigmoid-shaped distortions in the predicted class scores — as this method was originally developed to address distortions in classification scores produced by support vector machines (SVM).

Platt's scaling, however, has several disadvantages. As shown in Pereira *et al.* [105] it is adequate for calibration of SVM' output that produces characteristic sigmoid-shaped distortion in classificaiton scores. Further, Kull *et al.* [65] show that parametric assumptions in Platt's scaler are equivalent to assuming that classifier scores are normally distributed with the same variance (σ^2) for each class. Experiments have shown (see, e.g., [65], [105]) that using Platt's scaling to calibrate output of many classifiers including AdaBoost and

Naïve Bayes can result in probability estimates that are in fact worse than scores produced by the underlying classifier in cases where score distributions have heavy tails.

Using Platt's scaling as a mapping function for well calibrated cases can result in model distortion as it does not include the identity function [65] (hence for well calibrated scores where no transformation is needed the well-calibrated scores produced by the underlying classifier will instead be distorted into poorly calibrated scores). Whist Platt's scaling is a parametric method and requires less data than isotonic regression (being a non-parametric method, isotonic regression can overfit on small datasets), it can result in miscalibration in cases where the score distribution model is misspecified. Kull *et al.* [65] demonstrate that "model mismatch is a real danger for a range of widely used machine learning models and can make classifiers less calibrated." In addition, Platt's scaling, being a variant of temperature scaling approach, suffers from the same drawbacks as temperature scaling, including dampening of confidence for both correct and incorrect predictions and making calibration worse under the covariat shift [102].

Another algorithm for probability calibration is isotonic regression (described in more detail in Section 2.2.3 of Chapter 2, [152]). The disadvantages of the isotonic regression include the assumption that the underlying classifier correctly ranks classification scores (this in turn is equivalent to assuming the ROC AUC score of 1 on the test dataset — the unrealistic assumption for most real life datasets. In addition, the isotonic regression is prone to overfitting, especially when the data is scarce. In the study of individual patients by Pereira *et al.* [105] isotonic regression performed better than Platt's scaling for classifiers other than SVM, however produced poor results when using decision trees as the underlying classifier .

Both classical and modern deep neural networks are miscalibrated (see, e.g., [57], [44], [89], [94], [105]). Despite increasing classification accuracy of modern deep learning neural architectures such as ResNet [45], their performance — as measured by classification accuracy — does not translate into improvements in terms of quality of probabilistic predictions. As shown in Guo *et al.* [44], modern neural networks become significantly miscalibrated even as their classification accuracy improves. The authors suggest that the effects of miscalibration appear to be linked to recent advancements in deep learning

designed to improve training and accuracy of neural networks — with both batch normalization and regularization having strong impact on network miscalibration [44] (see Chapter 2 for detailed overview of state-of-the-art in calibration methods for both classical machine learning models and deep learning networks).

More recently, Vovk *et al.* [141] introduced two new computationally efficient probabilistic predictors: IVAP (inductive Venn-Abers predictors) and CVAP (cross Venn-Abers predictors). IVAP can be considered a regularized form of calibration based on the isotonic regression. Whilst IVAP is constructed using two isotonic regression using different labels for the test object, the multiprobability prediction output (p_0,p_1) computed by IVAP is a form of regularisation and the interval width indicates confidence in prediction. Due to its regularized nature, IVAP are less prone to overfitting than isotonic regression (isotonic regression is described in more detail in Section 2.2.3 of Chapter 2). As IVAP are a special case of Venn-Abers predictors that have automatic guarantees of validity [141], IVAP are also automatically well-calibrated. CVAP are an extension of IVAP using the idea of cross-validation [141]. In empirical studies of pairwise classification problem Vovk *et al.* [141] demonstrated consistent accuracy of CVAP compared to the existing methods such as isotonic regression and Platt's scaling. In a recent biomedical informatics study by Pereira *et al.* [105], Venn-Abers predictors were shown to reduce both false positives and false negatives by pushing uncertainty estimates for incorrectly classified cases to the center of probability prediction intervals. Given that conformal predictors are automatically valid and validity of Venn-Abers predictors and IVAPs is also theoretically guaranteed, the study concluded [105] that conformal predictors and Venn-Abers predictors are the preferable methods to uncertainty estimation for individual patient predictions.

All probability calibration methods described so far were designed for binary classification problems. For a multi-class classification problem there are several techniques of assigning test objects to one of the k classes. If the conditional probabilities for each of the k classes are known or can be estimated, the multi-class classification problem is reduced to a trivial task of finding the number i of the class maximizing the conditional probability $p_i(x)$ computed for each of the k classes. In practice, estimating conditional class probabilities is a hard task, especially in high-dimensional setting with limited data

(the curse of dimensionality). In a binary classification task, finding a good separating function instead of conditional probabilities often gives better prediction results.

Binary probabilistic classifiers output class probabilities or decision scores. We use Platt's scaling, IVAP and CVAP to convert output of binary classifiers into calibrated binary class probabilities. We then use the PKPD method described in Section 3.2 to convert calibrated binary class probabilities into the multi-class probability distribution over the k classes.

The classical approach to multi-class classification is to consider a collection of binary classification problems and then combine their solutions (when solutions include pairwise class probabilities) to obtain multi-class probabilities. A number of methods for converting output of binary classifiers into multi-class probabilities are available. In a simple *one-versus-all* approach, k classifiers are built with the kth classifier separating all objects in the ith class from the objects in all other $k-1$ classes. The multi-class classifier $f(x)$ is then a function attaining $\arg\max_i f_i(x)$, where $f_i(x)$ is a classifier in binary classification problem of separating ith class from all the other classes. In another *one-versus-one* method (also called *all-pairs classification*) $\frac{k(k-1)}{2}$ binary classifiers are built to classify test objects between each pair of the ith and jth classes. If such classifiers are denoted as f_{ij}, the multi-class classification problem is reduced to finding $f_i(x)$ such that $f_i(x) = \arg\max_j \sum f_{ij}(x)$. Both one-versus-all and one-versus-one approaches generally perform well and the suitability of a method to the specific problem or application depends on the time needed to build a particular classifier in comparison with time required to repeat the classification task. For one-versus-one (OVO) algorithm, the number of repetitions is $O\left(N^2\right)$, whilst for one-versus-all (OVA) the number of repetitions is $O\left(N\right)$. However, if the time required to build a classifier is super-linear in terms of the number of objects, one-versus-one (OVO) is a more efficient choice.

As an alternative to solving multi-class classification by combining solutions to binary classification problems, approaches such as *single machine* and the *error correcting code* can be used. In the single machine approach in Weston and Watkins [150], a single optimization problem is solved by training a multi-class support vector machines to solve generalised binary support vector machines problem with the decision function

$f(x) = \arg\max_i(w_i x + b_i)$. This approach can be used for simultaneous multi-class separation in situations where binary classification by one-versus-all and one-versus-one fails. As an additional benefit, the single machine approach results in reducing the number of support vectors and more efficient kernel computations. The benefits of the single machine are, however, limited to the situations where it is hard to separate the data whilst at the same time meaningful subsets exist that allow for assigning a higher value to the decision function for the correct class as compared to other classes.

Other methods for solving multi-class classification problem include voting [111] and various methods based on combining binary probabilities to obtain multi-class probabilities. Such methods rely on obtaining estimates r_{ij} of pairwise probabilities $\mu_{ij} = P(y = i \mid y \in \{i, j\}, x)$. Estimates r_{ij} are obtained by building binary classifiers for each of the pairwise unions of the ith and jth classes and using r_{ij} as an approximation for μ_{ij}. In the next section, we describe a method of converting estimates r_{ij} of pairwise class probabilities into estimates p_i of multi-class probabilities for k classes.

3.2 Obtaining multi-class probabilities from pairwise classification

In order to obtain estimates of multi-class probabilities we convert binary class probabilities using the method in Price *et al.*[111]:

$$p_i^{\text{PKPD}} = \frac{1}{\sum_{j:j\neq i} \frac{1}{r_{ij}} - (k-2)}. \tag{3.1}$$

Class probabilities p_i^{PKPD} computed using 3.1 need to be normalized to ensure that they sum to one. We will refer to this method as the "PKPD" method. We use this method to obtain multi-class probabilities from pairwise class probabilities produced by the underlying algorithms (we use logistics regression, support vector machines and neural network) and calibrated using Platt's scaling, IVAP and CVAP. We then use multi-class probabilities computed using the "PKPD" method 3.1 to assign test object to one of the k classes which

allows us to compute loss metrics on the test dataset and compare them across different underlying machine learning methods and calibration algorithms.

3.3 Inductive and cross Venn-Abers predictors

As described in Chapter 1, Venn predictors (introduced in [144]) and described in [137, Chapter 5], it is not possible to estimate the true conditional probabilities of the class labels. Venn predictors (VP) (see 2.2.16 for more details), while not as ideal as the true probabilities, are well-calibrated probabilistic predictors that, given a class label, output multi-probability distribution for each class label of the test object. As shown in [137, Chapter 6] Venn predictors achieve the calibration objective in a very strong non-asymptotic case. In general, one could expect Venn predictors to be well calibrated as long as we accept the assumption of randomness [137].

A natural extension of Venn predictors is Venn-Abers predictors described in [141]. Venn-Abers predictors are built on top of a wide range of machine learning classification algorithms that produce classification scores, the calibration scores produced by the underlying machine learning algorithms can then be converted into probabilities using the Venn-Abers method. Venn-Abers predictors are a modification of the isotonic regression [153] and unlike the classification scores produced using Platt's scaling method or the method of the isotonic regression, the classification scores produced Venn-Abers predictors have theoretical guarantees or validity and are perfectly calibrated.

Calibration algorithms IVAP (inductive Venn-Abers predictor) and CVAP (cross Venn-Abers predictor) are computationally efficient versions of Venn-Abers predictors studied in [141]. Whilst IVAP and CVAP are based on the calibration method used by the isotonic regression [152], IVAP and CVAP avoid problems associated with isotonic regression such as lack of calibration guarantees or overfitting when the data is scarce. This is achieved by assigning each of the test objects with two potential labels of 0 and 1 to fit two isotonic regressions instead of one. As Venn-Abers predictors are a special case of Venn predictors, they inherit the property of perfect calibration from Venn predictors [141]. As shown by Vovk *et al.* [141], IVAP are automatically perfectly calibrated and the experimental results reported in the paper suggest that this property is inherited

by CVAP [141]. The property of automatic calibration comes with a cost, however, as Venn-Abers predictors are multi-probability predictors. IVAP and CVAP are computationally efficient algorithms with predictive efficiency depending on the efficiency of the underlying algorithms [141].

3.3.1 Computational details of IVAP and CVAP

In binary classification problems, machine learning classifiers output prediction score s for each test object, class predictions are then obtained by comparing the score for each test objects to a specified threshold. To obtain calibrated scores, function $g(s(x))$ is then applied to convert uncalibrated scores s to calibrated probabilities. Assuming that the underlying classifier ranks test objects correctly (with objects that are more likely to end up in class 1 being assigned higher classification scores), such function $g(s)$ should be isotonic (non-decreasing) function in order to preserve original ranking produced by the classifier. Figure 3.1 shows a general illustration of an isotonic regression fit to an data set.

Isotonic function $g(s(x))$ can be constructed by fitting isotonic regression to the set of points $(s(x_i), y_i)$, where $s(x_i)$ is the classification score for the ith object in the calibration data set and y_i is the actual label of this object. This is achieved by maximizing the likelihood on the calibration set, where K is the size of the calibration set

$$\prod_{i=1,2,\ldots,K} p_i \tag{3.2}$$

and p_i is defined as follows:

$$p_i = \begin{cases} g(s(x_i)) & \text{if } y_i = 1 \\ 1 - g(s(x_i)) & \text{if } y_i = 0 \end{cases} \tag{3.3}$$

The most widely used algorithm for learning isotonic function $g(s)$ is the pool-adjacent-violators algorithm (PAVA, [7]). PAVA has linear time and memory complexity and works by scanning uncalibrated scores s_i starting with the first object in the test data set and each time comparing s_{i-1} with s_i to find violations of monotonicity. Every time such monotonicity violation is found, both s_{i-1} with s_i are replaced with their mean

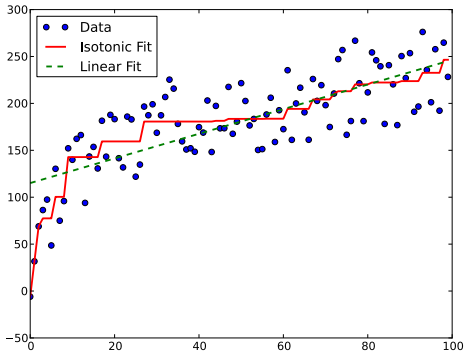

FIGURE 3.1: Isotonic regression, Wikipedia

$(s_{i-1} + s_i)/2$. If such replacement results in violations for test objects scanned earlier, the algorithm needs to go back to fix such earlier monotonicity violations via averaging before proceeding to next object. Figure 3.2 is an example of static snapshot of PAVA algorithm replacing monotonicity violations with values of their mean $(s_{i-1} + s_i)/2$.

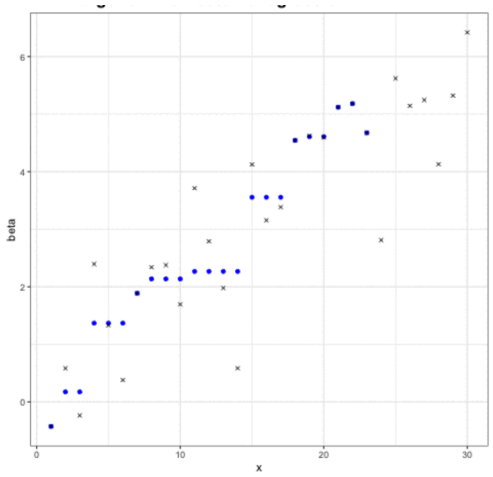

FIGURE 3.2: PAVA algorithm for isotonic regression

Venn-Abers predictors are based on the idea of isotonic regression [141]. Venn-Abers predictors are essentially a regularized version of isotonic regression [141], but instead of fitting isotonic regression to the calibration set once, it is fitted twice to the calibration set augmented with test object, each time assuming that the label of a test object is known it can be either 0 and 1 or accordingly. The first isotonic regression $g_0(s)$ is therefore fitted

to the set of points

$$((s_0(x_1), y_1), (s_0(x_2), y_2), \ldots, (s_0(x_K), y_K), (s_0(x), 0)) \tag{3.4}$$

whilst the second isotonic regression $g_1(s)$ is fitted to the set of points

$$((s_1(x_1), y_1), (s_1(x_2), y_2), \ldots, (s_1(x_K), y_K), (s_1(x), 1)) \tag{3.5}$$

Inductive Venn–Abers predictors (IVAP) uses classification scores $s_1, \ldots, s_K$ computed by the underlying machine learning or statistical classification algorithm on the calibration set of size K (calibration set can be obtained by reserving a part of the training set, the other remaining part of the original training set that we will refer to as *the proper training set* is used for training the underlying machine learning or statistical model) and also the score s computed for each new test object. The isotonic regression is then fitted twice to the set of computed scores $s_1, \ldots, s_K, s$ (used as the independent variables), and two sets of dependent variables formed by combining the labels of the calibration objects with the two (either 0 or 1) potential labels for the test object. By fitting isotonic regression twice, IVAP computes multi-probability prediction (p_0,p_1) for the test object that can be interpreted as the lower and the upper probability respectively of the object belonging to the class labelled as 1. IVAP computes (p_0,p_1) efficiently for each of the potential test objects by pre-computing two vectors F^0 and F^1 which store $f_0(s)$ and $f_1(s)$, respectively, for all possible values of s. As shown by Vovk *et al.* [141], given the scores $s_1, \ldots, s_K$ of the calibration objects computed by the underlying algorithm, the IVAP prediction rule can be computed in time $O(K \log K)$ and space $O(K)$, where K is the size of the calibration set.

A cross Venn-Abers predictor (CVAP) is just a combination of K IVAP, where K is the number of folds in the training set. To obtain class probabilities using CVAP, Vovk *et al.* [141] propose to use minimax method to merge K multiprobability predictions computed by K IVAP. For the Log-loss, the multiprobability prediction for CVAP is an interval $(1 - GM(1 - p_0), GM(p_1))$ obtained by computing geometric means of multiprobability

predictions arising out of repeated application of IVAP to K folds ($GM(p_1)$ is the geometric mean of $p_1^1, \ldots, p_1^K$ and $GM(1-p_0)$ is the geometric mean of $1-p_0^1, \ldots, 1-p_0^K$). For the Brier loss, the merged probability is given by formula $p = \frac{1}{K}\sum_{k=1}^{K}\left(p_1^k + \frac{1}{2}(p_0^k)^2 - \frac{1}{2}(p_1^k)^2\right)$.

The minimax method can also be applied to IVAP to obtain single probability prediction by combining multi-probability prediction as follows: $p := p_1/(1-p_0+p_1)$ (see Vovk *et al.* [141] for more details).

3.4 Experiments on multi-class data sets

We present the experimental results using several multi-class data sets: `waveform`, `iris` and `abalone` from the UCI Machine Learning Repository [33], `satimage` and `vehicle silhouettes` from the Statlog collection [90], `balance-scale`, `pasture` and `eye movements` from [132]. To measure the quality of probabilistic predictions and calibrators we follow Gneiting *et al.* [40] and use the proper scoring rules - the Log loss 1.2.3 and the Brier loss 1.2.4 (see Section 1.2.1 of Chapter 1 for more details).

In both cases p is the vector of class probabilities and y is the vector of true labels one-hot encoded across the K classes. Both the Log loss and the Brier loss are computed by taking arithmetic mean of the respective losses on the individual objects from the test dataset. One advantage of the Brier loss is that it is still possible to compare quality of predictions in cases where a prediction algorithm produces infinite Log loss. In this section we compare the performance of IVAP and CVAP with that of Platt's scaling [109] and the isotonic regression [153].

For all experiments, we use the same underlying algorithms: Naïve Bayes, KNN, Support-vector machine (SVM), the logistic regression, simple neural network, Random Forest, LightGBM, XGBoost, CatBoost and Ada Boost. With the exception of LightGBM [58], XGBoost [20], CatBoost [32], all the other algorithms are available in Scikit-learn [11]. All the underlying classifiers have been used with the default parameters. The underlying algorithms output binary classification scores which are then calibrated by applying Platt's scaling, isotonic regression, IVAP and CVAP. We have used implementations of Platt's scaling and the isotonic regression from Scikit-learn [11].

To obtain pairwise classification scores, we run the ten underlying machine learning algorithms: Naïve Bayes, KNN, Support-vector machine (SVM), the logistic regression, simple neural network, Random Forest, LightGBM, XGBoost, CatBoost and Ada Boost. We then apply the PKPD (method described in 3.1) to convert pairwise classification scores into calibrated multi-class probabilities.

We use OpenML [132] to access all the datasets to ensure consistency and reproducibility of the experiments: `waveform` (ID:60), `satimage (ID:182)`, `vehicle silhouettes (ID:54)`, `iris(ID:61)`, `abalone (ID:1557`, `balance-scale (ID:11)`, `eye movements (ID:1044)` and `pasture (ID:462)`.

From each dataset, three datasets were created. We use scikit-learn functionality to create test set using 20% of all data, and use the remaining training set to create calibration set equal in length to the test set. The remaining data is used for proper training set used to train underlying classifiers. To ensure reproducibility we use random seed of 42 and default values in scikit-learn across all experiments.

The data sets and the results of the experiments are described below.

3.4.1 `Waveform` data set

`Waveform` is an artificial data set [33] containing three different classes of waves with a total of 5,000 instances and 40 attributes. Each class is generated by combining two or three base waves and adding noise to each attribute. The waveform dataset is balanced with about one third of objects in each of the three classes.

We split the `waveform` dataset as described in section 3.4 into the training set of 4,000 observations) and the test set (1000 observations). We further split the training set into the proper training set and the calibration set equal in length to the test set. This split results in 3,000 objects allocated to the proper training set and 1,000 objects in both calibration and test set each. Tables 3.1 and 3.2 refer to the results of experiments.

Based on the Log loss, using CVAP to calibrate pairwise classification scores, results in performance improvements for seven out of the ten classifiers. Whilst isotonic regression improves calibration for KNN and neural network and Platt's scaler improves calibration for Random Forest, CVAP delivers the best overall improvement in calibration for four out

TABLE 3.1: The Log loss for the `waverfont` data set

	no calibration	sigmoid	isotonic	ivap	cvap
Naïve Bayes	0.5096	0.2756	0.3189	0.2279	0.2267
KNN	0.4759	0.2686	0.2655	0.3644	0.3506
Support Vector Machine	0.1881	0.1927	0.2536	0.1985	0.1914
logistic regression	0.1963	0.1987	0.2576	0.2052	0.2037
neural network	0.4172	0.3601	0.3333	0.4036	0.3925
Random Forest	0.2491	0.2002	0.2192	0.2066	0.2054
LightGBM	0.2377	0.2534	0.2331	0.2199	0.2080
XGBoost	0.2657	0.2510	0.2319	0.2189	0.2103
CatBoost	0.1948	0.2308	0.2246	0.2158	0.2013
Ada Boost	0.5808	0.2447	0.3360	0.2498	0.2278

TABLE 3.2: The Brier loss for the `waveform` data set

	no calibration	sigmoid	isotonic	ivap	cvap
Naïve Bayes	0.1114	0.0847	0.0689	0.0698	0.0694
KNN	0.0826	0.0843	0.0849	0.1080	0.1024
Support Vector Machine	0.0586	0.0595	0.0605	0.0603	0.0577
logistic regression	0.0605	0.0610	0.0616	0.0617	0.0611
neural network	0.1063	0.1136	0.1075	0.1280	0.1247
Random Forest	0.0741	0.0631	0.0626	0.0639	0.0621
LightGBM	0.0719	0.0751	0.0683	0.0683	0.0639
XGBoost	0.0752	0.0738	0.0681	0.0679	0.0645
CatBoost	0.0626	0.0695	0.0660	0.0665	0.0610
Ada Boost	0.1975	0.0766	0.0764	0.0771	0.0692

TABLE 3.3: The Log loss for the satimage data set

		sigmoid	isotonic	ivap	cvap
Naïve Bayes	1.0158	0.1651	0.1627	0.1555	0.1529
KNN	0.1849	0.0802	0.1140	0.2691	0.2717
Support Vector Machine	0.0740	0.0794	0.1403	0.0891	0.0907
logistic regression	0.0941	0.0990	0.1409	0.1064	0.1050
neural network	0.1035	0.0923	0.1139	0.1801	0.1766
Random Forest	0.0748	0.0768	0.1331	0.0947	0.0948
LightGBM	0.0732	0.0806	0.1153	0.0787	0.0775
XGBoost	0.0709	0.0845	0.1180	0.0832	0.0800
CatBoost	0.0639	0.0785	0.1373	0.0824	0.0812
Ada Boost	0.3098	0.0914	0.1494	0.0933	0.0915

ten classifers: Naïve Bayes, LightGBM, XGBoost and Ada Boost. Based on the Brier loss, using CVAP to calibrate pairwise classification scores results in performance improvements for seven out of the ten classifiers. Whilst isotonic regression improves calibration for Naïve Bayes, CVAP delivers the best overall improvement in calibration for five out ten classifers: Support Vector Machine, LightGBM, XGBoost, CatBoost and Ada Boost.

3.4.2 Satimage data set

The satimage data set [33] contains images representing seven different classes of soil, ranging from red or gray soil to soil containing crops such as cotton or vegetation stubble. The data set was collected to predict the soil type from the new satellite images, given the multi-spectral values. The number of attributes is 36 and the number of instances is 6 435. Tables 3.3 and 3.4 refer to the results of experiments.

Based on the Log loss, using CVAP to calibrate pairwise classification scores, results in significant performance improvement for Naïve Bayes and also Ada Boost. Platt's scaler provides significant improvement for KNN and neural network, whilst for other underlying classifiers no significant improvement in calibration occurs by using any of the four considered calibration methods. The situation is similar when using Brier loss — most underlying algorithms do not see improvement from additional calibration.

Based on the Brier loss, using CVAP to calibrate pairwise classification scores results in performance improvements for for Naïve Bayes, Support-Vector Machine and also neural

TABLE 3.4: The Brier loss for the satimage data set

		sigmoid	isotonic	ivap	cvap
Naïve Bayes	0.0617	0.0478	0.0434	0.0445	0.0431
KNN	0.0211	0.0227	0.0223	0.0785	0.0784
Support Vector Machine	0.0215	0.0226	0.0231	0.0240	0.0237
logistic regression	0.0291	0.0300	0.0299	0.0303	0.0297
neural network	0.0233	0.0244	0.0232	0.0464	0.0446
Random Forest	0.0213	0.0218	0.0226	0.0238	0.0228
LightGBM	0.0195	0.0214	0.0199	0.0210	0.0198
XGBoost	0.0186	0.0226	0.0219	0.0224	0.0205
CatBoost	0.0190	0.0217	0.0211	0.0220	0.0209
Ada Boost	0.0933	0.0268	0.0253	0.0256	0.0245

network.

3.4.3 Vehicle silhouettes data set

Vehicle silhouettes data set from the Statlog collection [33] was designed to find a method of distinguishing between 3D objects within a 2D image by application of an ensemble of shape feature extractors to the 2D silhouettes of the objects. Four Corgie model vehicles were used: Chevrolet van, SAAB 9000, double-decker bus and Opel Manta 400. This particular combination of vehicles was chosen with the expectation that buses, vans and either one of the car types would be readily distinguishable, but it would be more difficult to distinguish between the two car classes. The data set contains 946 instances and 18 attributes. We run the same underlying algorithms and calibration methods as for all the previous data sets. Tables 3.5 and 3.6 refer to the results of experiments.

Based on the Log loss, using CVAP to calibrate pairwise classification scores, results in significant performance improvement for Naïve Bayes and also neural network. Platt's scaler provides significant improvement for KNN and isotonic regression for Ada Boost. CVAP provides significant improvement for Naïve Bayes, KNN, neural network, XGBoost and Ada Boost.

TABLE 3.5: The Log loss for the `vehicle` data set

		sigmoid	isotonic	ivap	cvap
Naïve Bayes	1.1130	0.4949	0.7742	0.4738	0.4465
KNN	0.7008	0.3695	0.8102	0.4382	0.4051
Support Vector Machine	0.4350	0.4707	0.8722	0.4552	0.4370
logistic regression	0.1835	0.2224	0.6953	0.2328	0.2316
neural network	0.5645	0.5790	0.5791	0.5693	0.5634
Random Forest	0.2293	0.2249	0.6256	0.2530	0.2474
LightGBM	0.2142	0.2319	0.5319	0.2451	0.2455
XGBoost	0.2672	0.2367	0.4394	0.2528	0.2490
CatBoost	0.2505	0.2316	0.5384	0.2550	0.2517
Ada Boost	0.4547	0.2438	0.2034	0.2579	0.2483

TABLE 3.6: The Brier loss for the `vehicle` data set

		sigmoid	isotonic	ivap	cvap
Naïve Bayes	0.2137	0.1629	0.1628	0.1566	0.1457
KNN	0.1135	0.1162	0.1160	0.1414	0.1280
Support Vector Machine	0.1437	0.1566	0.1552	0.1511	0.1434
logistic regression	0.0590	0.0674	0.0616	0.0683	0.0678
neural network	0.1883	0.1937	0.1937	0.1901	0.1879
Random Forest	0.0737	0.0705	0.0703	0.0755	0.0739
LightGBM	0.0727	0.0729	0.0712	0.0738	0.0740
XGBoost	0.0856	0.0753	0.0719	0.0762	0.0753
CatBoost	0.0746	0.0723	0.0727	0.0770	0.0752
Ada Boost	0.1465	0.0759	0.0668	0.0769	0.0730

TABLE 3.7: The Log loss for the `balance scale` data set

		sigmoid	isotonic	ivap	cvap
Naïve Bayes	0.2964	0.2573	0.5345	0.2896	0.2291
KNN	0.9938	0.2953	0.3743	0.3998	0.3657
Support Vector Machine	0.1539	0.2190	2.3633	0.2373	0.2106
logistic regression	0.1761	0.2025	0.3493	0.2416	0.2118
neural network	0.2173	0.2047	0.5427	0.2657	0.2765
Random Forest	0.2763	0.3157	1.3064	0.3216	0.2910
LightGBM	0.2092	0.2986	2.4583	0.3091	0.2554
XGBoost	0.1808	0.2902	1.0373	0.2699	0.2250
CatBoost	0.2717	0.2636	2.0713	0.2850	0.2399
Ada Boost	0.5842	0.1352	1.1294	0.1388	0.1512

TABLE 3.8: The Brier loss for the `balance scale` data set

		sigmoid	isotonic	ivap	cvap
Naïve Bayes	0.0823	0.0771	0.0967	0.0876	0.0618
KNN	0.0756	0.0867	0.0868	0.1196	0.1030
Support Vector Machine	0.0468	0.0708	0.0862	0.0707	0.0578
logistic regression	0.0529	0.0611	0.0570	0.0689	0.0569
neural network	0.0547	0.0575	0.0616	0.0712	0.0720
Random Forest	0.0822	0.0961	0.1017	0.0967	0.0815
LightGBM	0.0615	0.0900	0.1185	0.0944	0.0699
XGBoost	0.0587	0.0830	0.0931	0.0817	0.0614
CatBoost	0.0722	0.0809	0.1008	0.0851	0.0654
Ada Boost	0.1988	0.0397	0.0405	0.0351	0.0347

3.4.4 `Balance scale` data set

`Balance scale` is a synthetic data set generated to model psychological experimental results. Each object is classified as having the balance scale tip to the right, tip to the left, or be balanced. The attributes are the left weight, the left distance, the right weight, and the right distance [33]. The data set contains 625 distinct values, 4 features and 3 classes. Tables 3.7 and 3.8 refer to the results of the experiments.

Based on the Log loss, using CVAP to calibrate pairwise classification scores results in significant performance improvement for Naïve Bayes and also CatBoost. Platt's scaler provides improvement for KNN, neural network and Ada Boost. CVAP provides significant improvement for Naïve Bayes, KNN, neural netrowk, XGBoost and Ada Boost.

TABLE 3.9: The Log loss for the eye movement data set

		sigmoid	isotonic	ivap	cvap
Naïve Bayes	1.2293	0.6141	0.6360	0.6025	0.6015
KNN	1.5009	0.5810	0.5796	0.5853	0.5832
Support Vector Machine	0.6004	0.6031	0.6027	0.5912	0.5900
logistic regression	0.5788	0.5799	0.6046	0.5758	0.5728
neural network	0.6316	0.6316	0.6446	0.6306	0.6311
Random Forest	0.4705	0.4490	0.5055	0.4506	0.4393
LightGBM	0.3996	0.4164	0.4364	0.4162	0.3965
XGBoost	0.3823	0.4197	0.4390	0.4181	0.3850
CatBoost	0.4031	0.4208	0.4675	0.4220	0.4078
Ada Boost	0.6279	0.5208	0.5362	0.5151	0.5063

TABLE 3.10: The Brier loss for the eye movement data set

		sigmoid	isotonic	ivap	cvap
Naïve Bayes	0.2560	0.2127	0.2083	0.2081	0.2077
KNN	0.2136	0.1996	0.1991	0.2008	0.2000
Support Vector Machine	0.2067	0.2072	0.2040	0.2037	0.2033
logistic regression	0.1973	0.1978	0.1972	0.1971	0.1959
neural network	0.2196	0.2195	0.2195	0.2196	0.2199
Random Forest	0.1518	0.1468	0.1470	0.1472	0.1431
LightGBM	0.1294	0.1358	0.1359	0.1362	0.1287
XGBoost	0.1237	0.1359	0.1363	0.1364	0.1236
CatBoost	0.1304	0.1373	0.1384	0.1384	0.1329
Ada Boost	0.2184	0.1757	0.1744	0.1746	0.1709

Based on the Brier loss, using CVAP improves to calibrate pairwise classification scores, results in calibration improvements for Naïve Bayes, Random Forest, CatBoost and Ada Boost.

3.4.5 Eye movement data set

The Eye movement data set is from the "Inferring relevance from eye movements" challenge [116]. The objective of the challenge was to predict, based on eye movement data, whether a reader finds a text relevant. The data set contains $10/,936$ distinct values and 22 features.Tables 3.9 and 3.10 refer to the results of the experiments.

TABLE 3.11: The Log loss for the pasture data set

		sigmoid	isotonic	ivap	cvap
Naive Bayes	4.1875	0.4446	6.5789	0.4898	0.5215
KNN	0.4272	0.5852	3.7178	0.5654	0.5684
Support Vector Machine	0.4830	0.9266	6.5789	0.5188	0.5621
logistic regression	6.6237	0.7512	10.0278	0.5879	0.5813
neural network	0.6583	0.6981	0.7352	0.6635	0.6438
Random Forest	0.4217	0.3991	0.2978	0.5301	0.4955
LightGBM	0.6583	0.6981	0.7352	0.6635	0.6438
XGBoost	0.3724	0.4631	3.3375	0.6231	0.6214
CatBoost	0.4200	0.4638	3.3498	0.5006	0.4900
Ada Boost	1.1328	0.4297	3.2894	0.6230	0.6312

Based on the Log loss, using CVAP to calibrate pairwise classification scores, results in performance improvements for eight out of the ten classifiers. Whilst isotonic regression delivers the best calibration result for KNN, IVAP is the best calibration approach for neural network and CVAP delivers the best overall improvement in calibration for the remaining eight out of ten classifiers.

Based on the Brier loss, using CVAP to calibrate pairwise classification scores, results in the best performance improvements for seven out of the ten classifiers. Whilst isotonic regression delivers the best calibration result for KNN and neural network, CVAP delivers the best overall improvement in calibration for the seven out of ten classifiers.

3.4.6 Pasture data set

The objective of the dataset was to predict pasture production from a variety of biophysical factors [132]. The dataset contains vegetation and soil variables from areas of grazed North Island hill country with different variables selected as potentially useful biophysical indicators. Tables 3.11 and 3.12 refer to the results of the experiments.

Based on the Log loss, whilst isotonic regression delivers the best calibration result for Random Forest, CVAP is the best calibration approach for neural network and CVAP delivers the best overall improvement in calibration for logistic regression, neural network and LightGBM.

TABLE 3.12: The Brier loss for the `pasture` data set

		sigmoid	isotonic	ivap	cvap
Naive Bayes	0.1905	0.1435	0.1905	0.1594	0.1704
KNN	0.1500	0.2112	0.2573	0.1931	0.1923
Support Vector Machine	0.1633	0.1955	0.1905	0.1706	0.1891
logistic regression	0.3786	0.2588	0.3452	0.2062	0.1999
neural network	0.2320	0.2466	0.2585	0.2336	0.2255
Random Forest	0.1273	0.1241	0.1080	0.1761	0.1595
LightGBM	0.2320	0.2466	0.2585	0.2336	0.2255
XGBoost	0.1018	0.1478	0.1036	0.2156	0.2151
CatBoost	0.1286	0.1488	0.1108	0.1626	0.1569
Ada Boost	0.0952	0.1325	0.0952	0.2164	0.2197

TABLE 3.13: The Log loss for the `abalone` data set

		sigmoid	isotonic	ivap	cvap
Naive Bayes	1.0651	0.5350	0.5877	0.5053	0.5026
KNN	1.4360	0.4756	0.4707	0.5008	0.4929
Support Vector Machine	0.4631	0.4728	0.5503	0.4669	0.4604
regression	0.4774	0.4777	0.5625	0.4798	0.4773
neural network	0.4524	0.4601	0.4999	0.4546	0.4525
Random Forest	0.4454	0.4527	0.5316	0.4514	0.4450
LightGBM	0.4726	0.4715	0.5701	0.4653	0.4499
XGBoost	0.5015	0.4712	0.5110	0.4641	0.4458
CatBoost	0.4423	0.4520	0.5845	0.4480	0.4433
Ada Boost	0.6075	0.4977	0.5625	0.4787	0.4598

Based on the Brier loss, using CVAP to calibrate pairwise classification scores, results in the best performance improvements for seven out of the ten classifiers. Whilst isotonic regression delivers the best calibration result for KNN and neural network, CVAP delivers the best overall improvement in calibration for the seven out of ten classifiers.

3.4.7 `Abalone` data set

Abalone data set was created to facilitate predicting the age of abalone from physical measurements [34]. Tables 3.13 and 3.14 refer to the results of the experiments.

TABLE 3.14: The Brier loss for the abalone data set

		sigmoid	isotonic	ivap	cvap
Naive Bayes	0.2187	0.1792	0.1701	0.1704	0.1696
KNN	0.1656	0.1570	0.1553	0.1647	0.1614
Support Vector Machine	0.1528	0.1565	0.1560	0.1560	0.1530
logistic regression	0.1590	0.1596	0.1603	0.1602	0.1594
neural network	0.1500	0.1515	0.1505	0.1508	0.1505
Random Forest	0.1481	0.1487	0.1481	0.1488	0.1467
LightGBM	0.1556	0.1555	0.1541	0.1544	0.1490
XGBoost	0.1630	0.1551	0.1538	0.1538	0.1471
CatBoost	0.1469	0.1485	0.1488	0.1482	0.1465
Ada Boost	0.2093	0.1626	0.1598	0.1595	0.1526

Based on the Log loss, using CVAP results in the best results for seven out of ten underlying classifiers, including Naïve Bayes, SVMs, logistic regression, Random Forest, LightGBM, XGBoost and Ada Boost.

Similar performance improvements are observed using the Brier loss, using CVAP to calibrate pairwise classification scores results in performance improvements for six out of the ten classifiers.

3.5 Conclusion

Machine learning has made a remarkable progress. A number of classical machine learning methods such as boosted trees, random forest and support vector machines demonstrate excellent performance exceeding the performance of earlier classical machine learning algorithms such as K-means or the logistic regression. Calibration of classification scores using traditional calibration methods such as Platt's scaling or the isotonic regression can improve the performance of the underlying algorithms.

More recently, a growing body of research demonstrated that modern deep neural networks are no longer well calibrated (see, e.g., [44], [89], [94], [105]). Whilst very large deep computer vision models such as ResNet [45] are much more accurate than previous classical architectures such as LeNet [77], such modern deep neural networks architectures become significantly miscalibrated even as classification accuracy continues

to improve. As demonstrated in Guo *et al.* [44] recent advances in deep learning such as model capacity, batch normalization, weight decay have strong negative effects on network calibration.

Whilst the earlier publications considered traditional neural networks well-calibrated [17], recent research has demonstrated [57] that both single neural network and ensembles of traditional neural networks are often poorly calibrated [57].

The calibration of both classical machine learning algorithms, neural and deep neural networks is becoming increasing important, especially in many real-world application such as healthcare [105] and self-driving cars [94]. In such critical applications obtaining accurate class probabilities significantly affects decision-making, such as whether to stop a self-driving car when the algorithm is unsure about its prediction about whether there is a pedestrian on the road [94].

The main contribution of this chapter is the empirical study of the performance of two computationally efficient calibration algorithms IVAP and CVAP in the multi-class classification setting. Multi-class probabilistic predictors based on IVAP and CVAP perform well, improving calibration in general and often resulting in performance improvements in comparison with the results obtained from using Platt's scaling and isotonic regression. The improvements in performance in comparison with the results produced by the underlying algorithms in multi-class classification problems are comparable to improvements reported for binary classification problems (see, e.g., Vovk *et al.* [141] and Pereira *et al.* [105]). The proposed multi-class predictors improve calibration for most classifiers and depending on the data set are often more experimentally accurate than classical calibration methods such as Platt's scaler and isotonic regression.

Chapter 4

Probabilistic regression

This chapter focuses on the study of probabilistic regression and applies conformal prediction to derive predictive distribution functions that are valid under a non-parametric assumption.

4.1 Introduction

Chapter 3 introduced a new method of probabilistic prediction in the multi-class classification setting. In this and further chapters (Chapters 4–6), the focus is on probabilistic prediction for machine learning regression problems. We introduce and develop non-parametric approach to predictive distribution functions using conformal prediction. This approach results in predictive distribution functions that are always valid for general machine learning assumption (IID observations) in terms of guaranteed coverage. We define predictive distribution functions in line with definitions and terminology in Shen *et al.* [121] and Schweder *et al.* [117].

In statistics, the theory of predictive distributions (see, e.g., [117], [121]) is based on the assumption that samples are generated from a parametric model. Our novel contribution extends statistical "predictive confidence distributions" in terms of their ability to generate probabilistic predictions using only limited assumptions customary of machine learning — namely that the observations are generated using the IID model. Unlike in the parametric approach of statistical predictive distributions, our novel approach is completely data-driven non-parametric approach that does not require specification of the data model.

Our approach generalizes the classical Dempster-Hill procedure (further formally defined in Section 4.8). Predictive distributions are briefly reviewed in 4.2, for a more detailed review of predictive distributions, we refer the reader to [73].

4.2 Predictive distributions

Lawless and Fredette [73] consider the general parametric prediction framework based on the family of models:

$$F(y \mid x; \theta) = \mathsf{P}(Y \leqslant y \mid X = x; \theta) \tag{4.1}$$

Where following our terminology **Y** and **X** represent sets of objects and labels accordingly. In this parametric approach, the distribution function F is specified by the parameter vector θ. Given the rather general model specification, **X** can specify both cross-sectional and time-series data. If θ is known, it fully quantifies cumulative predictive distribution of **Y** allowing to make probabilistic statements about **Y** given **X**. If θ is unknown, it must be estimated from the data.

Definition 4.2.1 (Coverage Probability)**.** Coverage probability is defined as:

$$CP(\theta) = \mathsf{P}(\{L_1(X) \leqslant Y \leqslant L_2(X); \theta\})$$

Prediction intervals have well defined frequentist probability interpretation [73]. In this and further chapters (Chapters 4–6), the focus is on probabilistic prediction for machine learning regression tasks using frequentist methods.

Lawless and Fredette [73] provided a unified treatment for both frequentist prediction intervals and predictive distributions by developing a definition of a predictive distribution as a confidence distribution, as well as outlining the method of obtaining predictive distributions using pivotal quantiles, the method they have referred to as the *pivotal method*. The benefits of such an approach include obtaining prediction intervals and predictive distributions that are both well-calibrated and have clear frequentist probability interpretations. Such predictive distributions can be generated efficiently and "possess

good properties when considered as estimators of the true distributions of **Y** given **X**" [73].

Shen *et al.* [121] further develop the approach in Lawless and Fredette [73] by defining a general approach for constructing predictive distribution of **Y** using a confidence distribution of the unknown parameter θ as follows (where $H(\theta; \mathbf{y})$ is a confidence distribution for θ derived from the training set):

$$Q(z_1, \ldots, z_n, (x_{n+1}, y)) = \int_{\theta \in \Theta} F_\theta(y)\, dH(\theta; \mathbf{y})$$

4.3 Predictive distribution functions

Many probabilistic regression applications require prediction of the label y_{n+1} (we use **y** and y_{n+1} interchangeably to describe the label of the $(n+1)$st observation) given a *training sequence* of observations $z_i = (x_i, y_i)$, $i = 1, \ldots, n$ and a new test object $x_{n+1} \in \mathbb{R}^p$. Given the rather general model specification, **X** covers both regression problems with *objects* **x** from the set **X** containing explanatory variables for *labels* **y** from the set **Y**, as well as time-series problems with objects **x** containing previous history and time-series based features of the label **y**.

We consider the regression problem with p attributes. Objects **x**, where $x \in \mathbb{R}^p$, and labels **y**, where $y \in \mathbb{R}$, are IID *observations* $z_i = (z_i, y_i)$ from the *observation space* **Z** ($\mathbf{Z} \in \mathbb{R}^{p+1} = \mathbb{R}^p \times \mathbb{R}$). The regression task is to predict y_{n+1}, given a training sequence of observations $z_i = (z_i, y_i)$ and a new test object $x_{n+1} \in \mathbb{R}$.

4.4 Randomized and conformal predictive distributions

The statistical predictive distributions in [73] and [121] rely on parametric models. In this chapter we use conformal prediction framework to develop novel methods of constructing predictive distributions for the general non-parametric case. We initially define distribution functions following Shen *et al.* [121, Definition 1], but allowing for randomization.

Randomization is an inherent feature of conformal predictive distributions (CPDs), however in practice they are affected little by it. Let U be the uniform probability measure on the interval $[0, 1]$.

Definition 4.4.1 (Randomized Predictive System)**.** A function $Q : (\mathbb{R}^{p+1})^{n+1} \times [0, 1] \to [0, 1]$ is called a *randomized predictive system* (**RPS**) if it satisfies the following three requirements:

R1a For each training sequence $(z_1, \ldots, z_n) \in (\mathbb{R}^{p+1})^n$ and each test object $x_{n+1} \in \mathbb{R}^p$, the function $Q(z_1, \ldots, z_n, (x_{n+1}, y), \tau)$ is monotonically increasing both in $y \in \mathbb{R}$ and in $\tau \in [0, 1]$ (where "monotonically increasing" is understood in the wide sense allowing for intervals of constancy). In other words, for each $\tau \in [0, 1]$, the function

$$y \in \mathbb{R} \mapsto Q(z_1, \ldots, z_n, (x_{n+1}, y), \tau)$$

is monotonically increasing, and for each $y \in \mathbb{R}$, the function

$$\tau \in [0, 1] \mapsto Q(z_1, \ldots, z_n, (x_{n+1}, y), \tau)$$

is monotonically increasing.

R1b For each training sequence $(z_1, \ldots, z_n) \in (\mathbb{R}^{p+1})^n$ and each test object $x_{n+1} \in \mathbb{R}^p$,

$$\lim_{y \to -\infty} Q(z_1, \ldots, z_n, (x_{n+1}, y), 0) = 0 \tag{4.2}$$

and

$$\lim_{y \to \infty} Q(z_1, \ldots, z_n, (x_{n+1}, y), 1) = 1.$$

R2 As the function of random training observations $z_1 \sim P$,..., $z_n \sim P$, a random test observation $z_{n+1} \sim P$, and a random number $\tau \sim U$, all assumed independent, the distribution of Q is uniform:

$$\forall \alpha \in [0, 1] : \mathsf{P}\{Q(z_1, \ldots, z_n, z_{n+1}, \tau) \le \alpha\} = \alpha.$$

The output of the randomized predictive system Q on a training sequence $z_1, \ldots, z_n$ and a test object x_{n+1} is the function

$$Q_n : (y, \tau) \in \mathbb{R} \times [0, 1] \mapsto Q(z_1, \ldots, z_n, (x_{n+1}, y), \tau), \tag{4.3}$$

which will be called the *randomized predictive distribution* (*function*) (**RPD**). The *thickness* of an RPD Q_n is the infimum of the numbers $\epsilon \geq 0$ such that the diameter

$$Q_n(y, 1) - Q_n(y, 0) \tag{4.4}$$

of the set

$$\{Q_n(y, \tau) \mid \tau \in [0, 1]\} \tag{4.5}$$

is at most ϵ for all $y \in \mathbb{R}$ except for finitely many values. The *exception size* of Q_n is the cardinality of the set of y for which the diameter (4.4) exceeds the thickness of Q_n. Notice that *a priori* the exception size can be infinite [146]. Of primary interest are RPDs of thickness $\frac{1}{n+1}$ everywhere but at most n points on axis y, where n is the size of the training set with $Q(z_1, \ldots, z_n, z_{n+1}, \tau)$ being a continuous function of τ. The set (4.5) will therefore be a closed interval in $[0, 1]$.

Four examples of predictive distributions are shown in Figure 4.1 below as shaded areas. The length of the training sequence for these plots is $n = 10$ (see Section 4.9 for details). The plots are examples of instance of Q_{10}, such instance has the width $1/11$ everywhere but 10 points on axis y. The width is given by the width of the interval $[Q(y, 0), Q(y, 1)]$, where $Q := Q_{10}$.

In conformal prediction, the starting point is the definition and choice of the conformity measure, we start by defining the conformity measure.

Definition 4.4.2 (Conformity measure)**.** The conformity measure is a measurable function A : $(\mathbb{R}^{p+1})^{n+1} \to \mathbb{R}$ that is invariant with respect to permutations of the first n observations: for any sequence $(z_1, \ldots, z_n) \in (\mathbb{R}^{p+1})^n$, any $z_{n+1} \in \mathbb{R}^{p+1}$, and any permutation π of $\{1, \ldots, n\}$,

$$A(z_1, \ldots, z_n, z_{n+1}) = A\left(z_{\pi(1)}, \ldots, z_{\pi(n)}, z_{n+1}\right).$$

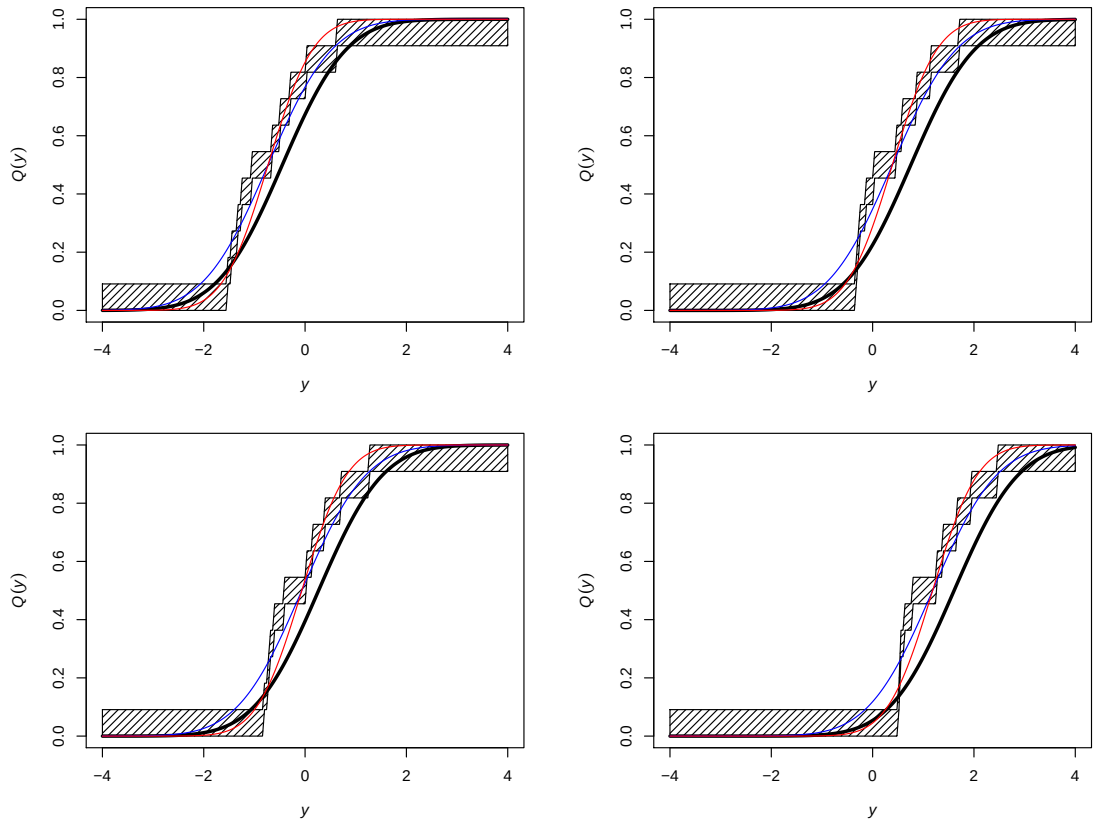

FIGURE 4.1: Examples of true predictive distribution functions (black), their conformal estimates (represented by the shaded areas), and the distribution functions output by simplified Oracle I (red) and Oracle II (blue) for a tiny training sequence (of length 10 with two attributes, the first one being the dummy all-1 attribute); in black and white, the true predictive distribution functions are the thick lines, and Oracle I is always farther from them in the uniform metric than Oracle II is

Function A measures *how large* the label y_{n+1} in z_{n+1} is, based on seeing the observations $z_1, \dots, z_n$ and the object x_{n+1} of z_{n+1}. A simple example of conformity measure is

$$A(z_1, \dots, z_{n+1}) := y_{n+1} - \hat{y}_{n+1}, \tag{4.6}$$

Where $\hat{y}_{n+1}$ is the prediction for y_{n+1} computed from x_{n+1} and $z_1, \dots, z_n$ as training sequence.

Definition 4.4.3 (Conformity score)**.** The conformity score α_i^y is defined by

$$\begin{aligned} \alpha_i^y &:= A(z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_n, (x_{n+1}, y), z_i), \qquad i = 1, \ldots, n, \\ \alpha_{n+1}^y &:= A(z_1, \ldots, z_n, (x_{n+1}, y)). \end{aligned} \tag{4.7}$$

Having given the definitions of conformity measure 4.4.2 and conformity score 4.4.3 we can proceed to define the conformal transducer as follows:

Definition 4.4.4 (Conformal transducer)**.** The conformal transducer determined by a conformity measure A is

$$\begin{aligned} Q(z_1, \ldots, z_n, (x_{n+1}, y), \tau) := \frac{1}{n+1} \left|\left\{i = 1, \ldots, n+1 \mid \alpha_i^y < \alpha_{n+1}^y\right\}\right| \\ + \frac{\tau}{n+1} \left|\left\{i = 1, \ldots, n+1 \mid \alpha_i^y = \alpha_{n+1}^y\right\}\right|, \end{aligned} \tag{4.8}$$

A function is a conformal transducer if it is the conformal transducer determined by some conformity measure [143].

Definition 4.4.5 (Conformal predictive system)**.** Conformal predictive system (**CPS**) is a function which is both a conformal transducer and a randomized predictive system.

Definition 4.4.6 (Conformal predictive distribution)**.** Conformal predictive distribution (**CPD**) is a function Q_n defined by (4.3) for a conformal predictive system Q.

Definition 4.4.7 (Conformal predictor)**.** Any conformal transducer Q and Borel set $A \subseteq [0, 1]$ defines the Conformal predictor:

$$\Gamma^A(z_1, \ldots, z_n, x_{n+1}, \tau) := \{y \in \mathbb{R} \mid Q(z_1, \ldots, z_n, (x_{n+1}, y), \tau) \in A\}.$$

Conformal transducers possess the standard property of validity - the values Q are distributed uniformly on $[0, 1]$ under the assumption that $z_1, \ldots, z_{n+1}$ are IID and τ is generated from the uniform probability distribution U on $[0, 1]$ and independently of $z_1, \ldots, z_{n+1}$ from the uniform probability distribution U on $[0, 1]$ (see [137, Proposition 2.8]). This property of conformal transducers is equivalent to the requirement **R2** in the definition 4.4.1 of a randomized predictive system (RPS) and ensures validity.

In conformal prediction, the customary interpretation of (4.8) is that it is a randomized p-value for testing the null hypothesis of the observations being IID. In the case of conformal predictive distributions the situation is slightly different — the informal alternative $H1$ hypothesis is that $y_{n+1} = y$ is smaller than expected under the IID model. Under such alternative hypothesis the (4.8) can be interpreted as a degree of conformity of the observation (x_{n+1}, y_{n+1}) to the remaining observations. One should note, that with the one-sided nature of this notion of conformity a label y can only be non-conforming if it is too small. A large label is never non-conforming ("strange"). Whilst such notion of conformity using a label can only be strange (non-conforming) if it is too small, large value of label is never strange. This notion of conformity specified by conformity measure (4.10) is somewhat counterintuitive, and will only be used as a technical tool.

4.5 Monotonic conformity measures

To understand properties of conformal predictive distributions, we first explore *monotonic* conformity measures that are defined as follows:

Definition 4.5.1 (Monotonic conformity measures)**.** A conformity measure A is *monotonic* if $A(z_1, \ldots, z_{n+1})$ if it is either monotonically increasing in y_{n+1},

$$y_{n+1} \leq y'_{n+1} \Longrightarrow A(z_1, \ldots, z_n, (x_{n+1}, y_{n+1})) \leq A(z_1, \ldots, z_n, (x_{n+1}, y'_{n+1}));$$

or monotonically decreasing in y_1,

$$y_{n+1} \leq y'_{n+1} \Longrightarrow A(z_1, \ldots, z_n, (x_{n+1}, y_{n+1})) \geq A(z_1, \ldots, z_n, (x_{n+1}, y'_{n+1})); \tag{4.9}$$

Conformal predictive distributions defined using monotonic conformity measures satisfy condition **R1a** of the definition 4.4.1 by Lemma 1 below. An simple example of a monotonic conformity measure is:

$$A(z_1, \ldots, z_m, (x, y)) := y - \hat{y} \tag{4.10}$$

Where $\hat{y}_{n+1}$ is produced by the K-nearest neighbours regression algorithm with value of $\hat{y}_{n+1}$ calculated as the average value of labels of the K nearest neighbours of x_{n+1} (here $y_{(1)}, \ldots, y_{(n)}$ is the sequence $y_1, \ldots, y_n$ sorted in the order of increasing distances between x_i and x_{n+1}):

$$\hat{y}_{n+1} := \frac{1}{K} \sum_{k=1}^{K} y_{(k)} \tag{4.11}$$

In the case of $\hat{y}_{n+1}$ defined by 4.11, the conformity measure is not only monotonically increasing but satisfies additionally the following condition:

$$\lim_{y \to \pm\infty} A(z_1, \ldots, z_n, (x_n, y)) = \pm\infty$$

The conformal transducer defined using conformity measure (4.10) where $\hat{y}_{n+1}$ is defined using 4.11 therefore also satisfies condition **R1b** of the definition and so is both a randomized predictive system (RPS) and a conformal predictive system (CPS).

Criterion of being a CPS

Unfortunately, many important conformity measures are not monotonic, and the next lemma introduces a weaker sufficient condition for a conformal transducer to be an RPS.

Lemma 1. *The conformal transducer determined by a conformity measure* ***A*** *satisfies condition* ***R1a*** *if, for each* $i \in \{1, \ldots, n\}$*, each training sequence* $(z_1, \ldots, z_n) \in (\mathbb{R}^{p+1})^n$*, and each test object* $x_{n+1} \in \mathbb{R}^p$*,* $\alpha^y_{n+1} - \alpha^y_i$ *is a monotonically increasing function of* $y \in \mathbb{R}$ *(in the notation of* (4.7)*).*

We can strengthen the conclusion of the lemma to the conformal transducer determined by A being an RPS (and, therefore, a CPS) if, e.g.,

$$\lim_{y \to \pm\infty} \left(\alpha^y_{n+1} - \alpha^y_i\right) = \pm\infty.$$

4.6 Least Squares Prediction Machine

In this section the Least Squares Prediction Machine (**LSPM**) is introduced. The LSPM is similiar to the Ridge Regression Confidence Machine (**RRCM**) described in [137, Section 2.3], with the key distinction that it produces predictive distribution functions rather than prediction intervals.

Definition 4.6.1 (Ordinary LSPM)**.** We define the *ordinary LSPM* as the conformal transducer with the conformity measure

$$A(z_1, \ldots, z_{n+1}) := y_{n+1} - \widehat{y}_{n+1} \tag{4.12}$$

Where y_{n+1} and $\widehat{y}_{n+1}$ are the label and the prediction for y_{n+1} accordingly. The prediction $\widehat{y}_{n+1}$ is computed from the training sequence $z_1, \ldots, z_{n+1}$ using the linear regression. The important distinction of the LSPM from the ordinary linear regression is that in the ordinary LSPM the training sequence $z_1, \ldots, z_{n+1}$ includes z_{n+1}.

The residual (4.12) in the definition 4.6.1 of the ordinary LSPM is the ordinary regression residual, however similar to how residuals are considered in statistics we consider three kinds of LSPM — *ordinary LSPM*, *deleted LSPM* and *studentized LSPM*.

Definition 4.6.2 (Deleted LSPM)**.** The deleted LSPM is defined using the conformity measure:

$$A(z_1, \ldots, z_{n+1}) := y_{n+1} - \hat{y}_{n+1}, \tag{4.13}$$

The deleted LSPM differs from the ordinary LSPM in that $\widehat{y}_{n+1}$ is replaced by the prediction $\hat{y}_{n+1}$ for y_{n+1} computed using the ordinary linear regression (OLS) from x_{n+1} and $z_1, \ldots, z_n$ as training sequence.

Unlike the definition 4.6.1 of ordinary LSPM in the definition 4.6.2 of deleted LSPM the training sequence does not include z_{n+1}. The *studentized LSPM* is somewhat midway between ordinary and deleted LSPM and is defined in 4.6.4. Both the ordinary and deleted LSPM are not RPS, as their output Q_n (see (4.3)) is not necessarily monotonically increasing in y as required in **R1a** of the definition 4.4.1. Whilst this is unfortunate, below we will show that this can happen only in the presence of high-leverage points.

4.6.1 High leverage points

We denote by $\bar{X}$ the data matrix that has dimensions of $(n+1) \times p$, each row of the matrix $\bar{X}$ represents the ith object (note that the data matrix $\bar{X}$ includes both the training set and the test object as it has $n+1$ rows.

Definition 4.6.3 (The hat matrix)**.** The hat matrix for the $n+1$ observations $z_1, \ldots, z_{n+1}$ is defined as:

$$\bar{H} = \bar{X}(\bar{X}'\bar{X})^{-1}\bar{X}'. \tag{4.14}$$

We use $\bar{h}_{i,j}$ to denote the elements of this hat matrix where i refers to the ith row and j refers to the jth column. For the diagonal elements of the hat matrix we use the shorthand expression $\bar{h}_i$. The following proposition can be derived from Lemma 1 in the explicit form (analogous to Algorithm 1 below but using (4.23)) of the ordinary LSPM.

Proposition 1. *The function Q_n that is output by the ordinary LSPM (see (4.3)) is monotonically increasing in y provided that $\bar{h}_{n+1} < 0.5$.*

The necessary condition for the Q_n to be monotonically increasing, $\bar{h}_{n+1} < 0.5$ defines test object x_{n+1} as *not a very influential point* following the terminology in Chatterjee and Hadi[18, Section 4.2.3.1]. The assumption of the test object x_{n+1} being not a very influential point ($\bar{h}_{n+1} < 0.5$) in Proposition 1 turns out to be essential:

Proposition 2. *Proposition 1 ceases to be true if the constant 0.5 in it is replaced by a larger constant.*

The next two propositions demonstrate that for the case of deleted LSPM, defined by (4.6.2), even stricter condition is required than for the ordinary LSPM: $\bar{h}_i < 0.5$ for all $i = 1, \ldots, n$.

Proposition 3. *The function Q_n that is output by the deleted LSPM according to (4.3) is monotonically increasing in y provided that $\max_{i=1,\ldots,n} \bar{h}_i < 0.5$.*

For the deleted LSPM the following analogue of Proposition 2 is as follows.

Proposition 4. *Proposition 3 ceases to be true if the constant 0.5 in it is replaced by a larger constant.*

Studentized LSPM

From the point of view of predictive distributions, the best choice is therefore studentized LSPM.

Definition 4.6.4 (Studentized LSPM)**.** The studentized LSPM is defined using the conformity measure:

$$A(z_1, \ldots, z_{n+1}) = \frac{y_{n+1} - \widehat{y}_{n+1}}{\sqrt{1 - \bar{h}_{n+1}}}$$

As deleted residuals $y_i - \hat{y}_i$ can be represented as $(y_i - \widehat{y}_i)/(1 - \bar{h}_i)$, where $\hat{y}_i$ is the prediction for y_i computed using $z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_{n+1}$ as training sequence, the studentized LSPM is in a way a form of an intermediate residual between those residuals in the definitions of the ordinary and deleted LSPM. An important advantage of studentized LSPM is that no additional assumptions of low leverage are required to satisfy the requirements of them being predictive distributions. Accordingly, the following proposition holds:

Proposition 5. *The studentized LSPM is an RPS and, therefore, a CPS.*

The studentized LSPM in an explicit form

Algorithms 1 and 2 derive two explicit forms for the studentized LSPM (Algorithms 1 and 2). The versions for the ordinary and deleted LSPM are similar, the explicit form is given only for the ordinary version of the LSPM and is particularly intuitive. Predictive distributions (4.3) are represented in the form:

$$Q_n(y) := [Q_n(y, 0), Q_n(y, 1)]$$

The function Q_n maps each potential label $y \in \mathbb{R}$ to a closed interval of $\mathbb{R}$. In the ordinary Least Squares regression, the vector $(\widehat{y}_1, \ldots, \widehat{y}_{n+1})'$ of predictions can be represented as the product of the hat matrix $\bar{H}$ and the vector $(y_1, \ldots, y_{n+1})'$ of labels. Therefore, we can represent the studentized residuals as:

$$\alpha^y_{n+1} - \alpha^y_i = B_i y - A_i, \quad i = 1, \ldots, n, \tag{4.15}$$

Algorithm 1 Least Squares Prediction Machine

Require: A training sequence $(x_i, y_i) \in \mathbb{R}^p \times \mathbb{R}$, $i = 1, \ldots, n$.
Require: A test object $x_{n+1} \in \mathbb{R}^p$.
1: Set $\bar{X}$ to the data matrix for the given $n+1$ objects.
2: Define the hat matrix $\bar{H}$ by (4.14).
3: **for** $i \in \{1, 2, \ldots, n\}$ **do**
4: Define A_i and B_i by (4.17) and (4.16), respectively.
5: Set $C_i := A_i / B_i$.
6: **end for**
7: Sort $C_1, \ldots, C_n$ in the increasing order obtaining $C_{(1)} \leq \cdots \leq C_{(n)}$.
8: Return the predictive distribution (4.18) for y_{n+1}.

Where, following the notation of (4.7), y is the label of the $(n+1)$st object x_{n+1} and

$$B_i := \sqrt{1 - \bar{h}_{n+1}} + \frac{\bar{h}_{i,n+1}}{\sqrt{1 - \bar{h}_i}}, \tag{4.16}$$

$$A_i = \frac{\sum_{j=1}^n \bar{h}_{j,n+1} y_j}{\sqrt{1 - \bar{h}_{n+1}}} + \frac{y_i - \sum_{j=1}^n \bar{h}_{i,j} y_j}{\sqrt{1 - \bar{h}_i}} \tag{4.17}$$

All the B_i are assumed to be defined and positive. We further set $C_i := A_i / B_i$ for all $i = 1, \ldots, n$ and then sort all C_i in the increasing order to obtain the sequence be $C_{(1)} \leq \cdots \leq C_{(n)}$. Let $C_{(0)} := -\infty$ and $C_{(n+1)} := \infty$. The predictive distribution is computed as:

$$Q_n(y) := \begin{cases} [\frac{i}{n+1}, \frac{i+1}{n+1}] & \text{if } y \in (C_{(i)}, C_{(i+1)}) \text{ for } i \in \{0, 1, \ldots, n\} \\ [\frac{i'-1}{n+1}, \frac{i''+1}{n+1}] & \text{if } y = C_{(i)} \text{ for } i \in \{1, \ldots, n\}, \end{cases} \tag{4.18}$$

where $i' := \min\{j \mid C_{(j)} = C_{(i)}\}$ and $i'' := \max\{j \mid C_{(j)} = C_{(i)}\}$. From 4.18 it is clear that the thickness of this CPD is $\frac{1}{n+1}$ with the exception size equal to the number of distinct C_i, at most n. The overall computation for the Least Square Prediction Machine (LSPM) is described in the Algorithm 1, where the data matrix $\bar{X}$ has x_i', $i = 1, \ldots, n+1$, as its ith row, the data matrix dimensions are $(n+1) \times p$.

The assumed condition that all B_i are defined and positive, $i = 1, \ldots, n$ is satisfied by using the result from [18, Property 2.6(b)]: $\bar{h}_{n+1} = 1$ implies that $\bar{h}_{i,n+1} = 0$ for $i = 1, \ldots, n$. Henceforth this condition is equivalent to $\bar{h}_i < 1$ for all $i = 1, \ldots, n+1$. By [93, Lemma 2.1(iii)], this means that the rank of the extended data matrix $\bar{X}$ is p and it remains p after removal of any one of its $n+1$ rows. If this condition is not satisfied, we

define $Q_n(y) := [0, 1]$ for all y. This ensures that the studentized LSPM is a CPS.

The batch version of the studentized LSPM

When test set consists of a sequence of objects $x_{n+1}, \ldots, x_{n+m}$ (instead of just one test object x_{n+1}), a much more efficient implementation of the LSPM can be designed by pre-computing the hat matrix for the training objects $x_1, \ldots, x_n$, and then updating computations for each test object x_{n+j} based on the results from Sherman-Morrison-Woodbury theorem: see, e.g., Chatterjee and Hadi [18, p. 23, (2.18)–(2.18c)].

The computational update proceeds as follows. Let's set

$$g_i := x_i'(X'X)^{-1}x_{n+1}, \quad i = 1, \ldots, n+1. \tag{4.19}$$

Where X is the data matrix for the the first n observations (matrix dimension is $n \times p$, denote its ith row as x_i', $i = 1, \ldots, n$. Finally, let H be the corresponding hat matrix ($n \times n$) for the first n objects:

$$H := X(X'X)^{-1}X' \tag{4.20}$$

With elements denoted $h_{i,j}$ and $h_{i,i}$ sometimes abbreviated to h_i. The full hat matrix $\bar{H}$ is therefore larger than H, with the extra entries computed as follows:

$$\bar{h}_{i,n+1} = \bar{h}_{n+1,i} = \frac{g_i}{1 + g_{n+1}}, \quad i = 1, \ldots, n+1. \tag{4.21}$$

The remaining entries of $\bar{H}$ are

$$\bar{h}_{i,j} = h_{i,j} - \frac{g_i g_j}{1 + g_{n+1}}, \quad i, j = 1, \ldots, n. \tag{4.22}$$

The overall algorithm for the Least Squares Prediction Machine (batch version) is summarized as Algorithm 2.

Algorithm 2 Least Squares Prediction Machine (batch version)

Require: A training sequence $(x_i, y_i) \in \mathbb{R}^p \times \mathbb{R}$, $i = 1, \ldots, n$.
Require: A test sequence $x_{n+j} \in \mathbb{R}^p$, $j = 1, \ldots, m$.
Set X to the data matrix for the n training objects.
Set $H = (h_{i,j})$ to the hat matrix (4.20).
for $j \in \{1, 2, \ldots, m\}$ **do**
    Set $x_{n+1} := x_{n+j}$.
    Define an $(n+1) \times (n+1)$ matrix $\bar{H} = (\bar{h}_{i,j})$ by (4.21) and (4.22).
    for $i \in \{1, 2, \ldots, n\}$ **do**
        Define A_i and B_i by (4.17) and (4.16), respectively.
        Set $C_i := A_i / B_i$.
    end for
    Sort $C_1, \ldots, C_n$ in the increasing order obtaining $C_{(1)} \le \cdots \le C_{(n)}$.
    Return the predictive distribution (4.18) for the label of x_{n+j}.
end for

The ordinary LSPM

A similar calculation demonstrates that the ordinary LSPM has an intuitive and efficient representation (see, e.g., [12, Appendix A]):

$$C_i = \frac{A_i}{B_i} = \hat{y}_{n+1} + (y_i - \hat{y}_i)\frac{1 + g_{n+1}}{1 + g_i}, \tag{4.23}$$

The Least Squares predictions for y_{n+1} and y_i are $\hat{y}_{n+1}$ and $\hat{y}_i$, that are both computed from the test objects x_{n+1} and x_i, respectively and the observations $z_1, \ldots, z_n$ as the training sequence.

The predictive distribution is defined by formula (4.18). the fraction $\frac{1+g_{n+1}}{1+g_i}$ in (4.23) is typically and asymptotically (at least under the assumptions A1–A4 stated in the next section) close to 1, and can usually be ignored. The two other versions of the LSPM also typically have

$$C_i \approx \hat{y}_{n+1} + (y_i - \hat{y}_i). \tag{4.24}$$

4.7 Validity of the LSPM in the online mode

Algorithm 1) outlined a procedure to compute "fuzzy" distribution function Q_n based on a training sequence $z_i = (x_i, y_i)$, $i = 1, \ldots, n$, and a test object x_{n+1}. We use notation

$Q_n(y)$ to denote an interval and $Q_n(y,\tau)$ to denote a point inside that interval, as explained previously.

In the online prediction mode, the computations proceed as follows:

Protocol 4.7.1. ONLINE MODE OF PREDICTION

Nature generates an observation $z_1 = (x_1, y_1)$ from a probability distribution P;

for $n = 1, 2, \ldots$ **do**

Nature independently generates a new observation $z_{n+1} = (x_{n+1}, y_{n+1})$ from P;

Forecaster announces Q_n, a predictive distribution based on $(z_1, \ldots, z_n)$ and x_{n+1};

set $p_n := Q_n(y_{n+1}, \tau_n)$, where $\tau_n \sim U$

end for

The "ground truth" distribution P and y_{n+1} are not available to the Forecaster when computing Q_n. We adapt condition R2 for the online mode by strengthening it as follows:

Theorem 1 ([137], Theorem 8.1)**.** *In the online mode of prediction (in which $(z_i, \tau_i) \sim P \times U$ are IID), the sequence $(p_1, p_2, \ldots)$ is IID and $(p_1, p_2, \ldots) \sim U^\infty$, provided that Forecaster uses the studentized LSPM (or any other conformal transducer).*

The property of validity asserted in Theorem 1 is marginal, in that we do not assert that the distribution of p_n is uniform conditionally on x_{n+1}. Conditional validity is attained by the LSPM only asymptotically and under additional assumptions, as is demonstrated in the the next section.

4.8 Asymptotic efficiency

In this section we outline some basic results about the LSPM's efficiency. When standard IID assumtpion holds, the nonparametric LSPM has a property of validity. It is natural to pose the question about the cost of validity (in terms of potential loss in efficiency) in situation when data distribution is parametric or even in ideal situation when Bayesian assumptions also hold [146]. Such question was asked independently by Evgeny Burnaev [13] and Larry Wasserman [79].

For our study of efficiency we will use the assumptions that are not in line with the general IID model by assuming that strong parametric data generation process for the

labels y_i given the corresponding objects x_i. In addition, we will also remove the assumption of randomness for the objects $x_1, x_2, \ldots$ by instead allowing them to be fixed vector. As it turns out, the two main results of this section — Theorems 2 and 3 — do not depend on the assumptions of randomness and IID.

Given fixed objects, $x_1, x_2, \ldots$, we assume that the labels $y_1, y_2, \ldots$ are generated using the linear parametric model, with w being a vector parameter ($w \in \mathbb{R}^p$) and ξ_i being IID normally distributed ($N(0, \sigma^2)$)random variables:

$$y_i = w' x_i + \xi_i, \tag{4.25}$$

This linear parametric model contains two parameters: vector w and positive number σ. We form the training set from the first n elements of an otherwise infinite sequence of observations $(x_1, y_1), (x_2, y_2), \ldots$. To study efficiency asymptotically, we then let $n \to \infty$.

To summarise all the assumptions required for the efficiency results:

A1 The sequence $x_1, x_2, \ldots$ is bounded: $\sup_i \|x_i\| < \infty$.

A2 The first component of each vector x_i is 1.

A3 The empirical second-moment matrix has its smallest eigenvalue eventually bounded away from 0:

$$\liminf_{n\to\infty} \lambda_{\min}\left(\frac{1}{n}\sum_{i=1}^{n} x_i x_i'\right) > 0,$$

where $\lambda_{\min}$ stands for the smallest eigenvalue.

A4 The labels $y_1, y_2, \ldots$ are generated according to the linear model (4.25): $y_i = w' x_i + \xi_i$, where ξ_i are independent Gaussian noise random variables distributed as $N(0, \sigma^2)$.

In our study of efficiency, we consider the three versions of the LSPM and also three "oracles". All of the three oracles know the data generation model (4.25). In addition Oracle III has complete information about both parameters of the data generation model (4.25) (both w and σ), Oracle II knows σ, but does not know w and finally Oracle I knows neither w nor σ. Oracles that don't know certain parameters have to estimate them from the data.

Proper Oracle I does not know both parameters and hence computes both w and σ using Ordinary Least Squares procedures to obtain the standard predictive distribution for the label y_{n+1} of the test object x_{n+1} based on the training sequence of the first n observations and x_{n+1}:

$$\hat{y}_{n+1} + \sqrt{1 + g_{n+1}}\hat{\sigma}_n t_{n-p}, \tag{4.26}$$

where g_{n+1} is defined in (4.19),

$$\hat{y}_{n+1} := x'_{n+1}(X'X)^{-1}X'Y,$$
$$\hat{\sigma}_n := \sqrt{\frac{1}{n-p}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}, \quad \hat{y}_i := x'_i(X'X)^{-1}X'Y,$$

X is the data matrix for the training sequence (the $n \times p$ matrix whose ith row is x'_i, $i = 1, \ldots, n$), Y is the vector $(y_1, \ldots, y_n)'$ of the training labels, and t_{n-p} is Student's t-distribution with $n - p$ degrees of freedom; see, e.g., [118, Section 5.3.1] or [148, Example 3.3].

The standard prediction distribution obtained by Oracle I 4.26 is slightly different from its simplified version that is more popular in the literature on empirical processes for residuals where *simplified Oracle I* outputs:

$$N\left(\hat{y}_{n+1}, \hat{\sigma}_n^2\right). \tag{4.27}$$

The difference between the two versions is, however, asymptotically negligible [108], and the results stated below will be applicable to both versions.

Proper Oracle II that has access to information about σ, but not w outputs the predictive distribution:

$$N\left(\hat{y}_{n+1}, (1 + g_{n+1})\sigma^2\right), \tag{4.28}$$

a simplified version of which (*simplified Oracle II*) is:

$$N\left(\hat{y}_{n+1}, \sigma^2\right). \tag{4.29}$$

Similar to the case of *Proper Oracle II*, the difference between the two versions of Oracle II computations (full and simplified) is again asymptotically negligible under our assumptions 4.8.

Finally, *Oracle III* outputs the predictive distribution:

$$N\left(w'x_{n+1}, \sigma^2\right).$$

We use our notation for the conformal predictive distribution Q_n (4.3), as before, Q_n^{I} denotes simplified or proper Oracle I's predictive distribution, (4.27) or (4.26) (Theorem 2 will hold for both), and Q_n^{II} to denote simplified or proper Oracle II's predictive distribution, (4.29) or (4.28) (Theorem 3 will hold for both). Theorems 2 and 3 are accordingly applicable to all three versions of the LSPM.

Theorem 2. *The random functions $G_n : \mathbb{R} \to \mathbb{R}$ defined by*

$$G_n(t) := \sqrt{n}\left(Q_n(\hat{y}_{n+1} + \hat{\sigma}_n t, \tau) - Q_n^{\mathrm{I}}(\hat{y}_{n+1} + \hat{\sigma}_n t)\right)$$

weakly converge to a Gaussian process Z with mean zero and covariance function

$$\operatorname{cov}(Z(s), Z(t)) = \Phi(s)\left(1 - \Phi(t)\right) - \phi(s)\phi(t) - \frac{1}{2}st\phi(s)\phi(t), \quad s \le t.$$

Theorem 3. *The random functions $G_n : \mathbb{R} \to \mathbb{R}$ defined by*

$$G_n(t) := \sqrt{n}\left(Q_n(\hat{y}_{n+1} + \sigma t, \tau) - Q_n^{\mathrm{II}}(\hat{y}_{n+1} + \sigma t)\right)$$

weakly converge to a Gaussian process Z with mean zero and covariance function

$$\operatorname{cov}(Z(s), Z(t)) = \Phi(s)\left(1 - \Phi(t)\right) - \phi(s)\phi(t), \quad s \le t.$$

In Theorems 2 and 3, we have $\tau \sim U$; alternatively, they will remain true if we fix τ to any value in $[0, 1]$. For simplified oracles,we have $Q_n^{\mathrm{I}}(\hat{y}_{n+1} + \hat{\sigma}_n t) = \Phi(t)$in Theorem 2 and $Q_n^{\mathrm{II}}(\hat{y}_{n+1} + \sigma t) = \Phi(t)$ in Theorem 3. For proofs of Theorem 2 and Theorem 3, see [146].

Applying Theorems 2 and 3 to a fixed argument t, we obtain (dropping τ altogether):

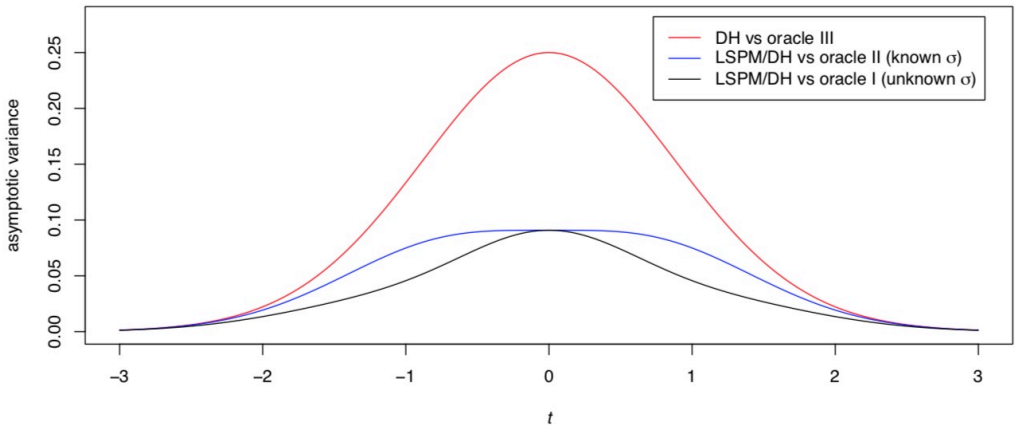

FIGURE 4.2: The asymptotic variances for the Dempster-Hill (DH) procedure as compared with the truth (Oracle III, red) and for the LSPM and DH procedure as compared with the oracular procedures for known σ (Oracle II, blue) and unknown σ (Oracle I, black); in black and white, red is highest, blue is intermediate, and black is lowest

Corollary 1. *For a fixed $t \in \mathbb{R}$,*

$$\sqrt{n}\left(Q_n(\hat{y}_{n+1} + \hat{\sigma}_n t) - Q_n^{\mathrm{I}}(\hat{y}_{n+1} + \hat{\sigma}_n t)\right) \Rightarrow N\left(0, \Phi(t)(1-\Phi(t)) - \phi(t)^2 - \frac{1}{2}t^2\phi(t)^2\right)$$

and

$$\sqrt{n}\left(Q_n(\hat{y}_{n+1} + \sigma t) - Q_n^{\mathrm{II}}(\hat{y}_{n+1} + \sigma t)\right) \Rightarrow N\left(0, \Phi(t)(1-\Phi(t)) - \phi(t)^2\right).$$

Figure 4.2 presents plots for the asymptotic variances, given in Corollary 1, for the two oracular predictive distributions: black for Oracle I ($\Phi(t)(1-\Phi(t)) - \phi(t)^2 - \frac{1}{2}t^2\phi(t)^2$ vs t) and blue for Oracle II ($\Phi(t)(1-\Phi(t)) - \phi(t)^2$ vs t); the red plot will be discussed later in this section. The two asymptotic variances coincide at $t = 0$, where they attain their maximum of between 0.0908 and 0.0909.

We can conclude that for data generated using the Gaussian model (4.25) under natural assumptions,the LSPM is asymptotically close to the predictive distributions generated by Oracles I and II. The LSPM is therefore approximately conditionally valid and efficient (i.e., valid and efficient given $x_1, x_2, \ldots$). On the other hand, Theorem 1 guarantees the

marginal validity of the LSPM under the general IID model, regardless of whether (4.25) holds.

Comparison with the Dempster-Hill procedure

This subsection outlines a classical procedure which we refer to as the *Dempster-Hill procedure* as it was most clearly articulated in [30, p. 110] and [50, 49]. Both Dempster and Hill trace their ideas to nonparametric version of Fisher's fiducial method ([38]; [37]). Fisher, however, was mostly interested in confidence distributions for quantiles, rather than full predictive distributions. Hill [49] referred to his procedure as Bayesian nonparametric predictive inference, which was later abbreviated to nonparametric predictive inference (NPI) by Frank Coolen [3]. An important predecessor of Dempster and Hill was Jeffreys [55], who postulated what was later denoted by Hill as $A_{(2)}$ ([70] and [119] contain discussions of Jeffreys's paper and Fisher's reaction).

Definition 4.8.1. The *Dempster-Hill* procedure is the conformal predictive system determined by the conformity measure

$$A(z_1, \dots, z_{n+1}) = A(y_1, \dots, y_{n+1}) = y_{n+1}; \tag{4.30}$$

Such procedure is used when the objects x_i are absent and it can be regarded as the special case of the LSPM when there are no objects ($p = 0$) — alternatively one can consider the situation when $p = 1$ but assume that all objects are $x_i = 0$. In this case, the predictions $\hat{y}$ will always be 0 and the hat matrices are $\bar{H} = 0$ and $H = 0$, this means that (4.12), (4.13), and (4.15) all reduce to (4.30).

In the absence of ties, such predictive distribution becomes:

$$Q_n(y) := \begin{cases} [\frac{i}{n+1}, \frac{i+1}{n+1}] & \text{if } y \in (y_{(i)}, y_{(i+1)}) \text{ for } i \in \{0, 1, \dots, n\} \\ [\frac{i-1}{n+1}, \frac{i+1}{n+1}] & \text{if } y = y_{(i)} \text{ for } i \in \{1, \dots, n\} \end{cases} \tag{4.31}$$

(cf. (4.18)), where $y_{(1)} \le \dots \le y_{(n)}$ are the y_i sorted in the increasing order, $y_{(0)} := -\infty$, and $y_{(n+1)} := \infty$.

This is essentially Hill's assumption $A_{(n)}$ (which he also denoted A_n); in his words: "A_n asserts that conditional upon the observations $X_1, \dots, X_n$, the next observation X_{n+1} is equally likely to fall in any of the open intervals between successive order statistics of the given sample" [50, Section 1]. The set of all continuous distribution functions F compatible with Hill's $A_{(n)}$ coincides with the set of all continuous distribution functions F satisfying $F(y) \in Q_n(y)$ for all $y \in \mathbb{R}$, where Q_n is defined by (4.31).

The LSPM, as presented in (4.24), is thus a very natural adaptation of Hill's $A_{(n)}$ to the Least Squares regression. The Dempster-Hill predictive system (4.31) is a conformal transducer under the condition that a point from an interval in (4.31) is chosen randomlyfrom the uniform distribution on that interval), resulting in the same guarantees of validity as those given above: the distribution of (4.31) is uniform over the interval $[0, 1]$.

In terms of efficiency, given the most standard case of IID Gaussian observations, our predictive distributions for linear regression are as precise as the Dempster-Hill ones asymptotically when compared with Oracles I and II.

Let us consider the Dempster-Hill procedure for the location/scale model $y_i = w + \xi_i$, $i = 1, 2, \dots$, where $\xi_i \sim N(0, \sigma^2)$ are independent. Similar to the case of the LSPM, the comparision if between the Dempster-Hill procedure and the three oracles:

In our study of efficiency, we consider the three versions of the LSPM and also three "oracles". All of the three oracles know the data generation model (4.25). In addition Oracle III has complete information about both parameters of the data generation model (4.25) (both w and σ), Oracle II knows σ, but does not know w and finally Oracle I knows neither w nor σ. Oracles that don't know certain parameters have to estimate them from the data. *Proper Oracle I* does not know both parameters, Oracle II knows σ, and Oracle III has complete information as it knows both w and σ.

It is interesting to note that Theorems 2 and 3 (and therefore the blue and black plots in Figure 4.2) are applicable to both the LSPM and Dempster-Hill predictive distributions (see, e.g., [107].) The situation with Oracle III is, however, different.

Donsker's ([31]) classical result allows to simplify Theorems 2 and 3 (Q^{III} denotes Oracle III's predictive distribution (independent of n).

Theorem 4. *In the case of the Dempster-Hill procedure, the random function $G_n : \mathbb{R} \to \mathbb{R}$ defined by*

$$G_n(t) := \sqrt{n}\left(Q_n(w + \sigma t, \tau) - Q^{\mathrm{III}}(w + \sigma t)\right) = \sqrt{n}\left(Q_n(w + \sigma t, \tau) - \Phi(t)\right) \tag{4.32}$$

weakly converges to a Brownian bridge, i.e., a Gaussian process Z with mean zero and covariance function

$$\operatorname{cov}(Z(s), Z(t)) = \Phi(s)\left(1 - \Phi(t)\right), \quad s \le t.$$

The variance $\Phi(t)(1 - \Phi(t))$ of the Brownian bridge is shown as the red line in Figure 4.2. However, under assumptions of this section including assumption of fixed objects, the analogue of the process (4.32) does not converge in general for the LSPM.

4.9 Experimental results

In this section we explore experimentally the validity and efficiency of the studentized LSPM.

Online validity

First, we verify experimental validity of our methods in the online mode of prediction. Such validity is guaranteed at the theoretical level, checking validity via experiments is a useful opportunity to test the correctness of our code implementation. We provide full details in order to enable reproducibility of the results.

We generate IID observations $z_n = (x_n, y_n)$, $n = 1, \dots, 1001$, the corresponding p-values $p_n := Q_n(y_{n+1}, \tau_n)$, $n = 1, \dots, N$, $N := 1000$, in the online mode. In our experiments, we generate objects x_n from standard normal distribution $x_n \sim N(0, 1)$ and labels y_n from linear model under Gaussian assumption as $y_n \sim 2x_n + N(0, 1)$. As usual, we consider τ_n to be uniformly distributed$\tau_n \sim U$ and all independent.

Figure 4.3 plots cumulative p-values $S_n := \sum_{i=1}^{n} p_i$ vs $n = 1, \dots, N$. As expected, it is an approximately straight line with slope 0.5. Figure 4.4 additionally presents three plots: the cumulative sums $S_n^{\alpha} := \sum_{i=1}^{n} \mathbf{1}_{\{p_i \le \alpha\}}$, where $\mathbf{1}$ is the indicator function, vs

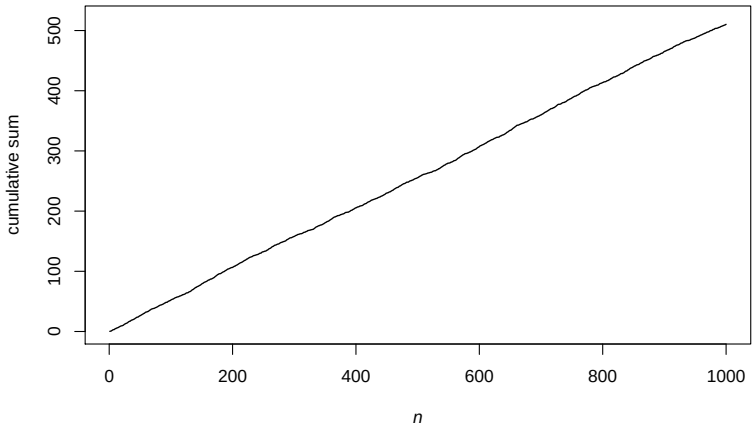

FIGURE 4.3: The cumulative sums S_n of the p-values vs $n = 1, \dots, 1000$

$n = 1, \dots, N$, for three values of α, $\alpha \in \{0.25, 0.5, 0.75\}$. For each of the three αs the result is an approximately straight line with slope α. Finally, Figure 4.5 plots A_N^α against $\alpha \in [0, 1]$, where $A_N^\alpha := \frac{1}{N} \sum_{i=1}^N \mathbf{1}_{\{p_i \le \alpha\}}$. The result is, approximately, the main diagonal of the square $[0, 1]^2$, as it should be.

Further, we explore empirically the efficiency of the studentized LSPM. Figure 4.1 compares the conformal predictive distribution with the true (Oracle III's) distribution for four randomly generated test objects and a randomly generated training sequence of length 10 with 2 attributes. The first attribute is a dummy all-1 attribute (Theorems 2 and 3 depend on the assumption that one of the attributes is an identical 1 as without it, the plots become qualitatively different: cf. [19, Corollary 2.4.1]).

The second attribute is generated from the standard Gaussian distribution, and the labels are generated as $y_n \sim 2x_{n,2} + N(0, 1)$, $x_{n,2}$ being the second attribute. Additionally We show (with thinner lines) the output of Oracle I and Oracle II (only for the simplified versions, in order not to clutter the plots). Instead, in the left-hand plot of Figure 4.6 we show the first plot of Figure 4.1 that is normalized by subtracting the true distribution function; this time, we show the output of both simplified and proper Oracles I and II; the difference is not large but noticeable. The right-hand plot of Figure 4.6 is similar except that the training sequence is of length 100 and there are 20 attributes generated independently from the standard Gaussian distribution except for the first one, which is the

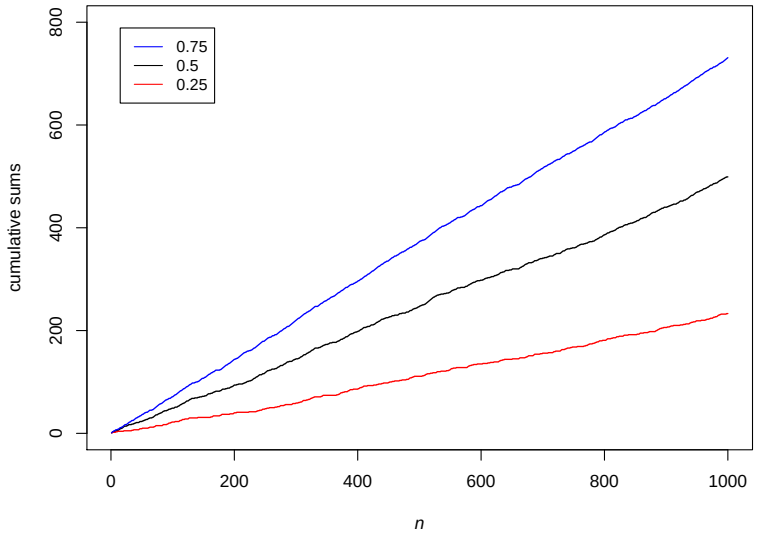

FIGURE 4.4: The cumulative sums S_n^α vs $n = 1, \dots, 1000$ for $\alpha \in \{0.25, 0.5.0.75\}$

dummy all-1 attribute; the labels are generated as before, $y_n \sim 2x_{n,2} + N(0,1)$.

Since Oracle III is more powerful than Oracles I and II as it knows the true data-generating distribution) it is more difficult to compete with. The black line is farther from the shaded area than the blue and red lines for all four plots in Figure 4.1. The estimated distribution functions being to the left of the true distribution functions is a coincidence: the four plots correspond to the values 0–3 of the seed for the R pseudorandom number generator, and for other seeds the estimated distribution functions are sometimes to the right and sometimes to the left.

4.10 Conclusion

The main contribution of this chapter is the introduction and development of non-parametric approach to predictive distribution functions using conformal prediction. In statistics, the theory of predictive distributions (see, e.g., [117], [121]) is based on the assumption that samples are generated from a parametric model. Our novel contribution (published paper [146]) extends statistical "prediction confidence distributions" in terms of ability to derive predictive distribution functions for machine learning regression tasks

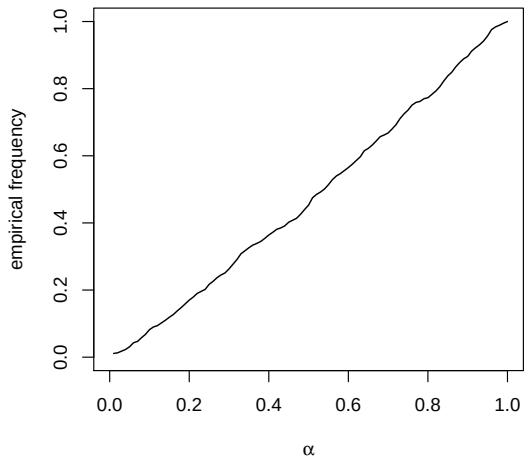

FIGURE 4.5: The calibration curve: A_N^α vs $\alpha \in [0, 1]$ for $N = 1000$

using only limited assumptions customary of machine learning, namely that the observations are generated using the IID model. Unlike in the parametric approach of statistical predictive distributions, our novel approach is non-parametric approach that does not require for the data model to be specified.

This approach results in predictive distribution functions that have requisite property of validity for IID observations in terms of guaranteed coverage. The advantage of predictive distribution functions over the usual conformal prediction intervals is that conformal predictive distributions contains more information — a conformal predictive distribution Q_n can produce a plethora of prediction intervals corresponding to each confidence level $1 - \epsilon$.

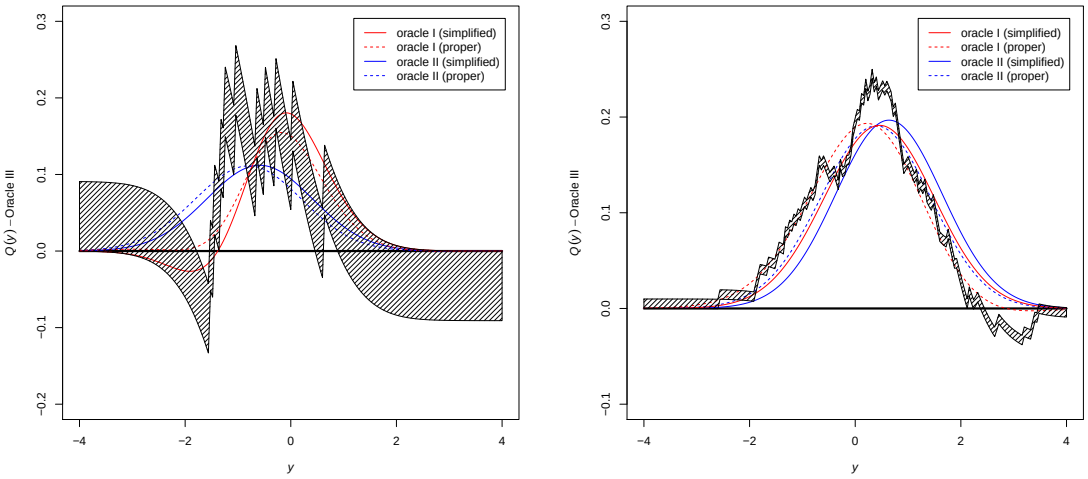

FIGURE 4.6: The left-hand plot is the first (upper left) plot of Figure 4.1 normalized by subtracting the true distribution function (the thick black line in Figure 4.1, which now coincides with the x-axis) and with the outputs of the proper oracles added; the right-hand plot is an analogous plot for a larger training sequence (of length 100 with 20 attributes, the first one being the dummy attribute)

Chapter 5

Kernel probabilistic regression

This chapter extends the study of probabilistic regression and combines conformal predictive distributions (CPDs) with kernel methods to derive kernelized versions of the algorithms described in Chapter 4. Kernelized versions of conformal predicitive distributions are studied theoretically to determine their computational efficiency. Experimental study of the predictive efficiency demonstrates important advantages of the kernelized versions of CPDs and shows that universal (Laplacian) kernel works remarkably well in terms of outputting accurate probabilistic predictions for the test objects.

5.1 Introduction

In this chapter, we continue to study probabilistic regression problem that began in Chapter 4 . As in Chapter 4, we require predictions to satisfy a reasonable property of validity (under natural assumptions standard for general machine learning problems). The natural probabilistic prediction for the label y is that of a complete probability measure on $\mathbb{R}$, such probability measure can be represented by its *distribution function* (see, e.g., [27, 28, 40]). Similar to definitions used in Chapter 4, this will be referred to as *predictive distribution* (PD). The necessary property of predictive distributions (same as for any probabilistic predictors) is that they are well-calibrated. We follow Gneiting *et al.* [40, Section 1.2] to formally define calibration as the"statistical compatibility between the probabilistic forecasts and the realizations." The less formal definition of validity is that predictive distributions should "tell the truth" (see Section 1.2.1 for more details on terminology in probabilistic prediction).

It can happen that the "truth" can be non-informative (as it should be in highly uncertain situations where predictive distribution should be flat rather than have the peak in the wrong place, as would happen with Bayesian prediction with an incorrect prior (see, e.g., [88]). The further requirement is that of efficiency, often referred to as sharpness [40, Section 2.3]. Our objective is to optimize the efficiency subject to the necessary assumption of validity [40, Section 1.2].

This chapter provides a very selective review of predictive distributions with validity guarantees. After outlining the prediction problem in Section 5.2, in Section 5.3 we turn to the oldest approach to predictive distributions — namely the Bayesian approach. Under a very restrictive assumption of full knowledge of the stochastic data generating mechanism such an approach gives rise to a perfect solution. Section 5.4 proceeds to Fisher's fiducial predictive distributions.

The first recent development, as described in Chapter 4, was to carry over predictive distributions to the framework of statistical machine learning as developed by two groups at the Institute of Automation and Remote Control (Aizerman's laboratory including Braverman and Rozonoer and Lerner's laboratory including Vapnik and Chervonenkis). For a brief history of the Institute and research on statistical learning there, including the role of Emmanuel Markovich Braverman see [21], especially Chapter 5. The first development, described in Chapter 4, consisted in adapting predictive distributions to the IID model for the simplest linear case (see [145] and Chapter 4) for more details). The second development (published in [139]), is the combination of conformal predictive distributions with kernel methods that were invented by the members of Aizerman's laboratory, first of all Braverman and Rozonoer [21, p. 48]. This development is outlined in Section 5.6 where kernelized versions of the main algorithms of Chapter 5 are described. The experimental section of this chapter (Section 5.8) demonstrates important advantages of kernelized versions. The computational efficiency of kernelized predictive distributions is studied theoretically in Section 5.6 and predictive efficiency is studied empirically in Section 5.8. As demonstrated in Section 5.8, a universal (Laplacian) kernel works remarkably well.

5.2 The problem

Probabilistic regression requires predicting the label y_{n+1} (we use $\mathbf{y}$ and y_{n+1} interchangeably to describe the label of the $(n+1)$ st observation) based on the training sequence of observations $z_i = (x_i, y_i)$, $i = 1, \ldots, n$ and a new test object $x_{n+1} \in \mathbb{R}^p$. This rather general model specification can cover both regression problems with objects $\mathbf{x}$ containing explanatory variables for labels $\mathbf{y}$ from the set $\mathbf{Y}$, as well as time-series problems with objects $\mathbf{x}$ containing previous history and other time-series based features of the object $\mathbf{z}$. Each observation $z_i = (x_i, y_i)$, $i = 1, \ldots, n+1$, consists of two components, the object x_i from a measurable space $\mathbf{X}$ that we call the *object space* and the label y_i that belongs to a measurable space $\mathbf{Y}$ that we call the *label space*. In the probabilistic regression setting the object space is the real line, $\mathbf{Y} = \mathbb{R}$. In the probabilistic forecasting setting, our prediction is the probability measure on the label space $\mathbf{Y} = \mathbb{R}$, this measure will be represented by its cumulative distribution function (CDF).

The various approaches to the probabilistic prediction problem include that of Bayesian methods (see Section 5.3 below), statistical parametric approach (discussed in Section 5.4), and more recently nonparametric approach based on standard assumptions in machine learning (namely the IID model) that is discussed in this chapter. Using the terminology in [138], in conformal prediction it is convenient to differentiate between the two kinds of assumptions: hard and soft. The hard assumption is the standard assumption of machine learning — namely that the observations are generated independently from the same probability distribution (the IID assumption). The validity of our probabilistic forecasts will depend only on the model with hard assumption. The soft assumption is the assumption that the label y of an object x depends on x in an approximately linear manner — this soft assumption will be used to optimize efficiency.

Similar to standard machine learning notation, we will use a fixed parameter $a > 0$ that determines the degree of regularization applied to the solution, regularization becomes indispensable when kernel methods are used.

5.3 Bayesian solution

Bayesian approach provides a very satisfactory solution to our probabilistic prediction problem. This approach has dominated statistical inference for more than 150 years (covering period of time roughly from 1770 to 1930) — it has started with the works of Thomas Bayes and Pierre-Simon Laplace and was continued by Karl Pearson. Bayesian approach, however, has limitations as it is based on very strong and restrictive assumptions.

We assume linear statistical model in a feature space and that the noise follows Gaussian distribution. Our model is linear in terms of feature mapping of a deterministic sequence of objects $x_1, \dots, x_{n+1}$:

$$y_i = w \cdot F(x_i) + \xi_i, \quad i = 1, \dots, n+1, \tag{5.1}$$

where $F : \mathbf{X} \to H$ is a mapping from the object space to a Hilbert space H, "$\cdot$" is the dot product in Hilbert space H and w is a random vector distributed as $N(0, (\sigma^2/a)I)$ (I being the identity operator on H), and ξ_i are random variables distributed as $N(0, \sigma^2)$ and independent of w and between themselves. We have introduced a at the end of Section 5.2 as the regularization constant, whilst $\sigma > 0$ is another parameter, the standard deviation of the noise variables ξ_i.

By applying expectation operator $\mathbb{E}$ to the model (5.1), it is easy to check that

$$\begin{aligned} \mathbb{E}\, y_i &= 0, & i &= 1, \dots, n, \\ \operatorname{cov}(y_i, y_j) &= \frac{\sigma^2}{a} \mathcal{K}(x_i, x_j) + \sigma^2 1_{\{i=j\}}, & i, j &= 1, \dots, n, \end{aligned} \tag{5.2}$$

where $\mathcal{K}(x, x') := F(x) \cdot F(x')$. By the theorem on normal correlation (see, e.g., [122, Theorem II.13.2]), the Bayesian predictive distribution for y_{n+1} given x_{n+1} and the training sequence is

$$N\left(k'(K + aI)^{-1}Y, \frac{\sigma^2}{a}\kappa + \sigma^2 - \frac{\sigma^2}{a}k'(K + aI)^{-1}k\right), \tag{5.3}$$

where k is the n-vector $k_i := \mathcal{K}(x_i, x_{n+1})$, $i = 1, \dots, n$; K is the kernel matrix for the first n observations (the training observations only); $K_{i,j} := \mathcal{K}(x_i, x_j)$, $i, j = 1, \dots, n$; $I = I_n$ is the $n \times n$ unit matrix; $Y := (y_1, \dots, y_n)'$ is the vector of the n training labels; and

$\kappa := \mathcal{K}(x_{n+1}, x_{n+1})$.

The weakness of the model (5.1) (used, e.g., in [137, Section 10.3]) is that the Gaussian measure $N(0, (\sigma^2/a)I)$ exists only when H is finite-dimensional, but we can circumvent this difficulty by using (5.2) directly as our Bayesian model, for a given symmetric positive semidefinite $\mathcal{K}$. The mapping F in not part of the picture any longer. This is the standard approach in Gaussian process regression in machine learning.

In the Bayesian solution, there is no difference between the hard and soft model; in particular, (5.2) is required for the validity of the predictive distribution (5.3).

5.4 Fiducial predictive distributions

After its sesquicentennial rule, Bayesian statistics was challenged by Fisher and Neyman, who had little sympathy with each other's views apart from their common disdain for Bayesian methods [139]. Fisher's approach was more ambitious, and his goal was to compute a full probability distribution for a future value (test label in our context) or for the value of a parameter. Neyman and his followers were content with computing intervals for future values (prediction intervals) and values of a parameter (confidence intervals).

Fisher and Neyman relaxed the assumptions of Bayesian statistics by allowing uncertainty, in Knight's [61] terminology. In Bayesian statistics overall probability measure is known, i.e., there is a situation of risk without any uncertainty. Fisher and Neyman used the framework of parametric statistics, in this framework the value of the parameter is a number or an element of a Euclidean space (there is no stochastic model for the value of the parameter. In the next section 5.5 we allow for even greater amount of uncertainty: our statistical model will be the nonparametric IID model (as standard in machine learning).

The available properties of validity naturally become weaker as we relax our assumptions. For predicting future values, conformal prediction ensures calibration in probability, in the terminology of [40, Definition 1]. As to Bayesian prediction, it can be shown that it satisfies a stronger conditional version of this property: Bayesian predictive distributions are calibrated in probability conditionally on the training sequence and test object (more generally, on the past).

The property of being calibrated in probability for conformal prediction is, on the other hand, unconditional; or, in other words, it is conditional on the trivial σ-algebra. Fisher's fiducial predictive distributions satisfy an intermediate property of validity: they are calibrated in probability conditionally on what was called the σ-algebra of invariant events in [87], which is greater than the trivial σ-algebra but smaller than the σ-algebra representing the full knowledge of the past.

Fisher did not formalize his fiducial inference (this is often regarded as his "biggest blunder" [35]). By replacing probability distributions by intervals, Neyman's simplification allowed for him to state suitable notions of validity more easily, as a result of which his approach to statistics became mainstream until the Bayesian approach started to reassert itself towards the end of the 20th century [139]. However, there has been a recent revival of interest in fiducial inference: cf. the BFF (Bayesian, frequentist, and fiducial) series of conferences that began in 2014. Fiducial inference is a key topic of the series, both in the form of confidence distributions (the term introduced by David Cox [24] for distributions for parameters) and predictive distributions (which by definition [121, Definition 1] must be calibrated in probability).

Since fiducial inference was developed in the framework of parametric statistics, it has two versions, one targeting computing confidence distributions and the other predictive distributions. Under nonparametric assumptions, such as our IID model, we are not interested in confidence distributions as the set of all probability measures on the observation space $\mathbf{X} \times \mathbb{R}$ is just too big), we therefore concentrate on predictive distributions. The standard notion of validity for predictive distributions, introduced independently by Schweder and Hjort [117, Chapter 12] and Shen, Liu, and Xie [121], is calibration in probability going back to Philip Dawid's work (see, e.g., [27, Section 5.3] and [28]).

5.5 Conformal predictive distributions

Valid predictive distributions under the IID model can be obtained by slightly relaxing the notion of a predictive distribution as given in [121]. We follow [145] and [136] to define predictive distributions (see 4.4 for definition of conformal predictive distribution (CPD) and related definitions).

5.6 Kernel Ridge Regression Prediction Machine

In this section the Kernel Ridge Regression Prediction Machine (KRRPM) is introduced; it will be the conformal transducer determined by a conformity measure of the form 4.6.1, where $\hat{y}_{n+1}$ is computed using kernel ridge regression.

We use three natural versions of the definition, all three versions are based on (4.6) as soft model (with the IID model being the hard model).

Given a training sequence $(z_1, \dots, z_n) \in \mathbf{Z}^n$ and a test object $x_{n+1} \in \mathbf{X}$, the *kernel ridge regression* predicts

$$\hat{y}_{n+1} := k'(K + aI)^{-1}Y$$

for the label y_{n+1} of x_{n+1}. This is just the mean in (5.3), and the variance is ignored. Plugging this definition into 4.6, we obtain the *deleted KRRPM*. Alternatively, we can replace the conformity measure (4.6) by

$$A(z_1, \dots, z_{n+1}) := y_{n+1} - \widehat{y}_{n+1}, \tag{5.4}$$

where

$$\widehat{y}_{n+1} := \bar{k}'(\bar{K} + aI)^{-1}\bar{Y} \tag{5.5}$$

is the prediction for the label y_{n+1} of x_{n+1} computed using $z_1, \dots, z_{n+1}$ as the training sequence. The notation used in (5.5) is:

- $\bar{k}$ is the $(n+1)$-vector $k_i := \mathcal{K}(x_i, x_{n+1})$, $i = 1, \dots, n+1$
- $\bar{K}$ is the kernel matrix for all $n+1$ observations
- $\bar{K}_{i,j} := \mathcal{K}(x_i, x_j)$, $i, j = 1, \dots, n+1$
- $I = I_{n+1}$ is the $(n+1) \times (n+1)$ unit matrix
- $\bar{Y} := (y_1, \dots, y_{n+1})'$ is the vector of all $n+1$ labels

In this context, $\mathcal{K}$ is any given *kernel*, i.e., symmetric positive semidefinite function $\mathcal{K} : \mathbf{X}^2 \to \mathbb{R}$. The corresponding conformal transducer is the *ordinary KRRPM*. The disadvantage of the deleted and ordinary KRRPM is that they are not RPSs (they can fail to produce a function increasing in y in the presence of extremely high-leverage objects).

Let us set

$$\bar{H} := (\bar{K} + aI)^{-1}\bar{K} = \bar{K}(\bar{K} + aI)^{-1}. \tag{5.6}$$

This *hat matrix* "puts hats on the ys": according to (5.5), $\bar{H}\bar{Y}$ is the vector $(\widehat{y}_1, \dots, \widehat{y}_{n+1})'$, where $\widehat{y}_i$, $i = 1, \dots, n+1$, is the prediction for the label y_i of x_i computed using $z_1, \dots, z_{n+1}$ as the training sequence. We will refer to the entries of the matrix $\bar{H}$ as $\bar{h}_{i,j}$ (where i is the row and j is the column of the entry), abbreviating $\bar{h}_{i,i}$ to $\bar{h}_i$. The usual relation between the residuals in (4.6) and (5.4) is

$$y_{n+1} - \hat{y}_{n+1} = \frac{y_{n+1} - \widehat{y}_{n+1}}{1 - \bar{h}_{n+1}}.$$

The diagonal elements $\bar{h}_i$ of the hat matrix are always in the semi-open interval $[0, 1)$ and so the numerator is non-zero). Similar to Chapter 4, we will be using *studentized residuals* $(y_{n+1} - \widehat{y}_{n+1})(1 - \bar{h}_{n+1})^{-1/2}$, which are half-way between the deleted residuals in 4.6.2 and the ordinary residuals in 4.6.1.

The conformal transducer with the corresponding conformity measure is the (studentized) *KRRPM* [139].

$$A(z_1, \dots, z_{n+1}) := \frac{y_{n+1} - \widehat{y}_{n+1}}{\sqrt{1 - \bar{h}_{n+1}}} \tag{5.7}$$

Later in this section we will see that the KRRPM is an RPS. This version of the conformal transducer will be the primary version considered in this chapter, with "studentized" usually omitted.

An explicit form of the KRRPM

According to (4.8), in order to compute the studentized version of predictive distributions produced by the KRRPM, the following equation need to be solved:

$$\alpha_i^y = \alpha_{n+1}^y$$

Algorithm 3 Kernel Ridge Regression Prediction Machine

Require: A training sequence $(x_i, y_i) \in \mathbf{X} \times \mathbb{R}$, $i = 1, \dots, n$.
Require: A test object $x_{n+1} \in \mathbf{X}$.
1: Define the hat matrix $\bar{H}$ by (5.6), $\bar{K}$ being the $(n+1) \times (n+1)$ kernel matrix.
2: **for** $i \in \{1, 2, \dots, n\}$ **do**
3: Define A_i and B_i by (5.8) and (5.9), respectively.
4: Set $C_i := A_i / B_i$.
5: **end for**
6: Sort $C_1, \dots, C_n$ in the increasing order obtaining $C_{(1)} \le \dots \le C_{(n)}$.
7: Return the following predictive distribution for y_{n+1}:

$$Q_n(y, \tau) := \begin{cases} \frac{i+\tau}{n+1} & \text{if } y \in (C_{(i)}, C_{(i+1)}) \text{ for } i \in \{0, 1, \dots, n\} \\ \frac{i'-1+\tau(i''-i'+2)}{n+1} & \text{if } y = C_{(i)} \text{ for } i \in \{1, \dots, n\}. \end{cases} \tag{5.10}$$

(together with the corresponding inequality $\alpha_i^y < \alpha_{n+1}^y$) for $i = 1, \dots, n+1$.

Combining the definition (4.7) of the conformity scores α_i^y with the definition (5.7) of the studentized version of the conformity measure, and the fact that the predictions $\widehat{y}_i$ can be obtained from $\bar{Y}$ by applying the hat matrix $\bar{H}$ (cf. (5.6)), we can rewrite $\alpha_i^y = \alpha_{n+1}^y$ as

$$\frac{y_i - \sum_{j=1}^n \bar{h}_{ij} y_j - \bar{h}_{i,n+1} y}{\sqrt{1 - \bar{h}_i}} = \frac{y - \sum_{j=1}^n \bar{h}_{n+1,j} y_j - \bar{h}_{n+1} y}{\sqrt{1 - \bar{h}_{n+1}}}.$$

This is a linear equation, $A_i = B_i y$, and by solving it we obtain $y = C_i := A_i / B_i$, where

$$A_i := \frac{\sum_{j=1}^n \bar{h}_{n+1,j} y_j}{\sqrt{1 - \bar{h}_{n+1}}} + \frac{y_i - \sum_{j=1}^n \bar{h}_{ij} y_j}{\sqrt{1 - \bar{h}_i}}, \tag{5.8}$$

$$B_i := \sqrt{1 - \bar{h}_{n+1}} + \frac{\bar{h}_{i,n+1}}{\sqrt{1 - \bar{h}_i}}. \tag{5.9}$$

The following Algorithm 3 for computing the conformal predictive distribution (4.3), allows us to compute (4.8) easily.

The notation i' and i'' used in line 7 of the Algorithm 3 is defined as $i' := \min\{j \mid C_{(j)} = C_{(i)}\}$ and $i'' := \max\{j \mid C_{(j)} = C_{(i)}\}$, to ensure that $Q_n(y, 0) = Q_n(y-, 0)$ and $Q_n(y, 1) = Q_n(y+, 1)$ at $y = C_{(i)}$; $C_{(0)}$ and $C_{(n+1)}$ are understood to be $-\infty$ and ∞, respectively.

Algorithm 3 is not computationally efficient for a large test set, since the hat matrix $\bar{H}$ (cf. (5.6)) needs to be computed from scratch for each test object. To obtain a more efficient version, we use a standard formula for inverting partitioned matrices (see, e.g.,

[46, (8)] or [137, (2.44)]) to obtain

$$\begin{aligned}
\bar{H} = (\bar{K} + aI)^{-1}\bar{K} &= \begin{pmatrix} K + aI & k \\ k' & \kappa + a \end{pmatrix}^{-1} \begin{pmatrix} K & k \\ k' & \kappa \end{pmatrix} \\
&= \begin{pmatrix} (K + aI)^{-1} + d(K + aI)^{-1}kk'(K + aI)^{-1} & -d(K + aI)^{-1}k \\ -dk'(K + aI)^{-1} & d \end{pmatrix} \begin{pmatrix} K & k \\ k' & \kappa \end{pmatrix} \\
&= \begin{pmatrix} H + d(K + aI)^{-1}kk'H - d(K + aI)^{-1}kk' \\ -dk'H + dk' \end{pmatrix} && (5.11) \\
&\qquad \begin{pmatrix} (K + aI)^{-1}k + d(K + aI)^{-1}kk'(K + aI)^{-1}k - d\kappa(K + aI)^{-1}k \\ -dk'(K + aI)^{-1}k + d\kappa \end{pmatrix} && (5.12) \\
&= \begin{pmatrix} H + d(K + aI)^{-1}kk'(H - I) & d(I - H)k \\ dk'(I - H) & -dk'(K + aI)^{-1}k + d\kappa \end{pmatrix} && (5.13) \\
&= \begin{pmatrix} H - ad(K + aI)^{-1}kk'(K + aI)^{-1} & ad(K + aI)^{-1}k \\ adk'(K + aI)^{-1} & d\kappa - dk'(K + aI)^{-1}k \end{pmatrix}, && (5.14)
\end{aligned}$$

where

$$d := \frac{1}{\kappa + a - k'(K + aI)^{-1}k} \tag{5.15}$$

(the denominator is positive by the theorem on normal correlation, already used in Section 5.3), the equality in line (5.13) follows from $\bar{H}$ being symmetric (which allows us to ignore the upper right block of the matrix (5.11)–(5.12)), and the equality in line (5.14) follows from

$$I - H = (K + aI)^{-1}(K + aI) - (K + aI)^{-1}K = a(K + aI)^{-1}.$$

We have been using the notation H for the training hat matrix

$$H = (K + aI)^{-1}K = K(K + aI)^{-1}. \tag{5.16}$$

Notice that the constant ad occurring in several places in (5.14) is between 0 and 1:

$$ad = \frac{a}{a + \kappa - k'(K + aI)^{-1}k} \in (0, 1] \tag{5.17}$$

(the fact that $\kappa - k'(K + aI)^{-1}k$ is nonnegative follows from the lower right entry $\bar{h}_{n+1}$ of the hat matrix (5.14) being nonnegative.

The important components in the expressions for A_i and B_i (cf. (5.8) and (5.9)) are, according to (5.14),

$$\begin{aligned} 1 - \bar{h}_{n+1} &= 1 + dk'(K + aI)^{-1}k - d\kappa = 1 + \frac{k'(K + aI)^{-1}k - \kappa}{\kappa + a - k'(K + aI)^{-1}k} \\ &= \frac{a}{\kappa + a - k'(K + aI)^{-1}k} = ad, \end{aligned} \tag{5.18}$$

$$\begin{aligned} 1 - \bar{h}_i &= 1 - h_i + ade_i'(K + aI)^{-1}kk'(K + aI)e_i \\ &= 1 - h_i + ad(e_i'(K + aI)^{-1}k)^2, \end{aligned} \tag{5.19}$$

where $h_i = h_{i,i}$ is the ith diagonal entry of the hat matrix (5.16) for the n training objects and e_i is the ith vector in the standard basis of $\mathbb{R}^n$ (so that the jth component of e_i is $1_{\{i=j\}}$ for $j = 1, \dots, n$). Let $\hat{y}_i := e_i'HY$ be the prediction for y_i computed from the training sequence $z_1, \dots, z_n$ and the test object x_i. Using (5.18) (but not using (5.19) for now), we can transform (5.8) and (5.9) as

$$\begin{aligned} A_i &:= \frac{\sum_{j=1}^n \bar{h}_{n+1,j}y_j}{\sqrt{1 - \bar{h}_{n+1}}} + \frac{y_i - \sum_{j=1}^n \bar{h}_{ij}y_j}{\sqrt{1 - \bar{h}_i}} \\ &= (ad)^{-1/2} \sum_{j=1}^n ady_j k'(K + aI)^{-1}e_j \\ &\quad + \frac{y_i - \sum_{j=1}^n h_{ij}y_j + \sum_{j=1}^n ady_j e_i'(K + aI)^{-1}kk'(K + aI)^{-1}e_j}{\sqrt{1 - \bar{h}_i}} \\ &= (ad)^{1/2}k'(K + aI)^{-1}Y + \frac{y_i - \hat{y}_i + ade_i'(K + aI)^{-1}kk'(K + aI)^{-1}Y}{\sqrt{1 - \bar{h}_i}}, \\ &= \sqrt{ad}\hat{y}_{n+1} + \frac{y_i - \hat{y}_i + ad\hat{y}_{n+1}e_i'(K + aI)^{-1}k}{\sqrt{1 - \bar{h}_i}}, \end{aligned} \tag{5.20}$$

where $\hat{y}_{n+1}$ is the Bayesian prediction for y_{n+1} (cf. the expected value in (5.3)), and

$$B_i := \sqrt{1-\bar{h}_{n+1}} + \frac{\bar{h}_{i,n+1}}{\sqrt{1-\bar{h}_i}} = \sqrt{ad} + \frac{adk'(K+aI)^{-1}e_i}{\sqrt{1-\bar{h}_i}}. \tag{5.21}$$

Therefore, we can implement Algorithm 3 as follows. Preprocessing the training sequence takes time $O(n^3)$ (or faster if using, say, the Coppersmith–Winograd algorithm and its versions; we assume that the kernel $\mathcal{K}$ can be computed in time $O(1)$):

1. The $n \times n$ kernel matrix K can be computed in time $O(n^2)$.
2. The matrix $(K+aI)^{-1}$ can be computed in time $O(n^3)$.
3. The diagonal of the training hat matrix $H := (K+aI)^{-1}K$ can be computed in time $O(n^2)$.
4. All $\hat{y}_i$, $i = 1, \dots, n$, can be computed by $\hat{y} := HY = (K+aI)^{-1}(KY)$ in time $O(n^2)$ (even without knowing H).

Processing each test object x_{n+1} takes time $O(n^2)$:

1. Vector k and number κ (as defined after (5.3)) can be computed in time $O(n)$ and $O(1)$, respectively.
2. Vector $(K+aI)^{-1}k$ can be computed in time $O(n^2)$.
3. Number $k'(K+aI)^{-1}k$ can now be computed in time $O(n)$.
4. Number d defined by (5.15) can be computed in time $O(1)$.
5. For all $i = 1, \dots, n$, compute $1-\bar{h}_i$ as (5.19), in time $O(n)$ overall (given the vector computed in 2).
6. Compute the number $\hat{y}_{n+1} := k'(K+aI)^{-1}Y$ in time $O(n)$ (given the vector computed in 2).
7. Finally, compute A_i and B_i for all $i = 1, \dots, n$ as per (5.20) and (5.21), set $C_i := A_i/B_i$, and output the predictive distribution (5.10). This takes time $O(n)$ except for sorting the C_i, which takes time $O(n \log n)$.

5.7 Limitation of the KRRPM

The KRRPM makes a significant step forward as compared to the LSPM described in Chapter 4: our soft model (5.1) is no longer linear in x_i. In fact, by using a universal kernel (such as Laplacian kernel in Section 5.8) allows the function $x \in \mathbf{X} \mapsto w \cdot F(x)$ to approximate any continuous function arbitrarily well within any compact set in $\mathbf{X}$. However, since we are interested in predictive distributions rather than point predictions, using the soft model (5.1) still results in the KRRPM being restricted. In this section we discuss the nature of the restriction, using the ordinary KRRPM as a technical tool.

The Bayesian predictive distribution (5.3) is Gaussian and (as clear from (5.1) and from the bottom right entry of (5.14) being nonnegative) its variance is at least σ^2. We will see that the situation with the conformal distribution is not as bad, despite the remaining restriction. To understand the nature of the restriction it will be convenient to ignore the denominator in (5.7), i.e., to consider the ordinary KRRPM; the difference between the (studentized) KRRPM and ordinary KRRPM will be small in the absence of high-leverage objects (an example will be given in the next section). For the ordinary KRRPM we have, in place of (5.8) and (5.9),

$$A_i := \sum_{j=1}^{n} \bar{h}_{n+1,j} y_j + y_i - \sum_{j=1}^{n} \bar{h}_{i,j} y_j,$$

$$B_i := 1 - \bar{h}_{n+1} + \bar{h}_{i,n+1}.$$

Therefore, (5.20) and (5.21) become

$$A_i = ad\hat{y}_{n+1} + y_i - \hat{y}_i + ad\hat{y}_{n+1} e_i'(K + aI)^{-1} k$$

and

$$B_i = ad + ade_i'(K + aI)^{-1} k,$$

respectively. For $C_i := A_i/B_i$ we now obtain

$$C_i = \hat{y}_{n+1} + \frac{y_i - \hat{y}_i}{ad + ade_i'(K + aI)^{-1}k} = \hat{y}_{n+1} + \frac{\sigma^2_{\text{Bayes}}/\sigma^2}{1 + e_i'(K + aI)^{-1}k}(y_i - \hat{y}_i), \quad (5.22)$$

where $\hat{y}_{n+1}$ is, as before, the Bayesian prediction for y_{n+1}, and σ^2_{Bayes} is the variance of the Bayesian predictive distribution (5.3) (cf. (5.17)).

The second addend $e_i'(K + aI)^{-1}k$ in the denominator of (5.22) is the prediction for the label of the test object x_{n+1} in the situation where all training labels are 0 apart from the ith, which is 1. For a long training sequence we can expect it to be close to 0 (unless x_i or x_{n+1} are highly influential); therefore, we can expect the shape of the predictive distribution output by the ordinary KRRPM to be similar to the shape of the empirical distribution function of the residuals $y_i - \hat{y}_i$. In particular, this shape does not depend (or depends weakly) on the test object x_{n+1}. This lack of sensitivity of the predictive distribution on the test object prevents the conformal predictive distributions output by the KRRPM from being universally consistent in the sense of [136]. The predictive distribution is not necessarily Gaussian (as in (5.3)), and the distribution is fitted to all training residuals (and not just the residuals for objects similar to the test object). One possible way to get universally consistent conformal predictive distributions would be to replace the right-hand side of (4.10) by $\hat{F}_{n+1}(y_{n+1})$, where $\hat{F}_{n+1}$ is the Bayesian predictive distribution for y_{n+1} computed from x_{n+1} and $z_1, \dots, z_{n+1}$ as training sequence for a sufficiently flexible Bayesian model (in any case, more flexible than our homoscedastic model (5.1)). This idea was referred to as de-Bayesing in [137, Section 4.2] and frequentizing in [149, Section 3]. However, modelling input-dependent (heteroscedastic) noise efficiently is a well-known difficult problem in Bayesian regression, including Gaussian process regression (see, e.g., [42, 74, 123]).

5.8 Experimental results

In the first part of this section we illustrate the main advantage of the KRRPM over the LSPM introduced in Chapter 4, namely its flexibility — for a suitable kernel, it gets the location of the predictive distribution right. The second part of this section illustrates the limitation of the KRRPM discussed in the previous section: while the KRRPM adapts to the shape of the distribution of labels, the adaptation is not conditional on the test object. Both points will be demonstrated using artificial data sets.

In the experiment we generate a training sequence of length 1000 from the model

$$y_i = w_1 \cos x_{i,1} + w_2 \cos x_{i,2} + w_3 \sin x_{i,1} + w_4 \sin x_{i,2} + \xi_i, \tag{5.23}$$

where $(w_1, w_2, w_3, w_4) \sim N(0, I_4)$ (I_4 being the unit 4×4 matrix), $(x_{i,1}, x_{i,2}) \sim U[-1,1]^2$ ($U[-1,1]$ being the uniform probability distribution on $[-1,1]$), and $\xi_i \sim N(0,1)$, all independent. This corresponds to the Bayesian ridge regression model with $a = \sigma = 1$. The true kernel is

$$\begin{aligned}
&\mathcal{K}((x_1, x_2), (x_1', x_2')) \\
&= (\cos x_1, \cos x_2, \sin x_1, \sin x_2) \cdot (\cos x_1', \cos x_2', \sin x_1', \sin x_2') \\
&= \cos(x_1 - x_1') + \cos(x_2 - x_2'). \qquad (5.24)
\end{aligned}$$

By definition (see, e.g., [125]) a kernel is *universal* if any continuous function can be uniformly approximated (over each compact set) by functions in the corresponding reproducing kernel Hilbert space. An example of a universal kernel is the *Laplacian kernel*

$$\mathcal{K}(x, x') := \exp\left(-\left\|x - x'\right\|\right).$$

Laplacian kernels were introduced and studied in [128]; the corresponding reproducing kernel Hilbert space has the Sobolev norm

$$\|u\|^2 = 2\int_{-\infty}^{\infty} u(t)^2 \mathrm{d}t + 2\int_{-\infty}^{\infty} u'(t)^2 \mathrm{d}t$$

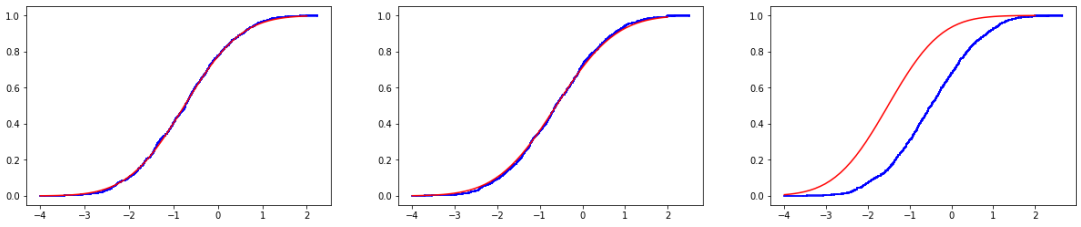

FIGURE 5.1: The predictive distribution for the label of the test object $(1, 1)$ based on a training sequence of length 1000 (all generated from the model (5.23)). The red line in each panel is the Bayesian predictive distribution based on the true kernel (5.24), and the blue line is the conformal predictive distribution based on: the true kernel (5.24) in the left-most panel; the Laplacian kernel in the middle panel; the linear kernel in the right-most panel.

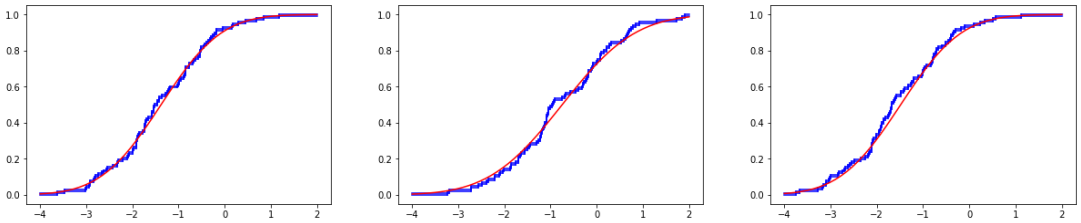

FIGURE 5.2: The analogue of Figure 5.1 for a training sequence of length 100.

(see [128, Corollary 1]). This expression shows that Laplacian kernels are indeed universal. On the other hand, the *linear kernel* $K(x, x') := x \cdot x'$ is far from being universal; remember that the LSPM [145] corresponds to this kernel and $a = 0$.

Figure 5.1 illustrates that for this data set, universal kernels lead to better results. The parameter a in Figure 5.1 is the true one, $a = 1$. In the case of the Bayesian predictive distribution, the parameter $\sigma = 1$ is also the true one — unlike for Bayesian predictive distributions predictive distributions do not require σ. The right-most panel shows that, the conformal predictive distribution based on the linear kernel can get the predictive distribution wrong. The other two panels show that the true kernel and, more importantly, the Laplacian kernel (chosen independently of the model (5.23)) are much more accurate. Figure 5.1 shows predictive distributions for a specific test object, $(1, 1)$, but this behaviour is typical. The effect of using a universal kernel becomes much less pronounced (or even disappears completely) for smaller lengths of the training sequence: see Figure 5.2 using 100 training observations (whereas Figure 5.1 uses 1000).

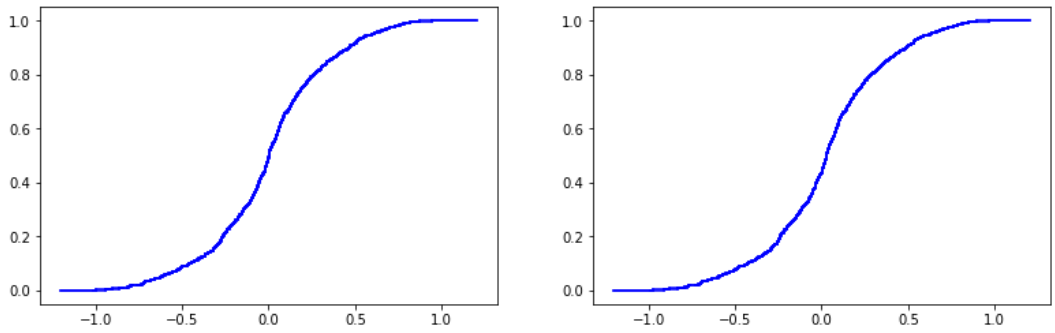

FIGURE 5.3: Left panel: predictions of the KRRPM for a training sequence of length 1000 and $x_{1001} = 0$. Right panel: predictions for $x_{1001} = 1$. The data are described in the text.

We now illustrate the limitation of the KRRPM that we discussed in the previous section. An artificial data set is generated as follows: $x_i \in [0, 1]$, $i = 1, \dots, n$, are chosen independently from the uniform distribution U on $[0, 1]$, and $y_i \in [-x_i, x_i]$ are then chosen independently, again from the uniform distributions $U[-x_i, x_i]$ on their intervals. Figure 5.3 shows the prediction for $x_{n+1} = 0$ on the left and for $x_{n+1} = 1$ on the right for $n = 1000$; there is no discernible difference between the studentized and ordinary versions of the KRRPM. The difference between the predictions for $x_{n+1} = 0$ and $x_{n+1} = 1$ is slight, whereas ideally we would like the former prediction to be concentrated at 0 whereas the latter should be close to the uniform distribution on $[-1, 1]$ [139].

Fine details can be seen in Figure 5.4, which is analogous to Figure 5.3 but uses a training sequence of length $n = 10$. It shows the plots of the functions $Q_n(y, 0)$ and $Q_n(y, 1)$ of y, in the notation of (4.3). These functions carry all information about $Q_n(y, \tau)$ as function of y and τ since $Q_n(y, \tau)$ can be computed as the convex mixture $(1-\tau)Q_n(y, 0)+\tau Q_n(y, 1)$ of $Q_n(y, 0)$ and $Q_n(y, 1)$ [139].

In our future experiments we can use three kinds of kernels:

- the *polynomial kernel*

$$K(x, x') := (1 + x \cdot x')^k$$

  parameterized by $k = 1, 2, \dots$ and sometimes abbreviated to "poly k";

- the *Gaussian kernel*

$$K(x, x') := \exp\left(-c \left\|x - x'\right\|^2\right)$$

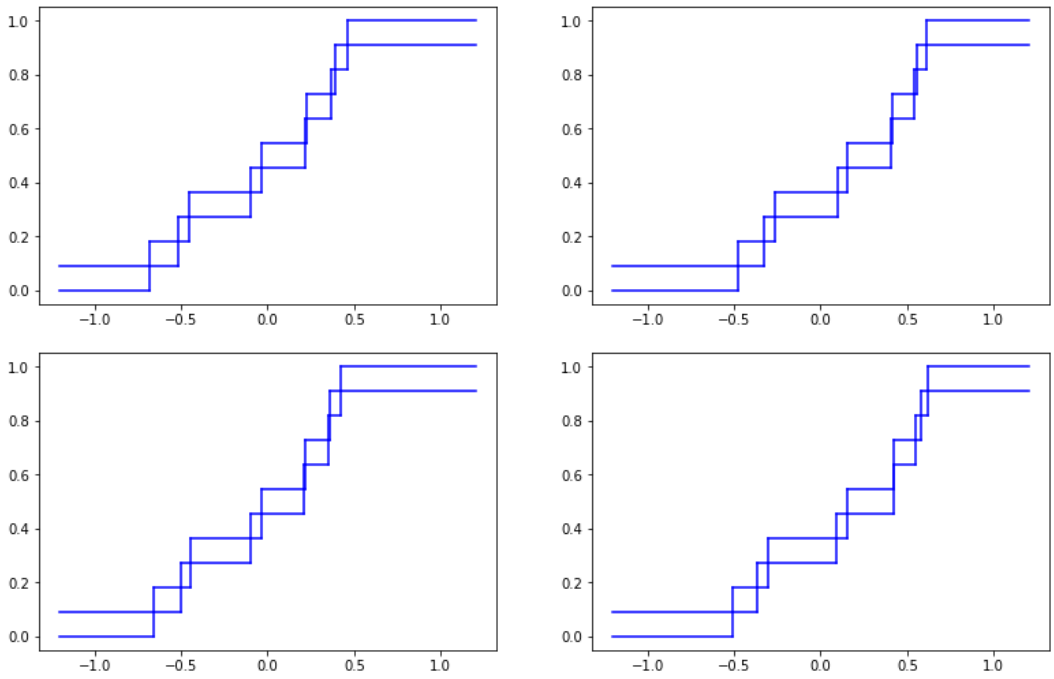

FIGURE 5.4: Upper left panel: predictions of the (studentized) KRRPM for a training sequence of length 10 and $x_{11} = 0$. Upper right panel: analogous predictions for $x_{11} = 1$. Lower left panel: predictions of the ordinary KRRPM for a training sequence of length 10 and $x_{11} = 0$. Lower right panel: analogous predictions for $x_{11} = 1$.

parameterized by $c > 0$ and sometimes abbreviated to "Gauss c";

- the *Laplacian kernel*

$$K(x, x') := \exp\left(-c \left\|x - x'\right\|\right)$$

parameterized by $c > 0$ and sometimes abbreviated to "Laplace c".

According to [110] (see also [47, Example 2.22]), polynomial kernels are not universal and the corresponding reproducing kernel Hilbert space consists of degree k polynomials. On the other hand, according to [126], Gaussian kernels are universal (and [126] gives an explicit description of the corresponding reproducing kernel Hilbert spaces). Laplacian kernels are also universal, as mentioned earlier.

Further experiments with artificial data

We follow the experiments in Section 10.3 of [137] but in the kernelized setting. In this chapter we use pictures of two kinds. First, those for the correct kernel (such as (5.24)):

1. The validity pictures like those in Section 10.3 of [137], Figures 10.1–10.4. The calibration curve would be a plot of the empirical frequency of $Q_n \le \epsilon$ vs ϵ. The efficiency would be measured by one number (no need to show it on the plot): the cumulative loss function (see below) over the test set.

2. Efficiency figures similar to those in the next subsection. Now we have the shaded areas for the CPDs and the lines for Oracles II and III.

Those for the standard kernels (different from the true one (5.24)), such as polynomial, Gaussian, and Laplacian: the only kind of pictures and figures would be those described in item 1.

Experiments with a benchmark data set

In our experiments we can use continuous ranked probability score, which is the most standard proper loss function for regression [48, 41]. To make it applicable to our conformal predictive distributions, we modify them slightly by replacing (5.10) by

$$Q_n(y,\tau) := \begin{cases} \frac{i}{n} & \text{if } y \in (C_{(i)}, C_{(i+1)}) \text{ for } i \in \{0,1,\dots,n\} \\ \frac{i}{n} & \text{if } y = C_{(i)} \text{ for } i \in \{1,\dots,n\}. \end{cases}$$

Since Q is now a bona fide distribution function, we can use the usual expression

$$\mathrm{CRPS}(Q,y) := \int_{-\infty}^{y} Q(u,0)^2 \mathrm{d}u + \int_{y}^{\infty} (1-Q(u,1))^2 \mathrm{d}u$$

(as in [41, (20)]). This loss function can be used for comparing conformal predictive distributions among themselves and with other predictive distributions.

First we use the standard Boston Housing data set available from the UCI repository [33]. It consists of 506 observations each with 14 attributes. We split it into a training set of size 406 and test set of size 100 (using the original order of the observations). All attributes were standardized making the mean of each attribute zero and its standard deviation 1 (based only on the training set).

Kernel	$a = 0.001$	$a = 0.01$	$a = 0.1$	$a = 1$	$a = 10$
poly 1	3.45	3.45	3.45	3.44	3.42
poly 2	2.71	2.63	2.58	2.56	2.70
poly 3	59.90	20.40	8.62	4.43	6.86
Gauss 0.01	2.30	2.29	2.38	3.23	5.04
Gauss 0.05	2.42	2.24	2.40	2.99	5.33
Gauss 0.1	2.74	2.41	2.60	3.50	6.29

TABLE 5.1: The cumulative losses for polynomial kernels and Gaussian kernels

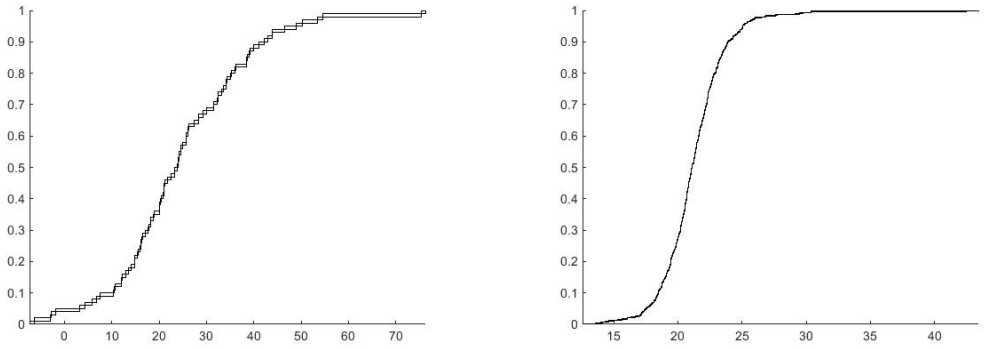

FIGURE 5.5: The predictive distribution for the label of the 100th observation (based on the previous 99 observations) and for the label of the 500th observation (based on the previous 499 observations) for the Boston Housing data set

Our results are given in Table 5.1. For polynomial kernels, the best result is for degree 2 and $a = 1$: the cumulative loss is 2.56. The best RBF result is slightly better: its parameter $c = 0.05$ and $a = 0.01$, the loss is 2.24, but this was more sensitive to parameter tuning.

We do not give validity results like those in [145] (they are useless since validity is guaranteed by our theoretical results). Two sample predictive distribution functions are shown in Figure 5.5. The kernel is Gaussian, and the parameters are $a = 0.01$ and $c = 0.05$ (giving the best values in Table 5.1).

5.9 Conclusion

The main contribution of this chapter is extension the study of probabilistic regression by combining conformal predictive distributions with kernel methods (published paper version [139]). Experimental study of the predictive efficiency demonstrates important

advantages of the kernelized versions of CPDs and shows that universal (Laplacian) kernel performs remarkably well in terms of outputting accurate probabilistic predictions for the test objects.

Chapter 6

Computationally efficient probabilistic regression

Conformal predictive distributions (CPD) output probability distributions of the label value in machine learning regression problems. This chapter extends the study of probabilistic regression to address the relative computational inefficiency of classical conformal predictors. Two novel computationally efficient conformal predictive systems are introduced — split conformal predictive systems (SPC) and cross-conformal predictive system (CCPS). Split conformal predictive systems provide guaranteed validity, whilst the main advantage of cross-conformal predictive systems is greater predictive efficiency. For cross-conformal predictive systems validity only holds empirically and in the absence of excessive randomization. The main aim of this chapter is to define and study computationally efficient versions of CPS without any restrictions related to the underlying algorithm.

6.1 Introduction

In this chapter, we continue the study of probabilistic regression problem introduced in Chapter 4 and continued in Chapter 5 with the study of kernel probabilistic regression. Chapter 4 described extension of parametric statistical "prediction confidence distributions" to machine learning regression tasks using only limited assumptions customary of machine learning, namely that the observations are generated using the IID model. Unlike the parametric approach of statistical predictive distributions, conformal predictive

distributions do not require specification of the data model and thus can be applied to a wide range of prediction and decision-making problems.

The disadvantage of CPS is that for many underlying algorithms CPS are computationally inefficient as they require re-training the underlying machine learning or statistical algorithm for each test object and each potential label for this object. In practice this can be done efficiently only for a narrow class of underlying algorithms, including for Least Squares and Kernel Ridge Regression as described in Chapter 4 and Chapter 4 accordingly.

A very recent development in Venn prediction has been the introduction of split Venn-Abers predictive systems in [101]. Venn-Abers predictive systems are another way to produce predictive distributions. In this chapter we explore several versions of Venn-Abers predictive systems and compare them with conformal predictive systems.

We follow the definition of randomized predictive systems (RPS) introduced in Chapter 4 and in section 6.2 define a special case of RPS — split conformal predictive systems (SCPS). Split conformal predictive systems (SCPS) are computationally efficient, but may suffer loss in predictive efficiency when compared with CPS, as CPS uses data more efficiently. A very important advantage of SCPS is their validity (similar to CPS) — we demonstrate validity of SCPS in Section 6.2.

Section 6.3 introduces cross-conformal predictive systems (CCPS), whilst CCPS use data more efficiently, they can lose their validity in principle as they are formally are no longer RPS — in practice CCPS usually satisfy the requirement of validity as demonstrated in Section 6.5.

Section 6.5 compares the predictive efficiency of SCPS and CCPS and explores their empirical validity. The predictive efficiency of predictive distributions is measured using a loss function called continuous ranked probability score (CRPS, see 1.2.5).

6.2 Split conformal predictive systems

In this section we will modify the definition 4.4.6 of conformal predictive systems introduced in Chapter 4 along the lines of [6, Section 2.3] by removing an unnecessary assumption in [137, Section 4.1]. The definitions below follow [139].

Definition 6.2.1 (Split Conformity Measure)**.** A split conformity measure is a family of measurable functions $A_m : \mathbf{Z}^{m+1} \to \mathbb{R} \cup \{-\infty, \infty\}$, $m = 1, 2, \ldots$. The intention is that $A_m(z_1, \ldots, z_{m+1})$ measures how large the label y_{m+1} in z_{m+1} is, as compared with the labels in $z_1, \ldots, z_m$.

Consider the training sequence $z_1, \ldots, z_n$ that is split into two parts: the proper training sequence $z_1, \ldots, z_m$ and the calibration sequence $z_{m+1}, \ldots, z_n$. Based on the object x from the test set, the task of probabilistic regression is to predict probability distribution for y (similar notation use in Chapter 4 we will use y and y_{n+1} interchangeably to describe the label of the test object that requires prediction.

Definition 6.2.2 (Split Conformal Transducer)**.** The output of the split conformal transducer determined by the split conformity measure A is defined as

$$Q(z_1, \ldots, z_n, (x, y), \tau) := \frac{1}{n-m+1} \left|\{i = m+1, \ldots, n \mid \alpha_i < \alpha^y\}\right| + \frac{\tau}{n-m+1} \left|\{i = m+1, \ldots, n \mid \alpha_i = \alpha^y\}\right| + \frac{\tau}{n-m+1}, \quad (6.1)$$

Definition 6.2.3 (Conformity Scores)**.** The conformity scores α_i, $i = m+1, \ldots, n$, and α^y, $y \in \mathbb{R}$, are defined by

$$\alpha_i := A(z_1, \ldots, z_m, (x_i, y_i)), \qquad i = m+1, \ldots, n,$$
$$\alpha^y := A(z_1, \ldots, z_m, (x, y)).$$

Definition 6.2.4 (Split Conformal Transducer)**.** A function is a split conformal transducer if it is the split conformal transducer determined by some split conformity measure.

Definition 6.2.5 (Split Conformal Predictive System)**.** A split conformal predictive system (SCPS) is a function which is both a split conformal transducer and a randomized predictive system.

The standard property of validity for split conformal transducers (that is satisfied automatically) is that the values $Q(z_1, \ldots, z_n, z, \tau)$ have uniform distribution on $[0, 1]$ if $z_1, \ldots, z_n, z$ are IID and τ is generated independently of $z_1, \ldots, z_n, z$ from the uniform probability distribution U on $[0, 1]$ (see, e.g., [137, Proposition 4.1]).

It is much easier to get an RPS using split conformal transducers than using conformal transducers.

Definition 6.2.6 (Isotonic Split Conformity Measure)**.** A split conformity measure A is isotonic if, for all m, $z_1, \dots, z_m$, and x, $A(z_1, \dots, z_m, (x, y))$ is isotonic in y, i.e.,

$$y \le y' \Longrightarrow A(z_1, \dots, z_m, (x, y)) \le A(z_1, \dots, z_m, (x, y')) \tag{6.2}$$

Definition 6.2.7 (Balanced Split Conformity Measure)**.** An isotonic split conformity measure A is *balanced* if, for any m and $z_1, \dots, z_m$, the set

$$\operatorname{conv} A(z_1, \dots, z_m, (x, \mathbb{R})) := \operatorname{conv} \{A(z_1, \dots, z_m, (x, y)) \mid y \in \mathbb{R}\} \tag{6.3}$$

does not depend on x, where conv stands for the convex closure in $\mathbb{R}$.

The set (6.3) then coincides with $\operatorname{conv} A(z_1, \dots, z_m, \mathbf{Z})$ and has one of four forms: (a, b), $[a, b)$, $(a, b]$, or $[a, b]$, where $a < b$ are elements of the extended real line $\mathbb{R} \cup \{-\infty, \infty\}$; we will be mainly interested in the case $\operatorname{conv} A(z_1, \dots, z_m, \mathbf{Z}) = (-\infty, \infty)$.

Proposition 6. *The split conformal transducer* (6.1) *based on a balanced isotonic split conformity measure is an RPS.*

Proof. Since property R2 in the definition of RPS 4.4.1 is automatic, we only need to check R1. It is clear that (6.1) is increasing in τ (and linear).

To show that it is increasing in y, split, in the context of (6.1), all $i \in \{m+1, \dots, n\}$ into three groups: the i in group 1 satisfy $\alpha_i < \alpha^y$, the i in group 2 satisfy $\alpha_i = \alpha^y$, and the i in group 3 satisfy $\alpha_i > \alpha^y$. Then (6.1) is the total weight of all i where the weights are 1, $\tau \in [0, 1]$, and 0 for i in groups 1, 2, and 3, respectively. As y increases, α^y increases as well, and therefore, each i can only move to a lower-numbered group thus increasing (6.1).

Out of the remaining two conditions, let us check, e.g., (4.2). It suffices to notice that, since A is balanced, we have $\alpha^y \ge \max_{i \in \{m+1, \dots, n\}} \alpha_i$ from some y on, for any $z_1, \dots, z_n$ and x. □

The next proposition shows that a split conformity measure being isotonic and balanced is not only a sufficient but also a necessary condition for the corresponding split conformal transducer to be an RPS.

Proposition 7. *If the split conformal transducer based on a split conformity measure A is an RPS, A is isotonic and balanced.*

Proof. Suppose A is not isotonic. Fix m, $z_1, \dots, z_m$, x, y, and y' such that $y < y'$ but the consequent of (6.2) is violated. Then the predictive distribution $Q(z_1, \dots, z_m, (x, y), (x, \cdot), 1)$, corresponding to the training sequence proper $z_1, \dots, z_m$, calibration sequence (x, y), test object x, and $\tau = 1$, will not be increasing: its value at y (which is 1) will be greater than its value at y' (which is 0.5).

Now suppose A is not balanced. Fix m, $z_1, \dots, z_m$, and $x, x' \in \mathbf{X}$ such that

$$\operatorname{conv} A(z_1, \dots, z_m, (x, \mathbb{R})) \neq \operatorname{conv} A(z_1, \dots, z_m, (x', \mathbb{R}))$$

(cf. (6.3)). Suppose, for concreteness, that there is $y \in \mathbb{R}$ such that

$$\operatorname{conv} A(z_1, \dots, z_m, (x, \mathbb{R})) \ni y < \operatorname{conv} A(z_1, \dots, z_m, (x', \mathbb{R})),$$

where $y < S$ means $\forall s \in S : y < s$ when $S \subseteq \mathbb{R}$. (The other three possible cases can be analyzed in the same way.) Let the training sequence proper be $z_1, \dots, z_m$, the calibration sequence be (x, y), the test object be x', and the random number be $\tau = 0$. Then we will have

$$\lim_{y' \to -\infty} Q(z_1, \dots, z_m, (x, y), (x', y'), 0) > 0,$$

which contradicts R1 (cf. (4.2)). □

Let us say that a split conformity measure A is *strictly isotonic* if (6.2) holds with both "$\le$" replaced by "$<$". A possible implementation of the SCPS based on a balanced strictly

Algorithm 4 Split Conformal Predictive System

Require: A training sequence $(x_i, y_i) \in \mathbf{Z}$, $i = 1, \dots, n$.
Require: A test object $x \in \mathbf{X}$.
  for $i \in \{1, \dots, n-m\}$ **do**
    Define C_i by the condition $A(z_1, \dots, z_m, z_{m+i}) = A(z_1, \dots, z_m, (x, C_i))$.
  end for
  Sort $C_1, \dots, C_{n-m}$ in the increasing order obtaining $C_{(1)} \le \dots \le C_{(n-m)}$.
  Set $C_{(0)} := -\infty$ and $C_{(n-m+1)} := \infty$.
  Return the predictive distribution (6.4) for the label y of x.

isotonic split conformity measure is shown as Algorithm 4, where the predictive distribution is defined by

$$Q(z_1, \dots, z_n, (x, y), \tau) := \begin{cases} \frac{i+\tau}{n-m+1} & \text{if } y \in (C_{(i)}, C_{(i+1)}) \text{ for } i \in \{0, 1, \dots, n-m\} \\ \frac{i'-1+(i''-i'+2)\tau}{n-m+1} & \text{if } y = C_{(i)} \text{ for } i \in \{1, \dots, n-m\}, \end{cases} \tag{6.4}$$

where $i' := \min\{j \mid C_{(j)} = C_{(i)}\}$ and $i'' := \max\{j \mid C_{(j)} = C_{(i)}\}$. To use the terminology of Chapter 4, the thickness of this predictive distribution is $\frac{1}{n-m+1}$ with the exception size at most $n - m$.

Computational complexity of the Algorithm 4 depends on how difficult it is to solve the equation defining C_i. For a standard choice of split conformity measure:

$$A(z_1, \dots, z_m, (x, y)) := \frac{y - \hat{y}}{\hat{\sigma}}, \tag{6.5}$$

with $\hat{y}$ being prediction for the label y that is computed based on the training sequence $z_1, \dots, z_m$ and test object x,and $\hat{\sigma}$ is an estimate of the quality of $\hat{y}$ computed from the same data. In this case the equation defining C_i

$$A(z_1, \dots, z_m, z_{m+i}) = A(z_1, \dots, z_m, (x, C_i)) \tag{6.6}$$

becomes

$$\frac{y_{m+i} - \hat{y}_{m+i}}{\hat{\sigma}_{m+i}} = \frac{C_i - \hat{y}}{\hat{\sigma}},$$

where $\hat{y}_{m+i}$ (resp. $\hat{y}$) is the prediction for y_{m+i} (resp. y) computed from x_{m+i} (resp. x) as test object and $z_1, \dots, z_m$ as training sequence, and $\hat{\sigma}_{m+i}$ (resp. $\hat{\sigma}$) is the estimate of the quality of $\hat{y}_{m+i}$ (resp. $\hat{y}$) computed from the same data. The last equation can be rewritten to define expression for C_i as:

$$C_i := \hat{y} + \frac{\hat{\sigma}}{\hat{\sigma}_{m+i}} (y_{m+i} - \hat{y}_{m+i}) .$$

For more complicated split conformity measures A, it might be more efficient to use the expression (6.1) directly for a grid of values of y.

6.3 Cross-conformal predictive distributions

Remember that a *multiset* (or bag) is different from a set in that it can contain several copies of the same element. A split conformity measure A is a *cross-conformity measure* if $A(z_1, \dots, z_m, z)$ does not depend on the order of its first m arguments; in other words, if $A(z_1, \dots, z_m, z)$ only depends on the multiset $\wr z_1, \dots, z_m \int$ and z (where $\wr \cdots \int$ is used as the analogue of $\{\cdots\}$ for multisets).

For a balanced isotonic cross-conformity measure A, the corresponding *cross-conformal predictive system* (CCPS) is defined as follows.

Definition 6.3.1 (Cross-Conformal Predictive System (CCPS))**.**

$$p^y = Q(z_1, \dots, z_n, (x, y), \tau) := \frac{1}{n+1} \sum_{k=1}^{K} \left| \left\{ i \in S_k \mid \alpha_{i,k} < \alpha_k^y \right\} \right| + \frac{\tau}{n+1} \sum_{k=1}^{K} \left| \left\{ i \in S_k \mid \alpha_{i,k} = \alpha_k^y \right\} \right| + \frac{\tau}{n+1}. \quad (6.7)$$

Where $(S_1, \dots, S_K)$ is a partition of the index set $\{1, \dots, n\}$, and z_{S_k} consists of all z_i, $i \in S_k$ for random split of the training sequence $z_1, \dots, z_n$ into K non-empty multisets (*folds*) z_{S_k}, $k = 1, \dots, K$, of equal (or as equal as possible) sizes (where $K \in \{2, 3, \dots\}$ is a parameter of the algorithm).

Algorithm 5 Cross-Conformal Predictive System

Require: A training sequence $(x_i, y_i) \in \mathbf{Z}$, $i = 1, \dots, n$.
Require: A test object $x \in \mathbf{X}$.
Split $z_1, \dots, z_n$ into K folds z_{S_k} as described in text.
Set $C := \emptyset$, where C is a multiset.
for $k \in \{1, \dots, K\}$ **do**
  for $i \in S_k$ **do**
    Define $C_{i,k}$ by the condition $A(z_{S_{-k}}, z_i) = A(z_{S_{-k}}, (x, C_{i,k}))$.
    Put $C_{i,k}$ in C.
  end for
end for
Sort C in the increasing order obtaining $C_{(1)} \le \dots \le C_{(n)}$.
Set $C_{(0)} := -\infty$ and $C_{(n+1)} := \infty$.
Return the predictive distribution (6.8) for the label y of x.

For each $k \in \{1, \dots, K\}$ and each potential label $y \in \mathbb{R}$ of the test object x, find the conformity scores of the observations in z_{S_k} and of (x, y) by

$$\alpha_{i,k} := A(z_{S_{-k}}, z_i), \quad i \in S_k, \qquad \alpha_k^y := A(z_{S_{-k}}, (x, y)),$$

where $S_{-k} := \cup_{j \neq k} S_j = \{1, \dots, n\} \setminus S_k$. The corresponding p-values and CCPS are defined by equation 6.7.

The intuition behind (6.7) is that it becomes an SCPS when the training multisets $z_{S_{-k}}$ are replaced by a single hold-out training sequence (one disjoint from and independent of $z_1, \dots, z_n$).

Algorithm 5 describes an implementation of the CCPS based on a balanced strictly isotonic cross-conformity measure, where the predictive distribution is now defined by

$$Q(z_1, \dots, z_n, (x, y), \tau) := \begin{cases} \frac{i+\tau}{n+1} & \text{if } y \in (C_{(i)}, C_{(i+1)}) \text{ for } i \in \{0, 1, \dots, n\} \\ \frac{i'-1+(i''-i'+2)\tau}{n+1} & \text{if } y = C_{(i)} \text{ for } i \in \{1, \dots, n\}, \end{cases} \tag{6.8}$$

where, as before, $i' := \min\{j \mid C_{(j)} = C_{(i)}\}$ and $i'' := \max\{j \mid C_{(j)} = C_{(i)}\}$; the only difference from (6.4) is that we use n in place of $n - m$ (now all training observations are used for calibration). The thickness of this predictive distribution is $\frac{1}{n+1}$ with the

exception size at most n. The size of the multiset C in Algorithm 5 grows from 0 to n as the algorithm runs. As in the case of SCPS, it might be easier to use (6.7) directly if the equations defining $C_{i,k}$ are difficult to solve. (Alternatively, one could use (6.10) below instead of (6.7).)

Definition 6.3.2 (Fold p-value)**.** Define a *fold p-value* as separate p-value for each fold

$$p_k^y := \frac{1}{|S_k|+1}\left|\left\{i \in S_k \mid \alpha_{i,k} < \alpha_k^y\right\}\right| + \frac{\tau}{|S_k|+1}\left|\left\{i \in S_k \mid \alpha_{i,k} = \alpha_k^y\right\}\right| + \frac{\tau}{|S_k|+1} \tag{6.9}$$

for each fold (cf. (6.1)); let us check that p^y is close to being an average of p_k^y. Comparing (6.7) and (6.9), we can see that

$$(n+1)p^y - \tau = \sum_{k=1}^{K} (|S_k|+1)\, p_k^y - K\tau,$$

which implies

$$p^y = \sum_{k=1}^{K} \frac{|S_k|+1}{n+1} p_k^y - \frac{K-1}{n+1}\tau. \tag{6.10}$$

The sum $\sum_{k=1}^{K} \ldots$ is not quite a weighted average of p_k^y since the sum of the weights is slightly above 1 ("slightly" assumes $K \ll n$), but this is partially compensated by the subtrahend in (6.10); overall, the right-hand side of (6.10) is a weighted average of p_k^y and τ, with the weight in front of τ being negative.

According to the intuition behind cross-conformal predictive distributions described earlier, we will get perfect validity for CCPS if we replace the K training multisets (the complements to the K folds) by one hold-out training sequence. But whereas SCPS are provably valid, in the sense of being RPS, real CCPS are not RPS: see the example in [135, Appendix A]. This has been demonstrated in experimental studies, e.g., Linusson *et al.* [81] has shown the danger of randomized and extremely unstable underlying algorithms. (Perhaps such unstable algorithms might be stabilized, to some degree, by using the same seed of the random numbers generator for each fold, or by averaging conformity scores over several seeds, or both.)

A useful intuition in [81] is that "the random fold p-value and then essentially averaged by cross-conformal predictors are to some degree independent. The distribution of

cross-conformal p-values is intermediate between the uniform and the Bates distributions, cross-conformal p-values are therefore conservative (for small significance levels) when not exact." According to the result in [147] (see, e.g., Table 1 for $r := 1$), one can obtain provably valid (but perhaps conservative) p-values, when p-values output by a cross-conformal transducer are multipled by 2. Linusson *et al.* [81] observed empirically that "for randomized and unstable underlying algorithms even unadjusted p-values output by a cross-conformal transducer are valid but perhaps overly conservative for interesting (not exceeding 0.5) significance levels".

A more general procedure than the cross-conformal predictor was proposed in [15] under the name of "aggregated conformal predictor". Similar methods might be applicable for producing conformal predictive distributions.

6.4 Continuous ranked probability score

The continuous ranked probability score was previously defined in Chapter 1 (see 1.2.5) as

$$\mathrm{CRPS}(F, y_i) := \int_{-\infty}^{\infty} \left(F(y) - \mathbf{1}_{\{y \geq y_i\}}\right)^2 \mathrm{d}y, \tag{6.11}$$

CRPS attains the lowest possible value of 0 when distribution function F is concentrated at y_i, and in all other cases $\mathrm{CRPS}(F, y_i)$ will be positive. (See, e.g., [40] for further details and references.)

Equation (6.11) is not directly applicable to split and cross-conformal predictive distributions, however in practice the fuzziness can usually be ignored, even for relatively small datasets: see, e.g., Figure 6.1 and [136]. However, conceptually we do need to change the definitions of split and cross-conformal predictive distributions slightly to remove their fuzziness [140].

Instead of (6.4) and (6.8) we use their crisp modifications

$$Q(z_1, \dots, z_n, (x, y)) := \begin{cases} \frac{i}{n-m} & \text{if } y \in (C_{(i)}, C_{(i+1)}) \text{ for } i \in \{0, 1, \dots, n-m\} \\ \frac{i}{n-m} & \text{if } y = C_{(i)} \text{ and } y \neq C_{(i+1)} \text{ for } i \in \{1, \dots, n-m\} \end{cases} \tag{6.12}$$

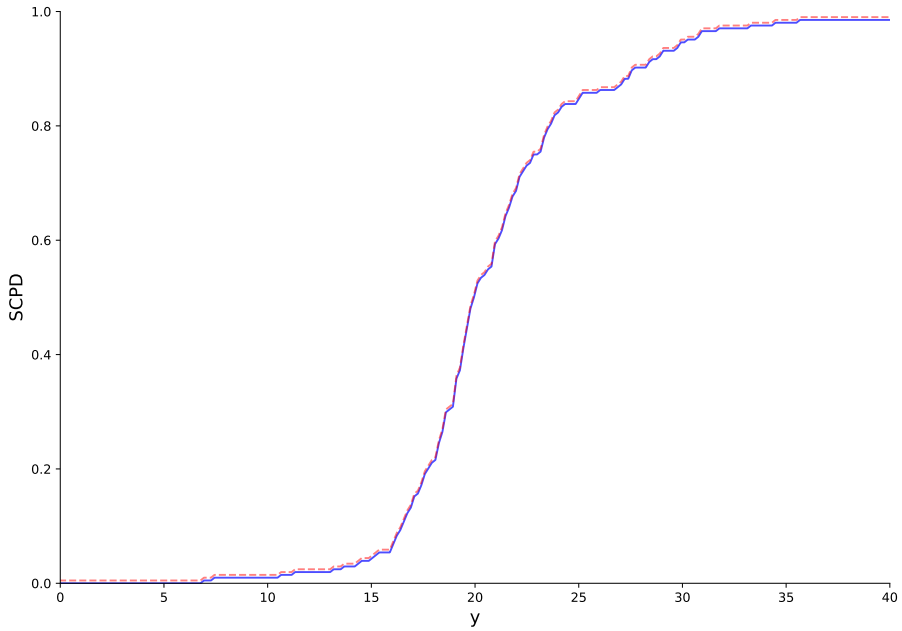

FIGURE 6.1: The split conformal predictive distribution for a random test object in the `Boston Housing` dataset (described in Section 6.5), the Least Squares underlying algorithm, and a random $50\% : 50\%$ split of the training sequence into proper training and calibration sequences. The blue solid line corresponds to $\tau = 0$ and the red dashed line to $\tau = 1$.

and

$$Q(z_1, \dots, z_n, (x, y)) := \begin{cases} \frac{i}{n} & \text{if } y \in (C_{(i)}, C_{(i+1)}) \text{ for } i \in \{0, 1, \dots, n\} \\ \\ \frac{i}{n} & \text{if } y = C_{(i)} \text{ and } y \neq C_{(i+1)} \text{ for } i \in \{1, \dots, n\}, \end{cases} \tag{6.13}$$

respectively; these modifications no longer depend on τ, and the convention for $y = C_{(i)}$ does not affect the value of CRPS. In cases where the equation (6.6) or its analogue for the CCPS are difficult to solve, we can instead use the following crisp modifications of (6.1) and (6.7), respectively:

$$Q(z_1, \dots, z_n, (x, y)) := \frac{1}{n - m} \left| \{ i = m + 1, \dots, n \mid \alpha_i \leq \alpha^y \} \right|,$$

$$Q(z_1, \dots, z_n, (x, y)) := \frac{1}{n} \sum_{k=1}^{K} \left| \left\{ i \in S_k \mid \alpha_{i,k} \leq \alpha_k^y \right\} \right|.$$

The last equation, defining a crisp CCPS, can be rewritten as

$$Q(z_1, \dots, z_n, (x, y)) = \sum_{k=1}^{K} \frac{|S_k|}{n} p_k^y$$

(cf. (6.10)), where the separate "p-values" for each fold are now defined as

$$p_k^y := \frac{1}{|S_k|} \left| \{ i \in S_k \mid \alpha_{i,k} \le \alpha_k^y \} \right|$$

(they, however, do not satisfy any validity properties).

6.5 Experiments

The purpose of this section is to compare the predictive performance of SCPS and CCPS and to recommend the choice of the parameter K for CCPS.

In our experiments we use five well-known benchmark datasets, namely `Boston Housing`, `Diabetes`, `Yacht Hydrodynamics`, `Wine Quality`, and `Condition Based Maintenance of Naval Propulsion Plants` (abbreviated to `Naval Propulsion`) available at http://scikit-learn.org/stable/datasets/ (the first two) and the UCI Machine Learning repository [33] (the other ones). The first three datasets are small: `Boston Housing` consists of 506 observations, `Diabetes` of 442 observations, and `Yacht Hydrodynamics` of 308 observations; for them we use test sequences of length $l := 100$. The `Wine Quality` dataset consists of 6497 observations, and we use test sequences of length $l := 1000$. Finally, the `Naval Propulsion` dataset consists of 11,934 observations, and we use test sequences of length $l := 4000$.

Given a training sequence $(z_1, \dots, z_n)$ (where $n \in \{406, 342, 208, 5497, 7934\}$) and a test sequence $(z_{n+1}, \dots, z_{n+l})$, the quality of prediction is represented by the distribution of $\mathrm{CRPS}(F_i, y_i)$, $i = n+1, \dots, n+l$, where F_i is the predictive distribution for the label y_i of the test object x_i. As already mentioned, the length l of the test sequence is 100, 1000, or 4000 in our experiments.

In order to obtain boxplots less affected by the split of each dataset into a training and test sequence and by the random split of each training sequence into a training sequence

proper and a calibration sequence (in the case of SCPS) or K folds (in the case of CCPS), we use the procedure describe below (see experiments section in [143] for more details):

- Each dataset is randomly permuted 10 times.
- The last l observations of each permutation are used for testing and the rest for training.
- The first m observations in the training sequence are used as training sequence proper in the case of SCPS and consecutive blocks of the training sequence are used as the K folds in the case of CCPS (using the `scikit-learn` `KFold` procedure with no randomization).
- The boxplots in all figures given below are indexed by the fractions m/n of the training sequence used as the training sequence proper (in the case of SCPS) or by the numbers K of folds (in the case of CCPS).
- For each split and each boxplot we find the l values $\mathrm{CRPS}(F_i, y_i)$ for all test observations (the same test sequence is used for each split); the resulting boxplot is based on all $10\,l$ numbers.

In all cases the SCPS and CCPS use the cross-conformity measure (a special case of (6.5))

$$A(z_1, \dots, z_m, (x, y)) := y - \hat{y}, \tag{6.14}$$

where $\hat{y}$ is the prediction computed using the underlying algorithm U for the label of x based on $z_1, \dots, z_m$ as training sequence. (Remember that each cross-conformity measure is also a split conformity measure.) Similarly to the CPS based on Least Squares and Kernel Ridge Regression (as discussed in previous chapters — Chapters 5, 4), this procedure is far from universal and can be expected to be efficient only for data that is not too far from being homoscedastic.

Notice that, the SCPS is no longer provably calibrate because parameter tuning depends on the full training sequence), we therefore also check its validity in our experiments. To check the validity of both SCPS and CCPS, we run Algorithm 6.5 replacing

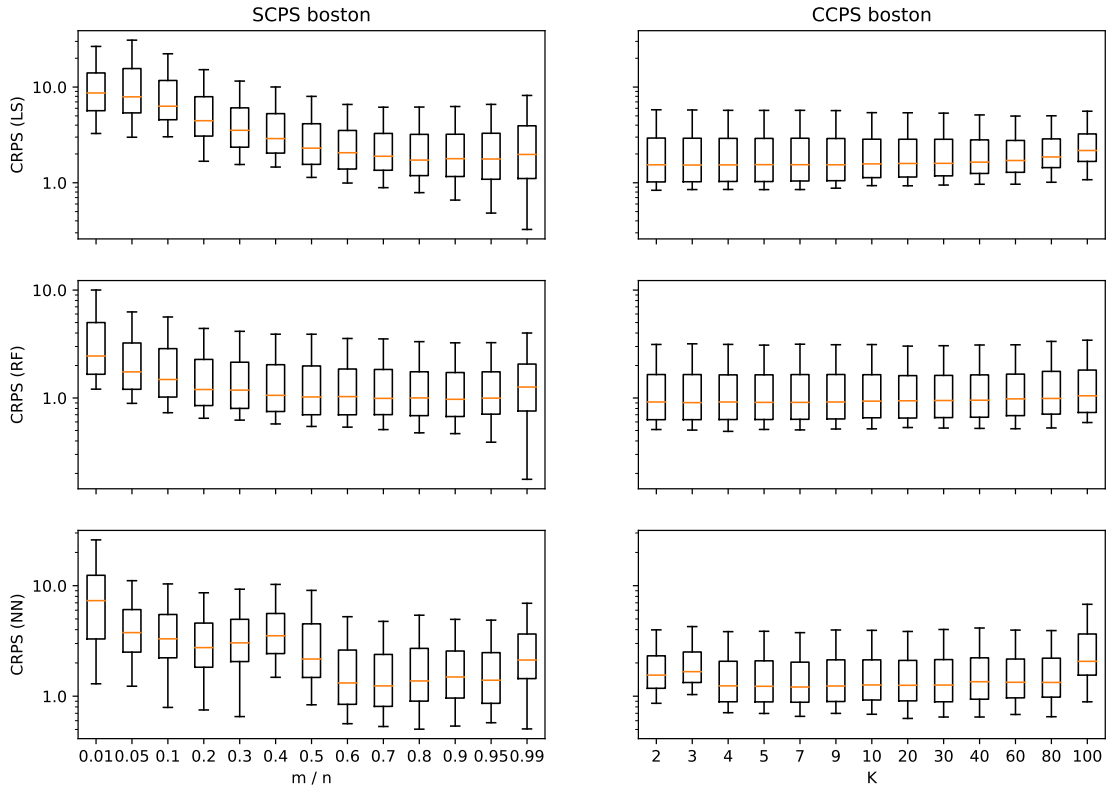

FIGURE 6.2: The performance of the SCPS (left panel) and CCPS (right panel) on the `Boston Housing` dataset using Least Squares (LS), Random Forest (RF), and Neural Networks (NN) as the underlying algorithms, as indicated on the left. The vertical axis uses the log scale and gives the CRPS. Left panel: the numbers on the horizontal axis are the fractions m/n of the training sequence used as the training sequence proper. Right panel: the numbers on the horizontal axis are the numbers K of folds.

$\mathrm{CRPS}(F_i, y_i)$ with $F_i(y_i)$ and replacing boxplots with plots, such as those in Figure 6.7 (described in detail at the end of this section).

The `Boston Housing` dataset consists of 506 observations each with 14 attributes (describing an area of Boston) and a real-valued label (median house price in that area). Figure 6.2 shows the performance of the SCPS and CCPS.

The horizontal axis in the left panel is labelled by $\alpha \approx m/n$; the values of α used in our experiments are between 0.1 and 0.9, plus a few more extreme values. For a given value of α we set $m := \lfloor \alpha n \rfloor$. We follow the approach in [141]. The CRPS loss is computed for the (crisp) SCPS based on (6.14) and the three underlying algorithms on each observation in the test sequence; as described above, we then represent the resulting 1000 CRPS losses as a boxplot. We can see a characteristic U-shape (especially pronounced on the left);

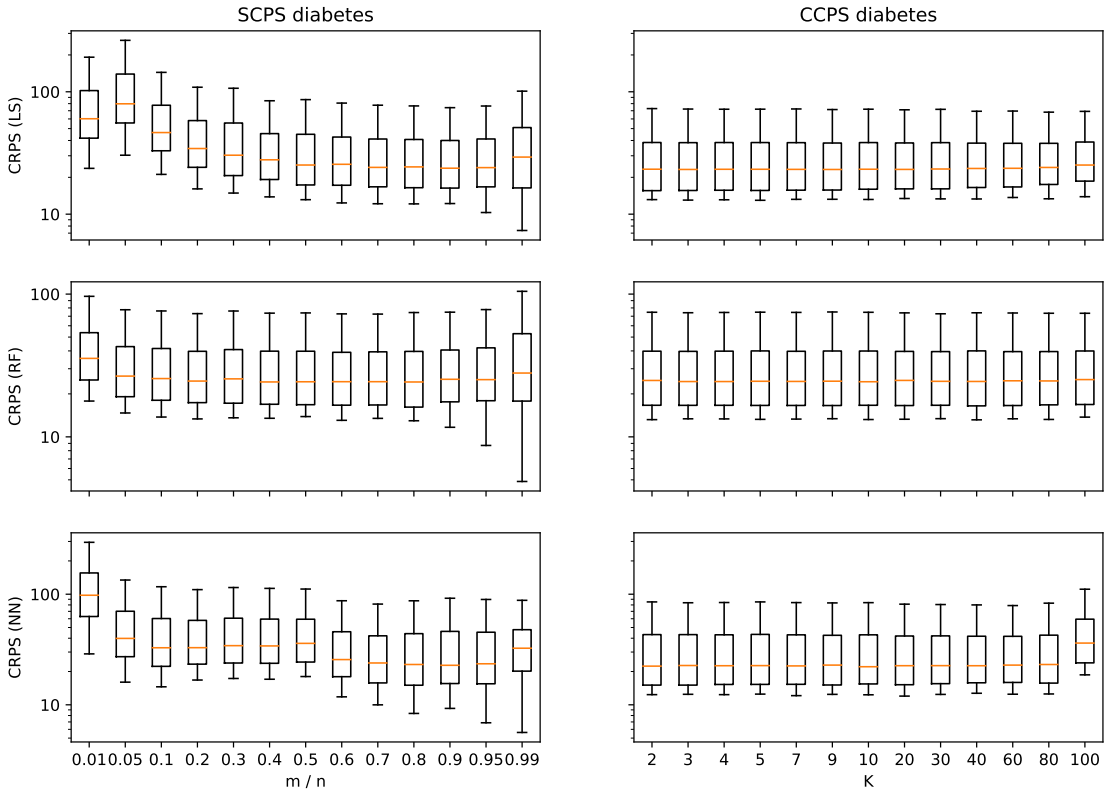

FIGURE 6.3: The analogue of Figure 6.2 for the `Diabetes` dataset.

small m/n lead to a significant increase in the CRPS loss, and large m/n lead to a slight increase in the CRPS loss but a significant increase in its variability (the rightmost box and its whiskers tend to be longer).

The right panel of Figure 6.2 is similar to the left panel, but now we use the CCPS and label the horizontal axis by the number K of folds. The usual advice in cross validation is to use $K \in \{5, 10\}$, and these two values produce reasonable results. In fact, the results are remarkably stable and barely depend on K [140].

The `Diabetes` dataset consists of 10 physiological measures on 442 patients, and the label indicates disease progression after one year. Figure 6.3 is the analogue of Figure 6.2 for this dataset. We can see the same tendencies, with $K \in \{5, 10\}$ still being reasonable numbers of folds for CCPS [140].

The `Yacht Hydrodynamics` is the smallest of our datasets. It consists of 7 attributes including the basic hull dimensions and the boat velocity for 308 experiments, and the task is to predict the residuary resistance of sailing yachts. Figure 6.4 suggests that the

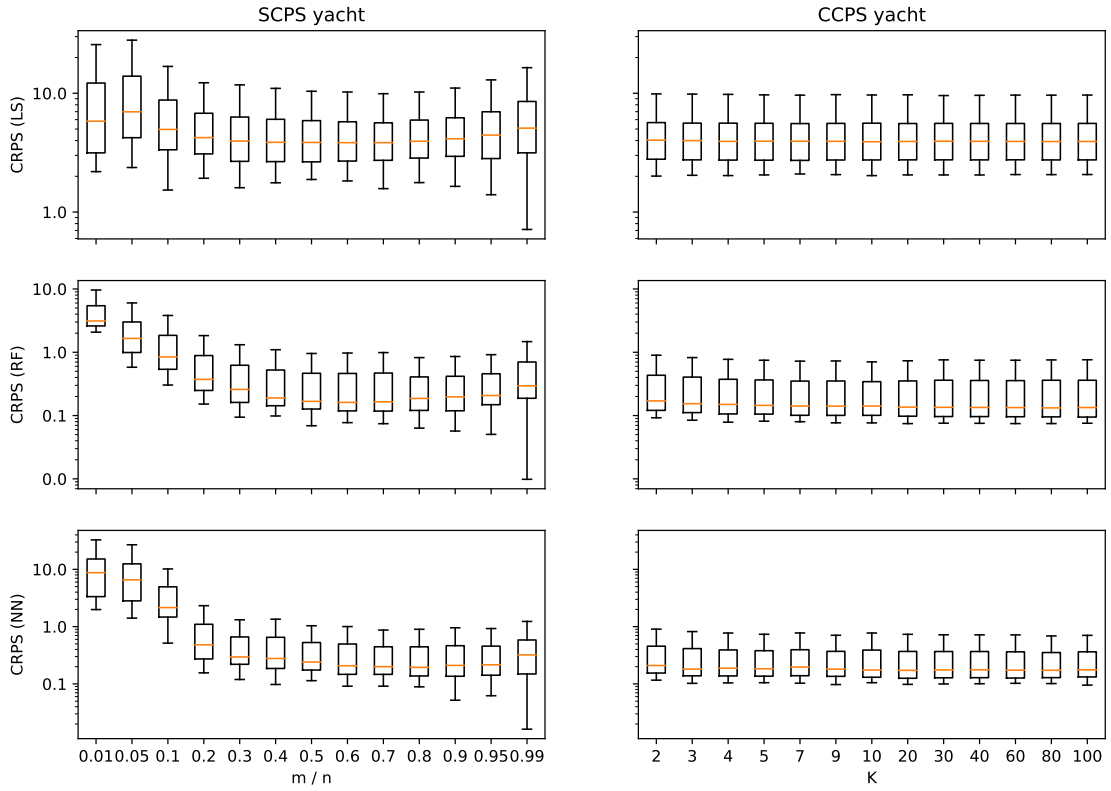

FIGURE 6.4: The analogue of Figure 6.2 for the `Yacht Hydrodynamics` dataset.

behavior shown in Figures 6.2 and 6.3 is in fact typical of small datasets.

The `Wine Quality` dataset has information about 1599 red wines and 4898 white wines. We merge these two groups creating another attribute taking two values, 0 for white and 1 for red. The label is the quality of wine expressed as a score between 0 and 10. (The most common labels are 5 and 6, labels 3 and 9 are very uncommon, and labels 0 and 1 are absent [140]).

Figure 6.5 is qualitatively similar to Figures 6.2 and 6.3. The shape of the plots for SCPS suggests that we need a reasonable length $n - m$ of the calibration sequence, such as 100 or 200, since it determines the granularity of the predictive distributions (see, e.g., [140]): as we have already mentioned in connection with (6.4), the thickness of the predictive distribution is $\frac{1}{n-m+1}$. Increasing the length of the calibration sequence further does not improve the predictive performance significantly, and starts hurting it when the training sequence proper becomes too short.

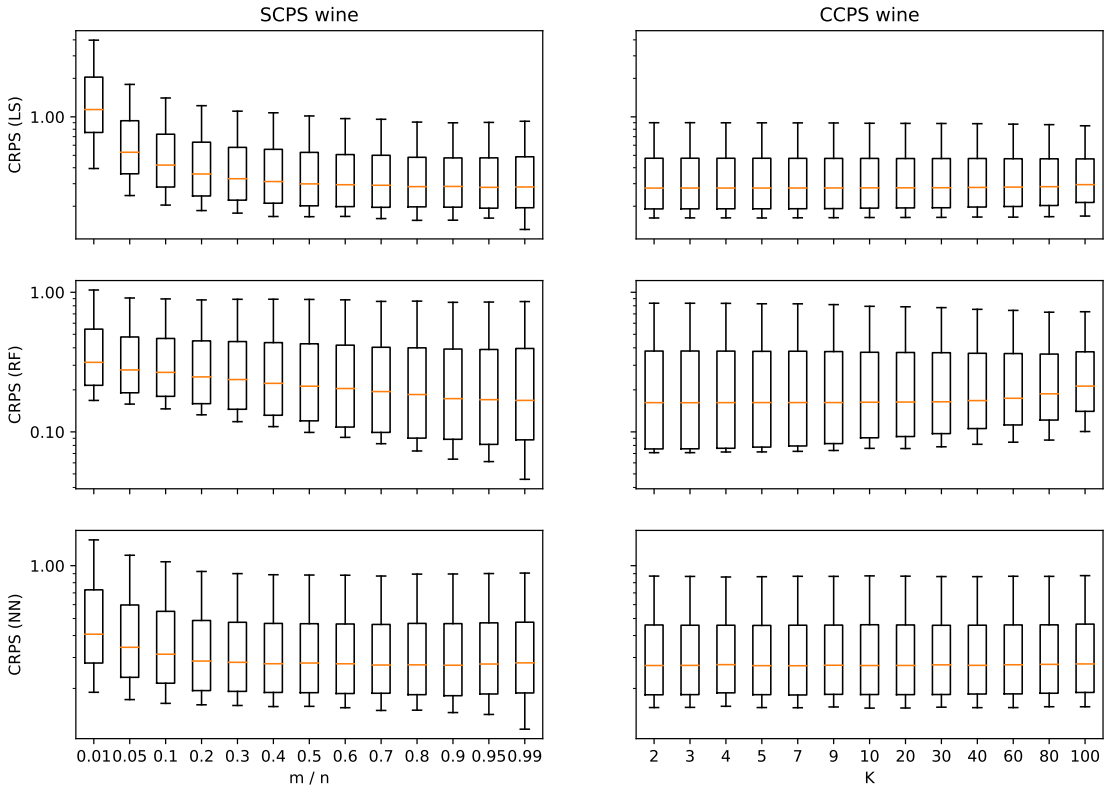

FIGURE 6.5: The analogue of Figure 6.2 for the `Wine Quality` dataset.

Figure 6.6 reports the results for the largest dataset that we use, `Naval Propulsion`. It contains information about 11,934 simulated experiments, each described by 16 attributes, and the task is to predict the Gas Turbine Compressor decay state coefficient for a propulsion plant. Here we observe the same general behavior.

The best results presented in Figures 6.2–6.6 are summarized in Table 6.1. Namely, the table reports the median CRPS losses shown in Figures 6.2–6.5 obtained by optimizing the parameters m/n in the case of SCPS and K in the case of CCPS. In the majority of cases CCPS perform better than SCPS. But what is even more important, CCPS are much less sensitive to choosing their parameter K, and so the best results given in Table 6.1 are in fact typical for them. In all our experiments, it is safe to choose any of the standard values for the number K of folds in the range from 5 to 10.

A natural question is whether the CCPS satisfy the property of validity 4.4.1 at least approximately; remember that there are no theoretical validity results for cross-conformal

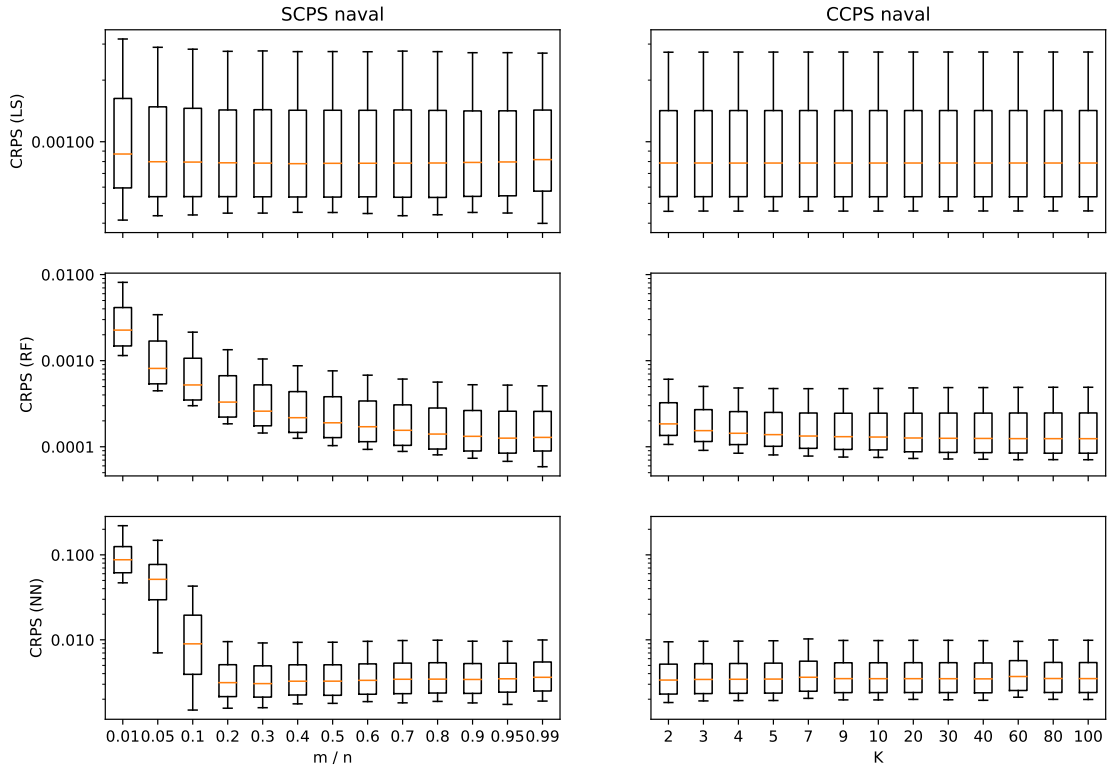

FIGURE 6.6: The analogue of Figure 6.2 for the `Naval Propulsion` dataset.

predictors, and it has been demonstrated theoretically [135, Appendix A] and experimentally [81] that a loss of validity is possible. Figure 6.7 (right panel) shows the distribution of the values (6.13) for `Boston Housing` and $K = 5$, where $z_1, \dots, z_n$ is the training sequence, and (x, y) range over the elements of the test sequence. Figure 6.7 (right panel) provides the *calibration curves*, which are the sets of points $(\alpha, F(\alpha))$, $\alpha \in (0, 1)$ ranging over the possible significance levels and $F(\alpha)$ being the percentage of the values $Q(z_1, \dots, z_n, (x, y))$ for (x, y) in the test sequence that do not exceed α [140]. The right panels of Figures 6.8, 6.9, 6.10, and 6.11 are the analogues for the `Diabetes`, `Yacht Hydrodynamics`, `Wine Quality`, and `Naval Propulsion` datasets, respectively. Under perfect validity **R2** in 4.4.1 and an infinitely long test sequence, the calibration curves should be the diagonals shown as dashed lines on both panels of Figures 6.7–6.11; the actual calibration curves are fairly close. The calibration curves for other K are roughly similar. As mentioned earlier, we also give calibration results for SCPS (in the left panels and with $m/n \approx 0.5$).

TABLE 6.1: Best results for the median CRPS loss for SCPS and CCPS for the five datasets and three underlying algorithms.

Dataset	underlying algorithm	SCPS	CCPS
Boston Housing	Least Squares	1.726	1.533
Boston Housing	Random Forest	0.972	0.906
Boston Housing	Neural Network	1.240	1.211
Diabetes	Least Squares	23.74	23.18
Diabetes	Random Forest	24.23	24.33
Diabetes	Neural Network	22.76	22.10
Yacht Hydrodynamics	Least Squares	3.840	3.910
Yacht Hydrodynamics	Random Forest	0.1615	0.1322
Yacht Hydrodynamics	Neural Network	0.1944	0.1725
Wine Quality	Least Squares	0.2810	0.2771
Wine Quality	Random Forest	0.1681	0.1618
Wine Quality	Neural Network	0.2711	0.2693
Naval Propulsion	Least Squares	0.0007812	0.0007866
Naval Propulsion	Random Forest	0.0001259	0.0001242
Naval Propulsion	Neural Network	0.003051	0.003360

Not only is the efficiency of the CCPS with respect to the CRPS loss better than that of the SCPS, it can also be argued that the CCPS may be safer from the point of view of validity. Suppose that, for some reason, we would like to avoid randomization and use (6.12) (in the case of SCPS) or (6.13) (in the case of CCPS) instead of (6.4) or (6.8), respectively. The CCPS is still empirically valid in our experiments, even in the extreme case of $K = 100$. On the other hand, when using (6.12) in place of (6.4), the SCPS lose not only theoretical but also empirical validity. For example, for `Boston Housing` and $m/n = 0.99$ (the right end of the horizontal axis in the left panel of Figure 6.2), the length of the calibration sequence is 4, and so the empirical predictive distribution (6.12) only takes values in $\{0, 0.25, 0.5, 0.75, 1\}$; the distribution of its values at the true labels is clearly very different from being uniform.

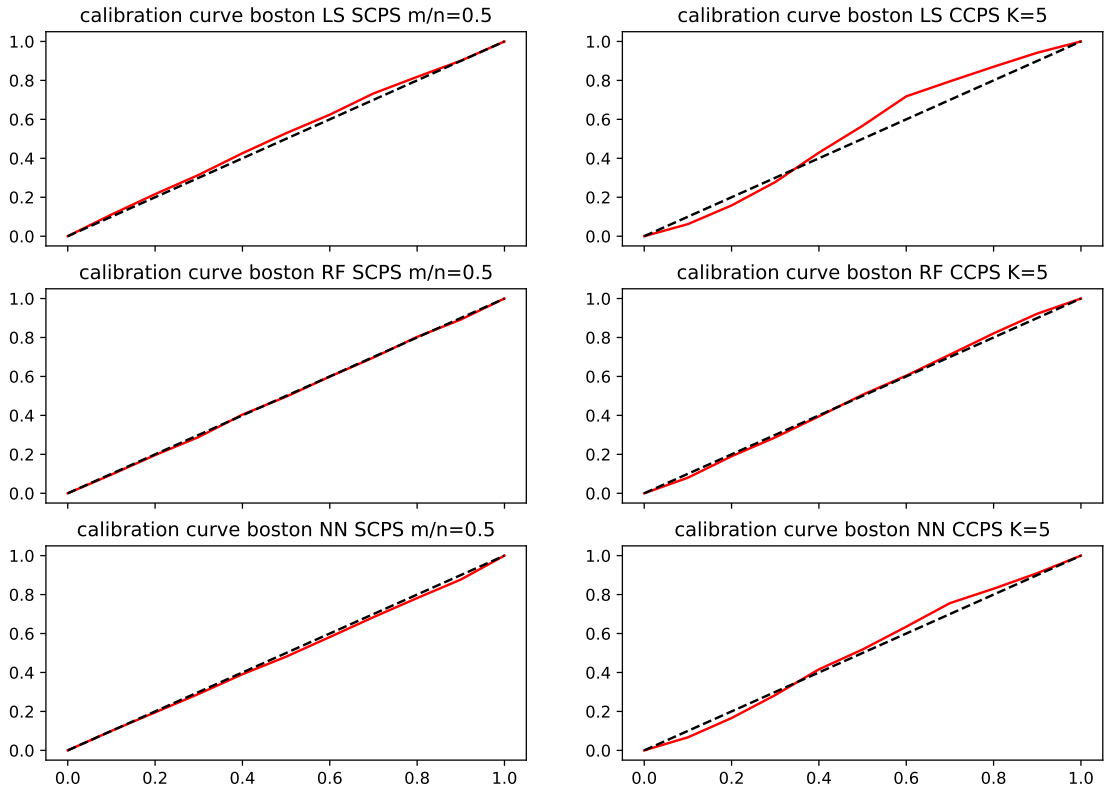

FIGURE 6.7: The calibration curves (i.e., the distributions of $Q(z_1, \ldots, z_n, (x, y))$ over the test sequence) for the SCPS and CCPS on the `Boston Housing` dataset.

6.6 Conclusion

The main contribution of this chapter is extension of the study of probabilistic regression to address the relative computational inefficiency of classical conformal predictors (published paper [140]). Two novel computationally efficient versions of conformal predictive systems are introduced — split conformal predictive systems (SPC) and cross-conformal predictive system (CCPS). The main advantage of split conformal predictive systems is their guaranteed validity. The main advantage of cross-conformal predictive systems is greater predictive efficiency, for such predictive systems validity only holds empirically and in the absence of excessive randomization. The main aim of this chapter is to define and study computationally efficient versions of CPS without any restrictions related to the underlying algorithm.

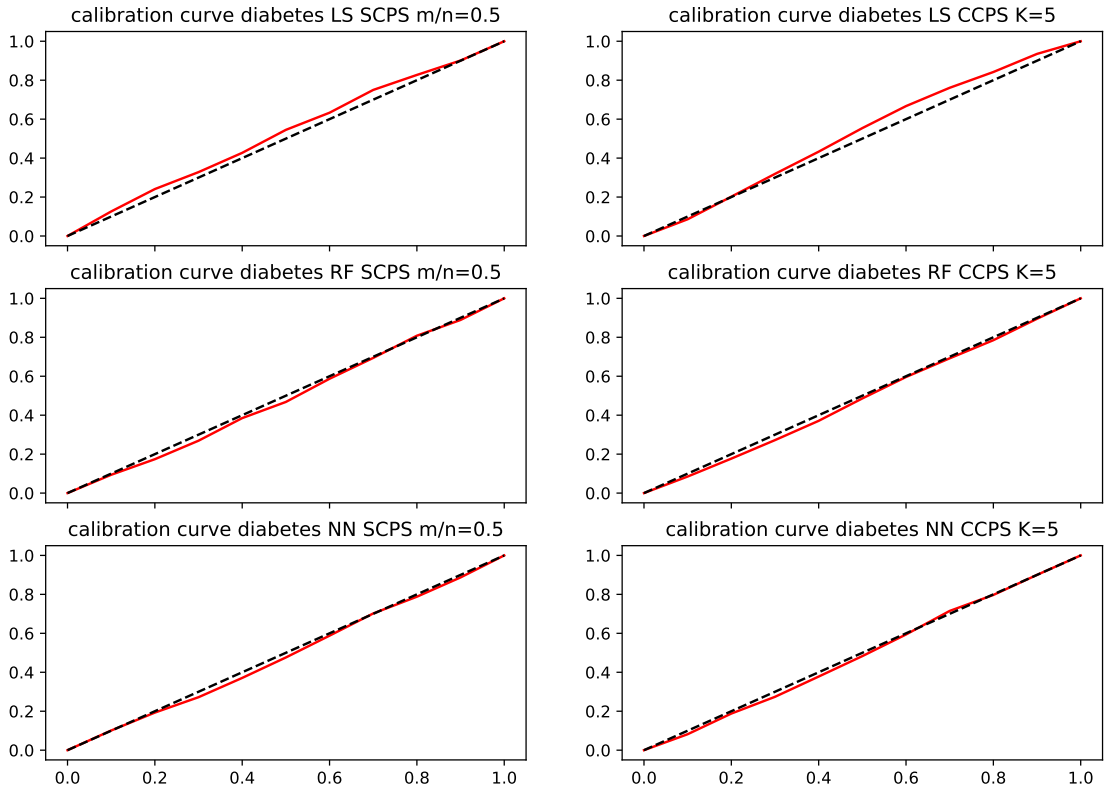

FIGURE 6.8: The analogue of Figure 6.7 for the `Diabetes` dataset.

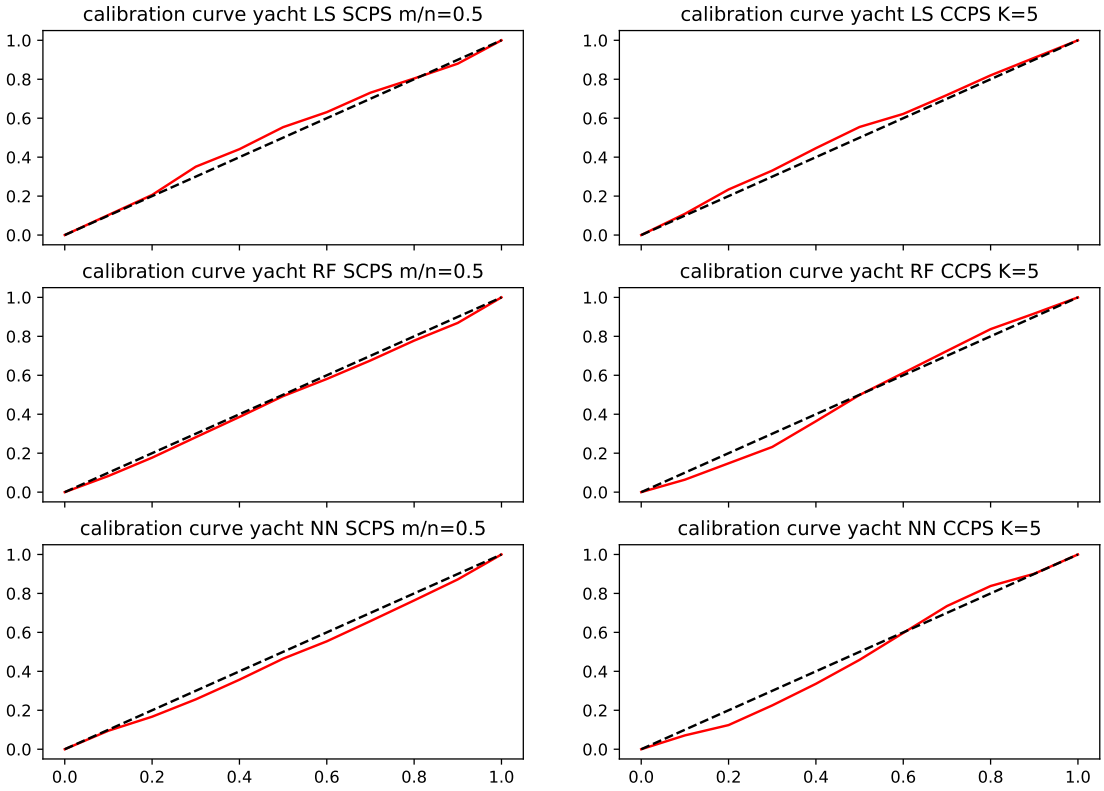

FIGURE 6.9: The analogue of Figure 6.7 for the `Yacht Hydrodynamics` dataset.

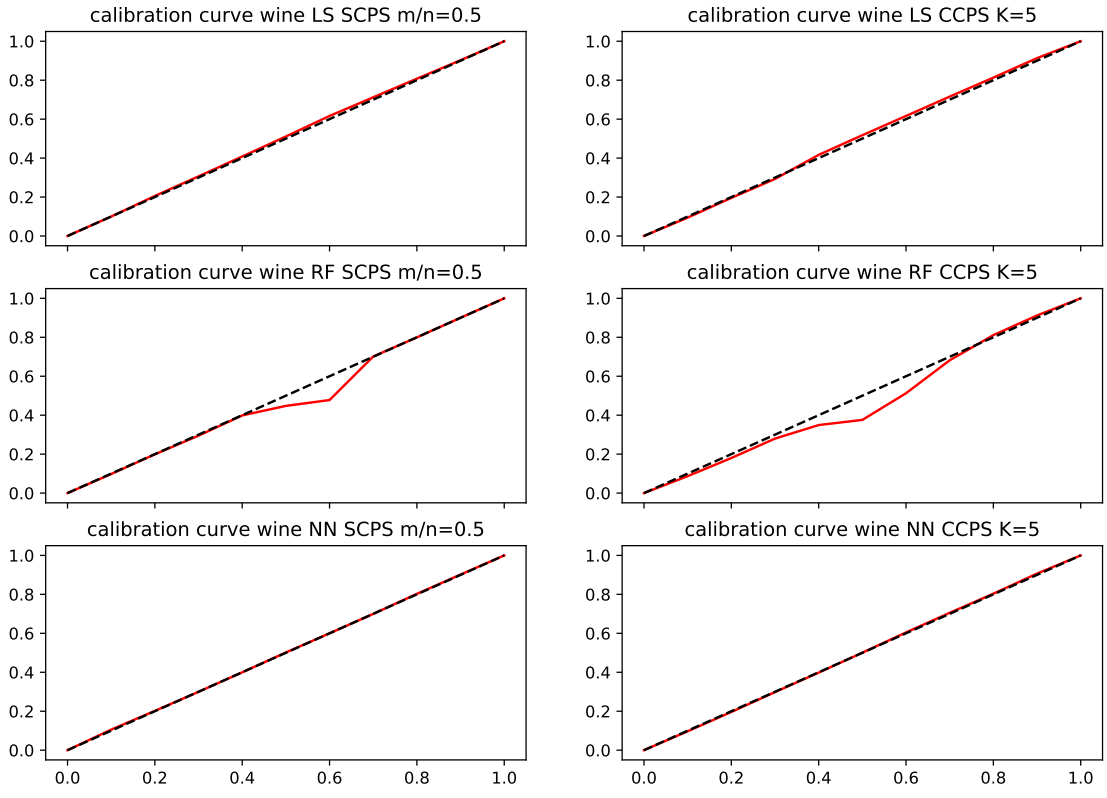

FIGURE 6.10: The analogue of Figure 6.7 for the `Wine Quality` dataset.

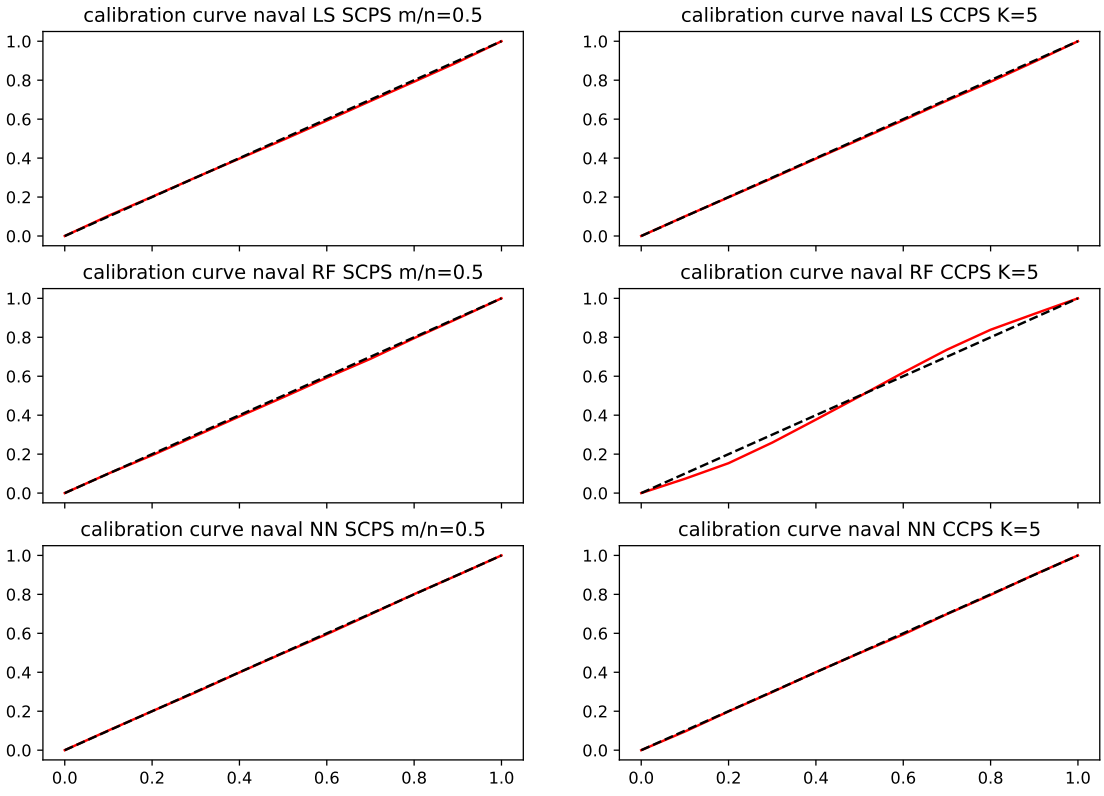

FIGURE 6.11: The analogue of Figure 6.7 for the `Naval Propulsion` dataset.

Chapter 7

Conclusion

This thesis expands and advances machine learning research in probabilistic prediction and introduces novel methods of producing well-calibrated probabilistic predictions for both machine learning classification and regression problems.

Chapter 1 outlines standard machine learning techniques, introduces the main concepts in probabilistic prediction and describes probabilistic prediction methods such as conformal prediction and Venn prediction. Chapter 2 introduces probabilistic machine learning and describes some of the calibration methods, both classical and modern, including recently developed computationally efficient algorithms IVAP (inductive Venn–Abers predictor) and CVAP (cross Venn–Abers predictor). The classical methods such as Platt's scaling and isotonic regression have disadvantages and do not produce theoretical guarantees of validity of predictions. In addition, modern deep learning computer vision models rely on convolution-based architectures that result in overconfident predictions. Whilst innovations in deep learning continue to maximise predictive accuracy this is often at the expense of predictive quality — whilst accuracy is being maximised, the incorrect predictions are often accompanied by overconfidence resulting in significant risk of wrong decisions, especially in critical applications such as healthcare and self-driving cars. There is therefore a clear need for non-parametric machine learning methods that are able to produce well-calibrated class probabilities for classification and valid prediction intervals for regression.

Chapter 3 introduces a novel method of probabilistic prediction in multi-class classification setting. The multi-class probabilistic prediction problem is solved via dividing it

into pairwise classification problems, calibrating binary class probabilities and then converting binary class probabilities into mutli-class probabilities in order to assign each test object to one of the k classes. Such approach allows to compute probabilistic loss metrics such as the Log loss and the Brier loss and compare them both across various underlying machine learning classification methods, as well across results obtained from applying various calibration methods. By using IVAP and CVAP to calibrate binary classifiers and then combining calibrated binary classification scores to obtain well-calibrated multi-class probabilities, the resulting multi-class probabilistic classification method results in more accurate probabilistic predictions that those obtained from both the underlying machine learning classifiers as well as existing calibration methods such as Platt's scaling and isotonic regression applied in multi-class classification setting.

Chapter 4 introduces non-parametric approach to predictive distribution functions using conformal prediction. In statistics, the theory of predictive distributions is based on the assumption that samples are generated from a parametric model. This chapter's novel contribution is the non-parametric extension of statistical "prediction confidence distributions" using only limited assumptions customary of machine learning, namely that the observations are generated using the IID model. Unlike in the parametric approach of statistical predictive distributions, this non-parametric approach to predictive distributions does not require for the data model to be specified. This approach results in predictive distribution functions that are always valid for IID observations in terms of guaranteed coverage. The advantage of predictive distribution functions over the usual conformal prediction intervals is that conformal predictive distributions contain more information - a conformal predictive distribution Q_n can produce a plethora of prediction intervals corresponding to each confidence level $1 - \epsilon$.

Chapter 5 extends the study of probabilistic regression and combines conformal predictive distributions (CPDs) with kernel methods to derive kernelized versions of the algorithms described in Chapter 4. Kernelized versions of conformal predicitive distributions are studied theoretically to determine their computational efficiency. Experimental study of the predictive efficiency demonstrates important advantages of the kernelized versions of CPDs and shows that universal (Laplacian) kernel works remarkably well.

Chapter 6 extends the study of probabilistic regression to address the relative computational inefficiency of classical conformal predictors. Two novel computationally efficient conformal predictive systems are introduced — split conformal predictive systems (SPC) and cross-conformal predictive system (CCPS). Split conformal predictive systems provide guaranteed validity, whilst the main advantage of cross-conformal predictive systems is greater predictive efficiency. For cross-conformal predictive systems validity only holds empirically and in the absence of excessive randomization. The main aim of this chapter is to define and study computationally efficient versions of CPS without any restrictions related to the underlying algorithm.

Bibliography

[1] Anastasios N. Angelopoulos and Stephen Bates. *A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification*. 2021 (Cited on page 28).

[2] Eugene A. Asarin. "Some properties of Kolmogorov σ-Random finite sequences." In: *Theory of Probability & Its Applications* 32.3 (Jan. 1988), pp. 507–508. DOI: 10.1137/1132070 (Cited on page 25).

[3] Thomas Augustin and Frank P. A. Coolen. "Nonparametric predictive inference and interval probability." In: *Journal of Statistical Planning and Inference* 124.2 (Sept. 2004), pp. 251–272. DOI: 10.1016/j.jspi.2003.07.003 (Cited on page 91).

[4] Anand Avati, Tony Duan, Sharon Zhou, Kenneth Jung, Nigam H. Shah, and Andrew Ng. "Countdown regression: sharp and calibrated survival predictions." In: *CoRR* abs/1806.08324 (June 2018). arXiv: 1806.08324 [cs.LG] (Cited on page 22).

[5] Miriam Ayer, Daniel Brunk, George Ewing, William Reid, and Edward Silverman. "An empirical distribution function for sampling with incomplete information." In: *The Annals of Mathematical Statistics* 26.4 (Dec. 1955), pp. 641–647. DOI: 10.1214/aoms/1177728423 (Cited on pages 36, 46).

[6] Vineeth Balasubramanian, Shen-Shyang Ho, and Vladimir Vovk. *Conformal Prediction for Reliable Machine Learning: Theory, Adaptations, and Applications*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2014. ISBN: 978-0-12-398537-8 (Cited on pages 24, 122).

[7] R. E. Barlow, D. J. Bartholomew, J. M. Bremner, and H. D. Brunk. *Statistical Inference under Order Restrictions; the Theory and Application of Isotonic Regression*. Wiley, 1972 (Cited on pages 35, 36, 56).

[8] Eric B. Baum and Frank Wilczek. "Supervised learning of probability distributions by neural networks." In: *Neural Information Processing Systems*. NIPS'87. Cambridge, MA, USA: American Institute of Physics, 1987, pp. 52–61. URL: https://proceedings.neurips.cc/paper/1987/file/eccbc87e4b5ce2fe28308fd9f2a7baf3-Paper.pdf (Cited on page 43).

[9] Leo Breiman. "Random forests." In: *Machine Learning* 45.1 (Oct. 2001), pp. 5–32. DOI: 10.1023/A:1010933404324. URL: https://link.springer.com/article/10.1023/A:1010933404324 (Cited on page 21).

[10] Glenn W. Brier. "Verification of forecasts expressed in terms of probability." In: *Monthly Weather Review* 78.1 (1950), pp. 1–3 (Cited on pages 24, 45).

[11] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. "API design for machine learning software: experiences from the scikit-learn project." In: *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*. 2013, pp. 108–122 (Cited on page 59).

[12] Evgeny Burnaev and Vladimir Vovk. "Efficiency of conformalized ridge regression." In: *Proceedings of The 27th Conference on Learning Theory*. Vol. 35. Proceedings of Machine Learning Research. Barcelona, Spain: PMLR, June 2014, pp. 605–622. URL: http://proceedings.mlr.press/v35/burnaev14.pdf (Cited on page 85).

[13] Evgeny Burnaev and Vladimir Vovk. "Efficiency of conformalized ridge regression." In: *Proceedings of The 27th Conference on Learning Theory*. Ed. by Maria Florina Balcan, Vitaly Feldman, and Csaba Szepesvári. Vol. 35. Proceedings of Machine Learning Research. 2014, pp. 605–622 (Cited on page 86).

[14] Ben Van Calster and Andrew J. Vickers. "Calibration of risk prediction models." In: *Medical Decision Making* 35.2 (Aug. 2014), pp. 162–169. DOI: 10.1177/0272989x14547233 (Cited on page 30).

[15] Lars Carlsson, Martin Eklund, and Ulf Norinder. "Aggregated conformal prediction." In: *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. Vol. 437. IFIP Advances in Information and Communication Technology. Berlin, Heidelberg: Springer International Publishing, 2014, pp. 231–240. ISBN: 978-3-662-44722-2. DOI: 10.1007/978-3-662-44722-2_25 (Cited on page 130).

[16] Rich Caruana and Alexandru Niculescu-Mizil. "An empirical comparison of supervised learning algorithms." In: *Proceedings of the 23rd International Conference on Machine Learning — ICML '06*. Vol. 2006. ICML '06. Pittsburgh, Pennsylvania, USA: Association for Computing Machinery, June 2006, pp. 161–168. ISBN: 1595933832. DOI: 10.1145/1143844.1143865 (Cited on page 50).

[17] Rich Caruana and Alexandru Niculescu-Mizil. "Predicting good probabilities with supervised learning." In: *Proceedings of the 22nd International Conference on Machine Learning*. ICML '05. New York, NY, USA: Association for Computing Machinery, 2005, pp. 625–632. ISBN: 1595931805. DOI: 10.1145/1102351.1102430 (Cited on pages 30, 50, 70).

[18] Samprit Chatterjee and Ali S. Hadi. *Sensitivity Analysis in Linear Regression*. Wiley Series in Probability and Statistics. John Wiley & Sons, Inc., Mar. 1988. ISBN: 9780470316764. DOI: 10.1002/9780470316764 (Cited on pages 81, 83, 84).

[19] Gemai Chen. "Empirical processes based on regression residuals: Theory and applications." PhD thesis. Simon Fraser University, Aug. 1991. URL: https://summit.sfu.ca/item/4526 (Cited on page 94).

[20] Tianqi Chen and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System." In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 785–794 (Cited on page 59).

[21] Alexey Chervonenkis. *Measures of Complexity : Festschrift for Alexey Chervonenkis*. Ed. by Vladimir Vovk, Harris Papadopoulos, and Alexander Gammerman. Cham: Springer International Publishing, 2015. ISBN: 978-3-319-21852-6. DOI: 10.1007/978-3-319-21852-6 (Cited on page 100).

[22] Corinna Cortes and Vladimir Vapnik. "Support-vector networks." In: *Machine Learning* 20.3 (Sept. 1995), pp. 273–297. ISSN: 0885-6125. DOI: 10.1023/A:1022627411411 (Cited on pages 21, 33).

[23] Thomas Cover and Patrick Hart. "Nearest neighbor pattern classification." In: *IEEE Transactions on Information Theory* 13.1 (Sept. 1967), pp. 21–27. ISSN: 0018-9448. DOI: 10.1109/TIT.1967.1053964 (Cited on page 21).

[24] D.R. Cox. "Some problems connected with statistical inference." In: *Annals of Mathematical Statistics* 29.2 (June 1958), pp. 357–372. DOI: 10.1214/aoms/1177706618. URL: https://projecteuclid.org/euclid.aoms/1177706618 (Cited on page 104).

[25] Mikhail Dashevskiy and Zhiyuan Luo. "Reliable probabilistic classification and its application to internet traffic." In: *International Conference on Intelligent Computing*. ICIC '08. Springer. Shanghai, China: Springer-Verlag, Sept. 2008, pp. 380–388. ISBN: 978-3-540-87440-9. DOI: 10.1007/978-3-540-87442-3_48 (Cited on page 46).

[26] A. P. Dawid. "The Well-Calibrated Bayesian." In: *Journal of the American Statistical Association* 77.379 (Sept. 1982), pp. 605–610. DOI: 10.1080/01621459.1982.10477856 (Cited on page 23).

[27] Alexander Philip Dawid. "Present position and potential developments: some personal views: Statistical theory: the prequential approach." In: *Journal of the Royal Statistical Society*. A (General) 147.2 (Mar. 1984), pp. 278–292. ISSN: 00359238. DOI: 10.2307/2981683 (Cited on pages 99, 104).

[28] Alexander Philip Dawid and Vladimir Vovk. "Prequential probability: principles and properties." In: *Bernoulli* 5.1 (Feb. 1999), pp. 125–162. DOI: 10.2307/3318616. URL: https://projecteuclid.org/download/pdf_1/euclid.bj/1173707098 (Cited on pages 99, 104).

[29] Morris H. DeGroot and Stephen E. Fienberg. "The comparison and evaluation of forecasters." In: *Journal of the Royal Statistical Society: Series D (The Statistician)* 32.1/2 (Mar. 1983), pp. 12–22. DOI: 10.2307/2987588 (Cited on page 45).

[30] Arthur P. Dempster. "On direct probabilities." In: *Journal of the Royal Statistical Society*. B (Methodological) 25.1 (1963). ISSN: 00359246. URL: http://www.jstor.org/stable/2984545 (Cited on page 91).

[31] Monroe Donsker. "Justification and extension of Doobs heuristic approach to the Kolmogorov–Smirnov theorems." In: *Annals of Mathematical Statistics* 23.2 (Nov. 1952), pp. 277–281. DOI: 10.1214/aoms/1177729445. URL: https://projecteuclid.org/euclid.aoms/1177729445 (Cited on page 92).

[32] Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. "CatBoost: gradient boosting with categorical features support." In: *ArXiv* abs/1810.11363 (2018) (Cited on page 59).

[33] Dheeru Dua and Casey Graff. *UCI Machine Learning Repository*. http://archive.ics.uci.edu/ml. University and o, 2017 (Cited on pages 38–40, 59, 60, 62, 63, 65, 117, 132).

[34] Tony Duan, Anand Avati, Daisy Yi Ding, Sanjay Basu, Andrew Y. Ng, and Alejandro Schuler. "NGBoost: Natural gradient boosting for probabilistic prediction." In: *CoRR* abs/1910.03225 (Oct. 2019). arXiv: 1910.03225 [cs.LG] (Cited on pages 23, 68).

[35] Bradley Efron. "R. A. Fisher in the 21st century." In: *Statistical Science* 13.2 (1998), pp. 95–122. DOI: 10.1214/ss/1028905930. URL: https://projecteuclid.org/download/pdf_1/euclid.ss/1028905930 (Cited on page 104).

[36] Tom Fawcett. "An introduction to ROC analysis." In: *Pattern Recognition Letters* 27.8 (June 2006), pp. 861–874. ISSN: 0167-8655. DOI: 10.1016/j.patrec.2005.10.010 (Cited on page 30).

[37] Ronald A. Fisher. "Conclusions fiduciaires." In: *Annales de l'institut Henri Poincaré* 10.3 (1948), pp. 191–213. URL: www.numdam.org/item/AIHP_1948__10_3_191_0/ (Cited on page 91).

[38] Ronald A. Fisher. "Student." In: *Annals of Eugenics* 9.1 (Jan. 1939), pp. 1–9. DOI: 10.1111/j.1469-1809.1939.tb02192.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1469-1809.1939.tb02192.x (Cited on page 91).

[39] Valentin Flunkert, David Salinas, and Jan Gasthaus. "DeepAR: probabilistic forecasting with autoregressive recurrent networks." In: *CoRR* abs/1704.04110 (Apr. 2017). arXiv: 1704.04110 [cs.AI] (Cited on page 22).

[40] Tilmann Gneiting and Matthias Katzfuss. "Probabilistic forecasting." In: *Annual Review of Statistics and Its Application* 1.1 (Jan. 2014), pp. 125–151. DOI: 10.1146/annurev-statistics-062713-085831 (Cited on pages 21–24, 59, 99, 100, 103, 130).

[41] Tilmann Gneiting and Adrian E. Raftery. "Strictly proper scoring rules, prediction, and estimation." In: *Journal of the American Statistical Association* 102.477 (Mar. 2007), pp. 359–378. ISSN: 01621459. DOI: 10.1198/016214506000001437 (Cited on page 117).

[42] Paul W. Goldberg, Christopher K. I. Williams, and Christopher M. Bishop. "Regression with Input-Dependent Noise: A Gaussian Process Treatment." In: *Proceedings of the 10th International Conference on Neural Information Processing Systems*. Ed. by M. I. Jordan, M. J. Kearns, and S. A. Solla. Vol. 10. MIT Press, Jan. 1998, pp. 493–499. URL: https://papers.nips.cc/paper/1997/file/afe434653a898da20044041262b3ac74-Paper.pdf (Cited on page 112).

[43] Yael Grushka-Cockayne and Victor Richmond R. Jose. "Combining prediction intervals in the M4 competition." In: *International Journal of Forecasting* 36.1 (Jan. 2020). Part of special issue: M4 Competition, pp. 178–185. ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2019.04.015 (Cited on pages 30, 31).

[44] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. "On calibration of modern neural networks." In: *Proceedings of the 34th International Conference on Machine Learning (PMLR)*. Vol. 70. Proceedings of Machine Learning Research. Sydney, NSW, Australia: PMLR, Aug. 2017, pp. 1321–1330. URL: http://proceedings.mlr.press/v70/guo17a.html (Cited on pages 30, 33, 38, 41, 42, 44, 45, 51, 52, 69, 70).

[45] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In: *CoRR* abs/1512.03385 (Dec. 2015). arXiv: 1512.03385 [cs.CV] (Cited on pages 30, 51, 69).

[46] H. V. Henderson and S. R. Searle. "On deriving the inverse of a sum of matrices." In: *SIAM Review* 23.1 (Jan. 1981), pp. 53–60. ISSN: 0036-1445. DOI: 10.1137/1023004 (Cited on page 108).

[47] Ralph Herbrich. *Learning Kernel Classifiers: Theory and Algorithms*. Cambridge, Mass: The MIT Press, Dec. 2001. ISBN: 0-262-08306-X (Cited on page 116).

[48] H. Hersbach. "Decomposition of the Continuous Ranked Probability Score for Ensemble Prediction Systems." In: *Weather and Forecasting* 15 (2000), pp. 559–570 (Cited on page 117).

[49] Bruce M. Hill. "De Finetti's theorem, induction, and $A_{(n)}$ or Bayesian nonparametric predictive inference (with discussion)." In: *Bayesian Statistics*. Ed. by Dennis V. Lindley, José M. Bernardo, Morris H. DeGroot, and Adrian F. M. Smith. Vol. 3. Oxford: Oxford University Press, 1988, pp. 211–241 (Cited on page 91).

[50] Bruce M. Hill. "Posterior distribution of percentiles: Bayes' theorem for sampling from a population." In: *Journal of the American Statistical Association* 63.322 (June 1968), pp. 677–691. ISSN: 01621459. DOI: 10.2307/2284038 (Cited on pages 91, 92).

[51] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. "Distilling the knowledge in a neural network." In: *CoRR* (Mar. 2015). arXiv: 1503.02531 [stat.ML] (Cited on page 41).

[52] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory." In: *Neural Computation* 9 (Dec. 1997), pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735. URL: https://www.researchgate.net/publication/13853244_Long_Short-term_Memory (Cited on page 21).

[53] Sergey Ioffe and Christian Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift." In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning*. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, July 2015, pp. 448–456. URL: http://proceedings.mlr.press/v37/ioffe15.pdf (Cited on page 41).

[54] E. T. Jaynes. "Information theory and statistical mechanics." In: *Physical Review* 106 (4 May 1957), pp. 620–630. DOI: 10.1103/physrev.106.620 (Cited on page 41).

[55] Harold Jeffreys. "On the theory of errors and least squares." In: *Proceedings of the Royal Society of London*. Series A, (Containing Papers of a Mathematical and Physical Character) 138.834 (Oct. 1932), pp. 48–55. ISSN: 09501207. DOI: 10.1098/rspa.1932.0170. URL: https://royalsocietypublishing.org/doi/pdf/10.1098/rspa.1932.0170 (Cited on page 91).

[56] Xiaoqian Jiang, Melanie Osl, Jihoon Kim, and Lucila Ohno-Machado. "Smooth isotonic regression: a new method to calibrate predictive models." In: *AMIA Joint Summits on Translational Science Proceedings*. Vol. 2011. American Medical Informatics Association. Mar. 2011, pp. 16–20 (Cited on pages 36, 47).

[57] Ulf Johansson and Patrick Gabrielsson. "Are Traditional Neural Networks Well-Calibrated?" In: 2019, pp. 1–8 (Cited on pages 30, 38, 50, 51, 70).

[58] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. "Lightgbm: A highly efficient gradient boosting decision tree." In: *Advances in neural information processing systems* 30 (2017), pp. 3146–3154 (Cited on page 59).

[59] Henry J. Kelley. "Gradient Theory of Optimal Flight Paths." In: *ARS Journal* 30.10 (Oct. 1960), pp. 947–954. DOI: 10.2514/8.5282. URL: https://doi.org/10.2514/8.5282 (Cited on page 21).

[60] Alex Kendall and Yarin Gal. "What uncertainties do we need in bayesian deep learning for computer vision?" In: *31st Conference on Neural Information Processing Systems (NIPS 2017)*. Long Beach, CA, USA: arXiv.org, Mar. 2017, pp. 5580–5590 (Cited on page 22).

[61] Frank Knight. *Risk, Uncertainty and Profit*. Boston, USA: Houghton Mifflin Company, 1921 (Cited on page 103).

[62] Alex Krizhevsky. *Learning multiple layers of features from tiny images*. Tech. rep. University of Toronto, Apr. 2009. URL: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf (Cited on page 38).

[63] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet classification with deep convolutional neural networks." In: *Proceedings of the 25th International Conference on Neural Information Processing Systems — Volume 1*. Advances in Neural Information Processing Systems'12 25. Lake Tahoe, Nevada: Curran Associates, Inc., Jan. 2012, pp. 1097–1105. DOI: 10.1145/3065386. URL: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf (Cited on pages 19, 20, 38).

[64] Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. "Accurate uncertainties for deep learning using calibrated regression." In: *Proceedings of the 35th International Conference on Machine Learning*. Vol. 80. Proceedings of Machine Learning Research. Stockholmsmässan, Stockholm Sweden: PMLR, July 2018, pp. 2796–2804. URL: http://proceedings.mlr.press/v80/kuleshov18a.html (Cited on page 38).

[65] Meelis Kull, Telmo Silva Filho, and Peter Flach. "Beta calibration: a well-founded and easily implemented improvement on logistic calibration for binary classifiers." In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. Vol. 54. Proceedings of Machine Learning Research. Fort Lauderdale, FL, USA: PMLR, Apr. 2017, pp. 623–631. URL: http://proceedings.mlr.press/v54/kull17a/kull17a.pdf (Cited on pages 35–38, 50, 51).

[66] Ananya Kumar, Percy Liang, and Tengyu Ma. "Verified uncertainty calibration." In: *CoRR* abs/1909.10155 (Sept. 2019). arXiv: 1909.10155 [cs.LG] (Cited on pages 38, 39).

[67] Aviral Kumar, Sunita Sarawagi, and Ujjwal Jain. "Trainable calibration measures for neural networks from kernel mean embeddings." In: *Proceedings of the 35th International Conference on Machine Learning*. Vol. 80. Proceedings of Machine Learning Research. Stockholmsmässan, Stockholm Sweden: PMLR, July 2018, pp. 2805–2814. URL: http://proceedings.mlr.press/v80/kumar18a.html (Cited on pages 42, 43).

[68] Antonis Lambrou, Harris Papadopoulos, Ilia Nouretdinov, and Alexander Gammerman. "Reliable probability estimates based on support vector machines for large multiclass datasets." In: *IFIP International Conference on Artificial Intelligence Applications and Innovations*. Vol. 382. Halkidiki, Greece: Springer Berlin Heidelberg, Sept. 2012, pp. 182–191. ISBN: 978-3-642-33411-5. DOI: 10.1007/978-3-642-33412-2_19. URL: https://hal.archives-ouvertes.fr/hal-01523046/document (Cited on page 47).

[69] Niels Landwehr, Mark Hall, and Eibe Frank. "Logistic model trees." In: *Machine Learning* 59.1–2 (May 2005), pp. 161–205. ISSN: 0885-6125. DOI: 10.1007/s10994-005-0466-3 (Cited on page 39).

[70] David A. Lane. "Fisher, Jeffreys, and the nature of probability." In: *R.A. Fisher: An Appreciation*. Ed. by Fienberg S. E. and D. V. Hinkley. New York: Springer New York, 1980, pp. 148–160. ISBN: 978-0-387-90476-4. DOI: 10.1007/978-1-4612-6079-0_15 (Cited on page 91).

[71] Nikolay Laptev, Jason Yosinski, Li Erran, and Slawek Smyl. "Time-series extreme event forecasting with neural networks at uber." In: *Time Series Workshop, (ICML 2017)*. 34. Sydney, Australia, Aug. 2017, pp. 1–5 (Cited on page 22).

[72] Steffen Lauritzen. *Extremal Families and Systems of Sufficient Statistics*. Lecture Notes in Statistics. New York: Springer-Verlag, Nov. 1988. ISBN: 0387968725 (Cited on page 25).

[73] Jerry F. Lawless and Marc Fredette. "Frequentist prediction intervals and predictive distributions." In: *Biometrika* 92.3 (Sept. 2005), pp. 529–542. ISSN: 0006-3444. DOI: 10.1093/biomet/92.3.529 (Cited on pages 72, 73).

[74] Quoc Le, Alex Smola, and Stéphane Canu. "Heteroscedastic gaussian process regression." In: *Proceedings of the 22nd International Conference on Machine Learning*. ICML '05. New York, NY, USA: Association for Computing Machinery, 2005, pp. 489–496. ISBN: 1595931805. DOI: 10.1145/1102351.1102413 (Cited on page 112).

[75] Tim Leathart, Eibe Frank, Geoffrey Holmes, and Bernhard Pfahringer. "Probability calibration trees." In: *CoRR* abs/1808.00111 (July 2018). arXiv: 1808.00111 [cs.LG] (Cited on page 39).

[76] Tim Leathart, Eibe Frank, Bernhard Pfahringer, and Geoffrey Holmes. "On calibration of nested dichotomies." In: *Advances in Knowledge Discovery and Data Mining*. Macau, China: Springer International Publishing, Apr. 2019, pp. 69–80. DOI: 10.1007/978-3-030-16148-4_6. URL: https://researchcommons.waikato.ac.nz/bitstream/handle/10289/12887/on-calibration-of-nested-dichotomies.pdf (Cited on pages 37, 47).

[77] Yann LeCun, Bernhard Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne Hubbard, and Lawrence D. Jackel. "Backpropagation applied to handwritten zip code recognition." In: *Neural Computation* 1.4 (Dec. 1989), pp. 541–551. ISSN: 0899-7667. DOI: 10.1162/neco.1989.1.4.541 (Cited on pages 30, 69).

[78] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. "Gradient-based learning applied to document recognition." In: *Proceedings of the IEEE*. Vol. 86. 11. Torino, Italy, Nov. 1998, pp. 2278–2324. DOI: 10.1109/5.726791 (Cited on pages 19, 20).

[79] Jing Lei and Larry Wasserman. "Distribution-free prediction bands for non-parametric regression." In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 76.1 (July 2013), pp. 71–96. DOI: https://doi.org/10.1111/rssb.12021. URL: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/rssb.12021 (Cited on page 86).

[80] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. "Focal loss for dense object detection." In: *CoRR* abs/1708.02002 (Aug. 2017). arXiv: 1708.02002 [cs.CV] (Cited on page 44).

[81] Henrik Linusson, Ulf Norinder, Henrik Boström, Ulf Johansson, and Tuve Löfström. "On the calibration of aggregated conformal predictors." In: *Proceedings of the Sixth Workshop on Conformal and Probabilistic Prediction and Applications*. Ed. by Alex Gammerman, Vladimir Vovk, Zhiyuan Luo, and Harris Papadopoulos. Vol. 60. Proceedings of Machine Learning Research. Stockholm, Sweden: PMLR, June 2017, pp. 154–173. URL: http://proceedings.mlr.press/v60/linusson17a.html (Cited on pages 129, 130, 138).

[82] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. "The M4 competition: 100, 000 time series and 61 forecasting methods." In: *International Journal of Forecasting* 36.1 (Jan. 2020), pp. 54–74. ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2019.04.014. URL: https://www.sciencedirect.com/science/article/pii/S0169207019301128 (Cited on pages 22, 31).

[83] Spyros Makridakis, Evangelos Spiliotis, and Vassilis Assimakopoulos. "The M5 Accuracy competition: Results, findings and conclusions." In: (Oct. 2020) (Cited on pages 22, 23).

[84] Valery Manokhin. *Awesome Conformal Prediction*. 2022. DOI: 10.5281/ZENODO.6467205. URL: https://zenodo.org/record/6467205 (Cited on page 27).

[85] Valery Manokhin. "Multi-class probabilistic classification using inductive and cross Venn-Abers predictors." In: *Proceedings of the Sixth Workshop on Conformal and Probabilistic Prediction and Applications*. Vol. 60. Proceedings of Machine Learning Research. Stockholm, Sweden: PMLR, June 2017, pp. 228–240. URL: http://proceedings.mlr.press/v60/manokhin17a/manokhin17a.pdf (Cited on page 26).

[86] Valery Manokhin. *valeman/Multi-class-probabilistic-classification: v0.1.0*. 2022. DOI: 10.5281/ZENODO.6685149. URL: https://zenodo.org/record/6685149 (Cited on page 26).

[87] Peter McCullagh, Vladimir Vovk, Ilia Nouretdinov, Dmitry Devetyarov, and Alex Gammerman. "Conditional prediction intervals for linear regression." In: *2009 International Conference on Machine Learning and Applications*. Miami, FL, USA: IEEE, Dec. 2009, pp. 131–138. DOI: 10.1109/icmla.2009.115 (Cited on page 104).

[88] Thomas Melluish, Craig Saunders, Ilia Nouretdinov, and Volodya Vovk. "Comparing the bayes and typicalness frameworks." In: *Machine Learning: ECML 2001*. Berlin, Heidelberg: Springer Berlin Heidelberg, Oct. 2001, pp. 360–371. ISBN: 978-3-540-42536-6. DOI: 10.1007/3-540-44795-4_31. URL: https://www.researchgate.net/publication/2393200_Comparing_the_Bayes_and_Typicalness_Frameworks/link/09e4150cee445659a0000000/download (Cited on pages 30, 31, 100).

[89] Soundouss Messoudi, Sylvain Rousseau, and Sébastien Destercke. "Deep conformal prediction for robust models." In: *Information Processing and Management of Uncertainty in Knowledge-Based Systems*. Ed. by Marie-Jeanne Lesot, Susana Vieira, Marek Z. Reformat, João Paulo Carvalho, Anna Wilbik, Bernadette Bouchon-Meunier, and Ronald R. Yager. Springer, Cham, June 2020, pp. 528–540. ISBN: 978-3-030-50146-4. DOI: https://doi.org/10.1007/978-3-030-50146-4_39 (Cited on pages 30, 31, 51, 69).

[90] Donald Michie, David Spiegelhalter, and Charles C. Taylor. *Machine Learning, Neural and Statistical Classification*. London: Overseas Press, 2009. ISBN: 8188689734 (Cited on page 59).

[91] Marvin Minsky and Seymour Papert. *Perceptrons. An Introduction to Computational Geometry*. Cambridge, Mass: MIT Press, Jan. 1969. ISBN: 9780262130431 (Cited on page 21).

[92] Tom Mitchell. *Machine Learning*. Singapore: McGraw-Hill Education, Mar. 1997. ISBN: 0070428077 (Cited on page 19).

[93] Mohammed Mohammadi. "On the bounds for diagonal and off-diagonal elements of the hat matrix in the linear regression model." In: *REVSTAT — Statistical*

Journal 14.1 (Feb. 2016), pp. 75–87. URL: https://www.ine.pt/revstat/pdf/rs160104.pdf (Cited on page 83).

[94] Jishnu Mukhoti, Viveka Kulharia, Amartya Sanyal, Stuart Golodetz, Philip H. S. Torr, and Puneet K. Dokania. "Calibrating deep neural networks using focal loss." In: (Feb. 2020). arXiv: 2002.09437 [cs.LG] (Cited on pages 29–31, 38, 44, 45, 51, 69, 70).

[95] Rafael Müller, Simon Kornblith, and Geoffrey E. Hinton. "When Does Label Smoothing Help?" In: *CoRR* abs/1906.02629 (June 2019). arXiv: 1906.02629 [cs.LG] (Cited on page 43).

[96] Allan H. Murphy. "The early history of probability forecasts: some extensions and clarifications." In: *Weather and Forecasting* 13.1 (Mar. 1998), pp. 5–15. ISSN: 0882-8156. DOI: 10.1175/1520-0434(1998)013<0005:TEHOPF>2.0.CO;2. URL: https://journals.ametsoc.org/view/journals/wefo/13/1/1520-0434_1998_013_0005_tehopf_2_0_co_2.xml (Cited on page 21).

[97] Kevin P. Murphy. *Probabilistic Machine Learning: An introduction*. MIT Press, 2022. URL: probml.ai (Cited on page 27).

[98] Mahdi Pakdaman Naeini and Gregory F. Cooper. "Binary classifier calibration using an ensemble of near isotonic regression models." In: *2016 IEEE 16th International Conference on Data Mining (ICDM)*. Vol. 2016. Los Alamitos, CA, USA: IEEE Computer Society, Dec. 2016, pp. 360–369. DOI: 10.1109/ICDM.2016.0047. URL: https://doi.ieeecomputersociety.org/10.1109/ICDM.2016.0047 (Cited on pages 40, 47).

[99] Mahdi Pakdaman Naeini, Gregory F. Cooper, and Milos Hauskrecht. "Obtaining Accurate Probabilities Using Classifier Calibration." In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. Association for the Advancement of Artificial Intelligence (AAAI), 2015, pp. 2901–2907 (Cited on pages 33, 40).

[100] Mahdi Pakdaman Naeini, Gregory F. Cooper, and Milos Hauskrecht. "Obtaining well calibrated probabilities using bayesian binning." In: *Proceedings of the Twenty-ninth AAAI Conference on Artificial Intelligence*. AAAI'15. Austin, Texas:

AAAI Press, Apr. 2015, pp. 2901–2907. ISBN: 0262511290. URL: https://ojs.aaai.org/index.php/AAAI/article/view/9602/9461 (Cited on pages 39–41).

[101] Ilia Nouretdinov, Denis Volkhonskiy, Pitt Lim, Paolo Toccaceli, and Alexander Gammerman. "Inductive Venn-Abers predictive distribution." In: *Proceedings of the Seventh Workshop on Conformal and Probabilistic Prediction and Applications*. Ed. by Alex Gammerman, Vladimir Vovk, Zhiyuan Luo, Evgueni Smirnov, and Ralf Peeters. Vol. 91. Proceedings of Machine Learning Research. PMLR, June 2018, pp. 15–36. URL: http://proceedings.mlr.press/v91/nouretdinov18a/nouretdinov18a.pdf (Cited on pages 27, 122).

[102] Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua V. Dillon, Balaji Lakshminarayanan, and Jasper Snoek. "Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift." In: *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*. Vancouver, Canada, Dec. 2019. URL: https://papers.nips.cc/paper/2019/file/8558cb408c1d76621371888657d2eb1d-Paper.pdf (Cited on pages 41, 51).

[103] Harris Papadopoulos. "Reliable probabilistic classification with neural networks." In: *Neurocomputing* 107 (May 2013). Part of special issue: Timely Neural Networks Applications in Engineering: Selected Papers from the 12th EANN International Conference, 2011, pp. 59–68. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2012.07.034 (Cited on pages 32, 46).

[104] Harris Papadopoulos. "Reliable probabilistic prediction for medical decision support." In: *IFIP Advances in Information and Communication Technology*. Ed. by Lazaros Iliadis, Ilias Maglogiannis, and Harris Papadopoulos. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 265–274. ISBN: 978-3-642-23960-1. DOI: 10.1007/978-3-642-23960-1_32 (Cited on page 32).

[105] Telma Pereira, Sandra Cardoso, Manuela Guerreiro, Alexandre Mendonça, and Sara C. Madeira. "Targeting the uncertainty of predictions at patient-level using an ensemble of classifiers coupled with calibration methods, Venn-ABERS, and conformal predictors: a case study in AD." In: *Journal of Biomedical Informatics* 101

(Jan. 2020), p. 103350. ISSN: 1532-0464. DOI: 10.1016/j.jbi.2019.103350 (Cited on pages 22, 23, 29, 30, 32, 35, 50–52, 69, 70).

[106] Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey E. Hinton. "Regularizing neural networks by penalizing confident output distributions." In: *CoRR* abs/1701.06548 (Jan. 2017). arXiv: 1701.06548 [cs.NE] (Cited on pages 41–43, 45).

[107] Donald A. Pierce and Kenneth J. Kopecky. "Testing goodness of fit for the distribution of errors in regression models." In: *Biometrika* 66.1 (Apr. 1979), pp. 1–5. ISSN: 0006-3444. DOI: 10.1093/biomet/66.1.1 (Cited on page 92).

[108] Iosif Pinelis. "Exact bounds on the closeness between the student and standard normal distributions." In: *ESAIM: Probability and Statistics* 19 (Mar. 2015), pp. 24–27. DOI: 10.1051/ps/2014014. URL: https://www.esaim-ps.org/articles/ps/pdf/2015/01/ps140014.pdf (Cited on page 88).

[109] John C. Platt. "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods." In: *Advances in Large Margin Classifiers*. Ed. by Alexander J. Smola and Peter Bartlett. Vol. 10. 3. MIT Press, Mar. 1999, pp. 61–74 (Cited on pages 33–35, 38, 50, 59).

[110] T. Poggio. "On optimal nonlinear associative recall." In: *Biological Cybernetics* 19.4 (Sept. 1975), pp. 201–209. DOI: 10.1007/bf02281970 (Cited on page 116).

[111] David Price, Stefan Knerr, Léon Personnaz, and Gérard Dreyfus. "Pairwise neural network classifiers with probabilistic outputs." In: *Advances in Neural Information Processing Systems*. Ed. by G. Tesauro, D. Touretzky, and T. Leen. Vol. 7. MIT Press, 1994, pp. 1109–1116. URL: https://proceedings.neurips.cc/paper/1994/file/210f760a89db30aa72ca258a3483cc7f-Paper.pdf (Cited on page 54).

[112] Yaniv Romano, Evan Patterson, and Emmanuel J. Candès. "Conformalized Quantile Regression." In: (2019). arXiv: 1905.03222 [stat.ME] (Cited on page 22).

[113] Frank Rosenblatt. "The perceptron: a probabilistic model for information storage and organization in the brain." In: *Psychological Review* 65.6 (1958), pp. 386–408. DOI: 10.1037/h0042519 (Cited on page 21).

[114] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. "Learning representations by back-propagating errors." In: *Nature* 323.6088 (Oct. 1986), pp. 533–536. ISSN: 1476-4687. DOI: 10.1038/323533a0 (Cited on pages 19, 21, 43).

[115] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. 3rd. Upper Saddle River, New Jersey: Pearson, 2013. ISBN: 0136042597 (Cited on page 21).

[116] Jarkko Salojärvi, Kai Puolamäki, Jaana Simola, Lea Kovanen, Ilpo Kojo, and Samuel Kaski. "Inferring Relevance from Eye Movements: Feature Extraction." In: 2005 (Cited on page 66).

[117] Tore Schweder and Nils Lid Hjort. *Confidence, Likelihood, Probability: Statistical Inference with Confidence Distributions*. New York, NY: Cambridge University Press, Feb. 2016. ISBN: 9780521861601 (Cited on pages 71, 95, 104).

[118] George A. F. Seber and Alan J. Lee. *Linear Regression Analysis*. Wiley Series in Probability and Statistics. Wiley, Jan. 2003. ISBN: 9780471415404. DOI: 10.1002/9780471722199 (Cited on page 88).

[119] Teddy Seidenfeld. "Jeffreys, Fisher, and Keynes: Predicting the Third Observation, Given the First Two." In: *History of Political Economy* 27.5 (Jan. 1995), pp. 39–52. DOI: 10.1215/00182702-27-supplement-39 (Cited on page 91).

[120] Glenn Shafer and Vladimir Vovk. "A tutorial on conformal prediction." In: *CoRR* abs/0706.3188 (June 2007) (Cited on page 25).

[121] Jieli Shen, Regina Y. Liu, and Min-ge Xie. "Prediction with Confidence — a General Framework for Predictive Inference." In: *Journal of Statistical Planning and Inference* 195 (May 2018), pp. 126–140. ISSN: 0378-3758. DOI: 10.1016/j.jspi.2017.09.012 (Cited on pages 71, 73, 95, 104).

[122] Albert N. Shiryaev. *Вероятность* (*Probability*). Vol. 95. Graduate Texts in Mathematics. New York: Springer-Verlag, 1989. DOI: 10.1007/978-1-4757-2539-1 (Cited on page 102).

[123] Edward Snelson and Zoubin Ghahramani. "Variable noise and dimensionality reduction for sparse gaussian processes." In: *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence*. Ed. by R. Dechter and T. S. Richardson.

UAI'06. Arlington, Virginia, USA: AUAI Press, 2006, pp. 461–468. ISBN: 0974903922 (Cited on page 112).

[124] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. "Dropout: a simple way to prevent neural networks from overfitting." In: *Journal of Machine Learning Research* 15.1 (Jan. 2014), pp. 1929–1958. ISSN: 1532-4435. URL: https://dl.acm.org/doi/pdf/10.5555/2627435.2670313 (Cited on page 41).

[125] Ingo Steinwart. "On the influence of the kernel on the consistency of support vector machines." In: *Journal of Machine Learning Research* 2 (Dec. 2001), pp. 67–93. ISSN: 1532-4435. DOI: 10.1162/153244302760185252. URL: https://www.jmlr.org/papers/volume2/steinwart01a/steinwart01a.pdf (Cited on page 113).

[126] Ingo Steinwart, Don Hush, and Clint Scovel. "An explicit description of the reproducing kernel Hilbert spaces of Gaussian RBF kernels." In: *IEEE Transactions on Information Theory* 52.10 (Oct. 2006), pp. 4635–4643. DOI: 10.1109/TIT.2006.881713 (Cited on page 116).

[127] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. "Rethinking the inception architecture for computer vision." In: *CoRR* abs/1512.00567 (Dec. 2015). arXiv: 1512.00567 [cs.CV] (Cited on pages 42, 43).

[128] Christine Thomas-Agnan. "Computing a family of reproducing kernels for statistical applications." In: *Numerical Algorithms* 13.1 (Mar. 1996), pp. 21–32. DOI: 10.1007/bf02143124 (Cited on pages 113, 114).

[129] Paolo Toccaceli, Ilia Nouretdinov, and Alexander Gammerman. "Conformal predictors for compound activity prediction." In: *Conformal and Probabilistic Prediction with Applications (COPA 2016)*. Vol. 9653. Lecture Notes in Computer Science (LNCS). Springer International Publishing, Apr. 2016, pp. 51–66. ISBN: 978-3-319-33394-6. DOI: 10.1007/978-3-319-33395-3_4 (Cited on page 22).

[130] Alan M Turing. "Computing machinery and intelligence." In: *Mind* LIX.236 (Oct. 1950), pp. 433–460. ISSN: 0026-4423. DOI: 10.1093/mind/LIX.236.433. URL: https:

//academic.oup.com/mind/article-pdf/LIX/236/433/30123314/lix-236-433.pdf (Cited on page 21).

[131] Stanford University. *Honesty in statistical models*. https://news.stanford.edu/2021/03/19/honesty-statistical-models/. Online; Accessed 20/4/2021. Mar. 2021 (Cited on pages 22, 49).

[132] Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. "OpenML: networked science in machine learning." In: *SIGKDD Explorations* 15.2 (2013), pp. 49–60 (Cited on pages 59, 60, 67).

[133] V. Vapnik. "Pattern recognition using generalized portrait method." In: *Automation and Remote Control* 24 (1963), pp. 774–780 (Cited on page 21).

[134] Vladimir N. Vapnik. *Statistical learning theory*. Vol. 1. New York: Wiley, Sept. 1998. ISBN: 978-0-471-03003-4 (Cited on page 35).

[135] Vladimir Vovk. "Cross-conformal predictors." In: *Annals of Mathematics and Artificial Intelligence* 74 (July 2013), pp. 9–28. DOI: 10.1007/s10472-013-9368-4 (Cited on pages 129, 138).

[136] Vladimir Vovk. "Universally consistent predictive distributions." In: *CoRR* abs/1708.01902 (Aug. 2017). arXiv: 1708.01902 [cs.LG] (Cited on pages 104, 112, 130).

[137] Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Berlin: Springer-Verlag, Mar. 2005. ISBN: 0387001522. DOI: 10.1007/b106715 (Cited on pages 25, 32, 46, 55, 77, 80, 86, 103, 108, 112, 116, 117, 122, 123).

[138] Vladimir Vovk, Ilia Nouretdinov, and Alex Gammerman. "On-line predictive linear regression." In: *The Annals of Statistics* 37.3 (June 2009), pp. 1566–1590. DOI: 10.1214/08-AOS622 (Cited on page 101).

[139] Vladimir Vovk, Ilia Nouretdinov, Valery Manokhin, and Alex Gammerman. "Conformal predictive distributions with kernels." In: *Braverman Readings in Machine Learning. Key Ideas from Inception to Current State*. Springer International Publishing, Apr. 2018, pp. 103–121. ISBN: 978-3-319-99491-8. DOI: 10.1007/978-3-319-

99492-5_4. URL: https://www.researchgate.net/publication/327171113_Conformal_Predictive_Distributions_with_Kernels_International_Conference_Commemorating_the_40th_Anniversary_of_Emmanuil_Braverman's_Decease_Boston_MA_USA_April_28-30_2017_Invited_Talks (Cited on pages 27, 100, 103, 104, 106, 115, 118, 122).

[140] Vladimir Vovk, Ilia Nouretdinov, Valery Manokhin, and Alexander Gammerman. "Cross-conformal predictive distributions." In: *Proceedings of the Seventh Workshop on Conformal and Probabilistic Prediction and Applications*. Vol. 91. Proceedings of Machine Learning Research. PMLR, June 2018, pp. 37–51. URL: http://proceedings.mlr.press/v91/vovk18a/vovk18a.pdf (Cited on pages 27, 130, 135, 136, 138, 140).

[141] Vladimir Vovk, Ivan Petej, and Valentina Fedorova. "Large-scale probabilistic predictors with and without guarantees of validity." In: *Advances in Neural Information Processing Systems—NIPS 2015*. Vol. 28. Curran Associates, Inc., Nov. 2015, pp. 892–900. URL: https://proceedings.neurips.cc/paper/2015/file/a9a1d5317a33ae8cef33961c34144f84-Paper.pdf (Cited on pages 52, 55–59, 70, 134).

[142] Vladimir Vovk, Ivan Petej, Ilia Nouretdinov, Valery Manokhin, and Alex J. Gammerman. "Computationally efficient versions of conformal predictive distributions." In: *CoRR* abs/1911.00941 (Nov. 2019). arXiv: 1911.00941 [cs.LG] (Cited on page 27).

[143] Vladimir Vovk, Ivan Petej, Ilia Nouretdinov, Valery Manokhin, and Alexander Gammerman. "Computationally efficient versions of conformal predictive distributions." In: *Neurocomputing* 397 (July 2020), pp. 292–308. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2019.10.110 (Cited on pages 27, 77, 133).

[144] Vladimir Vovk, Glenn Shafer, and Ilia Nouretdinov. "Self-calibrating probability forecasting." In: *Proceedings of the 16th International Conference on Neural Information Processing Systems*. Vol. 16. NIPS'03. Whistler, British Columbia, Canada: MIT Press, Dec. 2003, pp. 1133–1140. URL: https://proceedings.neurips.cc/

paper/2003/file/10c66082c124f8afe3df4886f5e516e0-Paper.pdf (Cited on pages 46, 55).

[145] Vladimir Vovk, Jieli Shen, Valery Manokhin, and Min-ge Xie. "Nonparametric predictive distributions based on conformal prediction." In: *Proceedings of the Sixth Workshop on Conformal and Probabilistic Prediction and Applications*. Ed. by Alex Gammerman, Vladimir Vovk, Zhiyuan Luo, and Harris Papadopoulos. Vol. 60. Proceedings of Machine Learning Research. Stockholm, Sweden: PMLR, June 2017, pp. 82–102. DOI: 10.1007/s10994-018-5755-8. URL: http://proceedings.mlr.press/v60/vovk17a/vovk17a.pdf (Cited on pages 26, 100, 104, 114, 118).

[146] Vladimir Vovk, Jieli Shen, Valery Manokhin, and Min-ge Xie. "Nonparametric predictive distributions based on conformal prediction." In: *Machine Learning* 108.3 (Aug. 2018), pp. 445–474. DOI: 10.1007/s10994-018-5755-8 (Cited on pages 26, 75, 86, 89, 95).

[147] Vladimir Vovk and Ruodu Wang. "Combining p-values via averaging." In: (Dec. 2012). arXiv: 1212.4966 [math.ST] (Cited on page 130).

[148] C. M. Wang, Jan Hannig, and Hari K. Iyer. "Fiducial prediction intervals." In: *Journal of Statistical Planning and Inference* 142.7 (July 2012), pp. 1980–1990. DOI: 10.1016/j.jspi.2012.02.021. URL: https://www.researchgate.net/publication/257199342_Fiducial_prediction_interval (Cited on page 88).

[149] Larry Wasserman. "Frasian inference." In: *Statistical Science* 26.3 (Oct. 2011), pp. 322–325. ISSN: 0883-4237. DOI: 10.1214/11-sts352c. URL: https://projecteuclid.org/euclid.ss/1320066921 (Cited on page 112).

[150] Jason Weston and Chris Watkins. "Support vector machines for multi-class pattern recognition." In: *European Symposium on Artificial Neural Network — ESANN'1999 Proceedings*. Vol. 99. Bruges, Belgium, Apr. 1999, pp. 219–224. URL: https://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es1999-461.pdf (Cited on page 53).

[151] Ronald J. Williams and Jing Peng. "Function optimization using connectionist reinforcement learning algorithms." In: *Connection Science* 3.3 (Sept. 1991), pp. 241–

268. DOI: 10.1080/09540099108946587. URL: https://www.researchgate.net/publication/2703232_Function_Optimization_Using_Connectionist_Reinforcement_Learning_Algorithms (Cited on page 42).

[152] Bianca Zadrozny and Charles Elkan. "Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers." In: *Proceedings of the Eighteenth International Conference on Machine Learning*. Vol. 1. ICML '01. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., May 2001, pp. 609–616. ISBN: 1558607781. URL: https://cseweb.ucsd.edu/~elkan/calibrated.pdf (Cited on pages 33, 38, 51, 55).

[153] Bianca Zadrozny and Charles Elkan. "Transforming classifier scores into accurate multiclass probability estimates." In: *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '02. Edmonton, Alberta, Canada: Association for Computing Machinery, Aug. 2002, pp. 694–699. ISBN: 158113567X. DOI: 10.1145/775047.775151. URL: https://www.researchgate.net/publication/2571315_Transforming_Classifier_Scores_into_Accurate_Multiclass_Probability_Estimates (Cited on pages 35, 36, 47, 55, 59).

[154] Jian Zhang and Yiming Yang. "Probabilistic score estimation with piecewise logistic regression." In: *Proceedings of the Twenty-first International Conference on Machine Learning —(ICML 2004)*. Ed. by Carla E. Brodley. Vol. 69. ICML '04. Banff, Alberta, Canada: Association for Computing Machinery (ACM), July 2004, pp. 115–123. ISBN: 1581138385. DOI: 10.1145/1015330.1015335 (Cited on page 35).

[155] Chenzhe Zhou, Ilia Nouretdinov, Zhiyuan Luo, Dmitry Adamskiy, Luke Randell, Nick Coldham, and Alex Gammerman. "A comparison of Venn Machine with Platt's method in probabilistic outputs." In: *12th Engineering Applications of Neural Networks (EANN 2011) and 7th Artificial Intelligence Applications and Innovations (AIAI)*. Vol. 364. Springer. Berlin, Heidelberg, Sept. 2011, pp. 483–490. ISBN: 978-3-642-23960-1. DOI: 10.1007/978-3-642-23960-1_56. URL: https://www.researchgate.net/publication/220827947_A_Comparison_of_

Venn_Machine_with_Platt's_Method_in_Probabilistic_Outputs (Cited on page 46).