Competitive online algorithms for probabilistic prediction

Submitted by

Raisa Dzhamtyrova

for the degree of Doctor of Philosophy

of the

Royal Holloway, University of London

2020

Declaration

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.

Raisa Dzhamtyrova

Supervisor and Examiners

Supervisor: *Dr Yuri Kalnishkan*
Internal Examiner: *Dr Vadim Shcherbakov*
External Examiner: *Dr Varun Kanade*

Abstract

The goal of this thesis is to develop new competitive online algorithms for making predictions. In online learning, as in real life, signals arrive sequentially, and decisions should be adjusted in regard to new information. In the competitive prediction framework, predictions are made by experts and a learner. The goal of the learner is to make predictions which are not too much worse than the best expert. The quality of the prediction is measured by the cumulative loss.

Our starting point is the Aggregating Algorithm (AA) which optimally merges experts' predictions and provides a theoretical guarantee on its performance in comparison with the best expert. An important feature of the AA is that it works under any circumstances and it does not require any statistical assumptions. For a lot of loss functions, the strategy of the AA predicts as well as the best expert. In other cases, it is possible to apply the Weak Aggregating Algorithm (WAA), which provides a weaker theoretical guarantee but still resolves the problem of competitive prediction.

In this thesis, we provide algorithms with strong theoretical guarantees for many loss functions and different sets of experts. We start with the generalisation of the AA with finitely many experts for prediction of vector-valued outcomes. We propose a number of merging algorithms for prediction of vector-valued outcomes with tight worst-case loss upper bounds similar to those for the AA. Another approach of applying the framework of prediction with expert advice with finitely many models is proposed for estimation of Value at Risk.

It is possible to achieve good theoretical guarantees even if the set of experts is infinite. In the second part of the dissertation, we provide competitive algorithms for making probabilistic predictions. The first proposed algorithm, which combines the class of quantile regression experts, can be used for making interval predictions. The second approach merges the class of linear regressions and output forecasts in the form of the cumulative distribution functions.

In this dissertation, an important problem of probabilistic multi-class classification is considered. The approach which combines the class of multinomial logistic regressions is proposed to solve this problem. The strategy is competitive in terms of the Kullback-Leibler cumulative loss.

The theoretical guarantees of the proposed algorithms are proven and experiments on artificial datasets and real-world datasets are conducted. Empirical results emphasise the properties of new algorithms and demonstrate that they have a good performance in practice.

Acknowledgements

I am grateful to my supervisor Yuri Kalnishkan for introducing me to the area of prediction with expert advice and providing help and guidance during my PhD. This research showed me new ways to approach and solve scientific problems. It has been an invaluable and exciting experience.

I thank my grandmother, parents and sister for their love and constant belief in me. Also I thank Marco Gorelli for his support and encouragement, it helped me a lot to stay positive and continue my work.

This research was made possible by the Reid Scholarship I was awarded from the Computer Science Department at Royal Holloway, for which I am very grateful. I thank all the staff of the Computer Science Department for providing a very friendly research environment and their help. I am grateful to the college for funding my trips abroad to present papers at conferences.

Contents

Notations

$\mathbb{R}$	the space of real numbers
$\mathbb{R}^n$	the n-dimentional space of real numbers
X'	transpose of matrix X
$\|x\|_1$	L_1-norm of $x \in \mathbb{R}^n$
$\|x\|_2$	the Eucledian norm (L_2-norm) of $x \in \mathbb{R}^n$
$\|x\|_\infty$	the maximum norm of $x \in \mathbb{R}^n$
$\mathcal{S}$	strategy of the learner
$\mathcal{S}_\theta$	strategy of the expert θ
L_T	loss of the learner at time T
L_T^θ	loss of the expert θ at time T

Acronyms

AA	Aggregating Algorithm
AAD	Aggregating Algorithm with Discounting
AAP	Aggregating Algorithm for Prediction of Packs
AAR	Aggregating Algorithm for Regression
APA	Aggregating Pseudo-Algorithm
CRPS	Continuous Ranked Probability Score
EWMA	Exponential Weighted Moving Average
GARCH	Generalised Autoregressive Conditional Heteroskedasticity
GBDT	Gradient Boosting Decision Trees
MCMC	Markov Chain Monte Carlo
MLR	Multinomial Logistic Regression
QR	Quantile Regression
QRF	Quantile Random Forests
RF	Random Forests
RKHS	Reproducing Kernel Hilbert Space
VaR	Value at Risk
WAA	Weak Aggregating Algorithm
WAAQR	Weak Aggregating Algorithm for Quantile Regression

Chapter 1

Introduction

> An expert is a man who has made all the mistakes which can be made, in a narrow field
>
> *Niels Bohr*

In everyday life people make their decisions based on predictions. For example, we make plans for a weekend based on a weather forecast or buy stocks of some company based on financial experts' predictions of their prices. Usually predictions are made by an algorithm which learns from past experience. The goal of this thesis is to develop online competitive algorithms for making predictions with theoretical bounds on their performance. These bounds should hold for any future time point and without any stochastic assumptions.

In the real world we usually receive new information sequentially. Hence, we study the online setting in which predictions and outcomes are given step-by-step. Contrary to batch mode, where the algorithm is trained on a training set and gives predictions on a test set, we learn as soon as new observations become available. For example, suppose that our aim is to predict outcomes of football matches of a new season based on data available from a previous season. A batch algorithm builds a model on previous season's data and this model is used to make predictions for all matches of the current season. In the online setting we add data sequentially and adjust parameters of the model after each match. More formally, we consider an online protocol where at each trial a learner observes a signal given by nature and attempts to predict an outcome, which is shown to the learner later. The leaner's performance is measured by means of the cumulative loss.

In this thesis, we consider the framework of online prediction with expert advice. At each trial we have access to predictions of experts and need to make a prediction

based on their past performance. In statistical learning, usually some assumptions are made about the mechanism that generates the data, and guarantees are given for the method working under these assumptions. For example, one may assume a linear dependence between electricity consumption and temperature and try to fit the best parameters for linear regression. Instead, we work in the adversarial setting where no assumptions are made about the data generating process.

In this thesis, we consider an approach called competitive prediction, where one provides guarantees compared to other predictive models that are called experts. Experts could be human experts, complex machine learning algorithms or even classes of functions. We treat an expert as a black box, i.e., we are not interested in its internal work. Our goal is to develop a merging strategy that will perform not much worse than the best expert. As a result, we do not try to build a model that works under certain assumptions but try to combine predictions that are given to us by experts. One may wonder why we do not just use predictions of only one best expert from the beginning and ignore predictions of others. First, sometimes we cannot have enough data to identify the best expert from the start. Second, good performance in the past does not necessary lead to good performance in the future.

The learner in the game of prediction plays against nature and experts from some pool. The aim of the learner is to keep its total loss small compared to the total losses of any expert. One of the approaches which provides a theoretical guarantee on the learner's performance is the Aggregating Algorithm (AA), which was first introduced in Vovk (1990). AA works as follows: we assign initial weights to experts and at each step the weights of experts are updated according to their current performance. The approach is similar to the Bayesian method, where the prediction is the average over all models based on the likelihood of the available data and model's prior distribution. For the case of finitely many experts, AA gives a guarantee ensuring that the learner's loss is as small as the best expert's loss plus a constant.

Even if the decision pool is infinite it is possible to achieve good theoretical guarantees. The history of algorithms competitive with large parametric classes of strategies can be traced back to universal portfolios by Cover and Ordentlich (1996) (a survey is provided by Cesa-Bianchi and Lugosi (2006)), which apply in the context of investment decisions and compete against portfolio selection techniques. In that framework one is interested in maximising wealth, but the problem can be restated in terms of losses.

One can consider outcomes and predictions from the one-dimensional interval $[0, 1]$ and signals $x_t \in \mathbb{R}^n$. A natural choice of competitor strategies are then linear functions on x_t. Vovk (2001) and Azoury and Warmuth (2001) propose an algorithm for this framework (Vovk-Azoury-Warmuth predictor, also known as the Aggregating Algo-

rithm for Regression) targeted at square loss. The resulting algorithm asymptotically performs as good as any linear regression in terms of the cumulative square loss up to a logarithmic term in the number of steps. Gammerman et al. (2004) obtain a kernelized version of the predictor. Zhdanov and Kalnishkan (2013) study similar competitive bounds for standard ridge regression. In this thesis, we primarily consider the problem of combining different classes of experts. These algorithms are called *universal* algorithms because they are competitive with any expert from the chosen class.

1.1 Original contributions

1. The generalisation of the Aggregating Algorithm is developed for prediction of vector-valued outcomes called *packs*. Three algorithms for this new setting are proposed (Algorithm 6, Algorithm 7 and Algorithm 8) and guarantees on their performances are proven (Theorem 3.4.1 and Theorem 3.4.2). Experimental results on sport and house price datasets are performed to compare algorithms.

2. A new approach to applying the method of prediction with expert advice is proposed for calculating Value at Risk. Experiments and comparisons with existing methods are performed with stocks of three companies.

3. Universal algorithms for probabilistic predictions are developed and performance guarantees are proven for the following settings: quantile regression under pinball loss function (Section 5.4 and Theorem 5.3.1) and linear regression under Continuous Ranked Probability Score (Section 6.4 and Theorem 6.3.1). Practical experiments show the performance of algorithms for renewable energy forecasting.

4. Universal algorithm for probabilistic multi-class classification under Kullback-Leibler game (Section 7.4 and Theorem 7.3.1) and its generalisation for Hilbert spaces (Section 7.6 and Theorem 7.6.4) are developed and their performance guarantees are proven for the class of multinomial logistic regressions.

1.2 Publications

The results described in this thesis are published either as conference proceedings or journal papers. The paper 'Competitive Online Regression under Continuous Ranked Probability Score' received the best student paper award.

1. Raisa Dzhamtyrova and Yuri Kalnishkan. Competitive Online Generalised Linear Regression with Multidimensional Outputs. In *Proceedings of the International Joint Conference on Neural Networks*, IEEE, 2019, pages 1–8.

2. Raisa Dzhamtyrova and Yuri Kalnishkan. Competitive Online Regression under Continuous Ranked Probability Score. In *Proceedings of Machine Learning Research, the 8th Symposium on Conformal and Probabilistic Prediction with Applications*, 2019, pages 105: 178–195

3. Raisa Dzhamtyrova and Yuri Kalnishkan. Universal algorithms for multinomial logistic regression under Kullback-Leibler game. *Neurocomputing*, 2019.

4. Dmitry Adamskiy, Tony Bellotti, Raisa Dzhamtyrova, and Yuri Kalnishkan. Aggregating algorithm for prediction of packs. *Machine Learning*, 2019, pages 108: 1231–1260.

1.3 Structure of the thesis

Chapter 2 gives an overview of prediction with expert advice and the main algorithms which are used to derive the new proposed algorithms. Chapter 3 is devoted to the problem of generalisation of the Aggregating Algorithms for prediction of packs. Chapter 4 describes a new approach to applying prediction with expert advice to calculate Value at Risk. In Chapter 5 the new universal algorithm for quantile regression under pinball loss is developed. Chapter 6 describes the competitive strategy for online linear regression under Continuous Ranked Probability Score. Chapter 7 gives new ways of approaching the problem of probabilistic multi-class classification.

Chapter 2

Main algorithms

In this chapter, we describe the main algorithms that are used in this work. This is the only chapter in the thesis where materials are not original, except for the proof of Lemma 2.4.2.

2.1 Framework

In the framework of prediction with expert advice we need to specify a *game* $\mathcal{G}$ which contains three components: a space of outcomes Ω, a decision space Γ, and a loss function $\lambda : \Omega \times \Gamma \to \mathbb{R}$. The *learner* or *prediction strategy* in the game of prediction plays against *experts* θ from some pool Θ and *nature*. At each step t experts output their predictions $\xi_t(\theta) \in \Gamma$. After seeing all experts' predictions, the learner outputs prediction $\gamma_t \in \Gamma$. After nature announces an outcome $y_t \in \Omega$, the experts and the learner suffer losses $\lambda(y_t, \xi_t(\theta))$ and $\lambda(y_t, \gamma_t)$ respectively. The aim of the learner is to keep its total loss L_t small compared to the total losses L_t^θ of all experts $\theta \in \Theta$. Sometimes we refer to the prediction strategy of the learner as $\mathcal{S}$ and the prediction strategy of the expert θ as $\mathcal{S}_\theta$. We define *regret* to be the difference between the cumulative losses of the best expert and the learner $R_t = L_t - \min_{\theta\in\Theta} L_t^\theta$.

Experts could be human experts, complex machine learning algorithms or even classes of functions. In Chapters 3 and 4, we consider a finite number of experts and treat an expert as a black box, i.e., we are not interested in its internal work. In the rest of the thesis we consider parametrised classes of experts, such as the class of quantile regressions (Chapter 5), the class of linear regressions (Chapter 6) and the class of multinomial logistic regressions (Chapter 7).

Below is the protocol of competitive online prediction:

Protocol 1.

$L_0 := 0$
$L_0^\theta := 0$
for $t = 1, 2, \ldots$
    Experts output $\xi_t(\theta) \in \Gamma,\ \theta \in \Theta$
    Learner outputs $\gamma_t \in \Gamma$
    Nature announces $y_t \in \Omega$
    $L_t := L_{t-1} + \lambda(y_t, \gamma_t)$
    $L_t^\theta := L_{t-1}^\theta + \lambda(y_t, \xi_t(\theta)),\ \theta \in \Theta$
end for

2.2 Aggregating Algorithm

2.2.1 Description of Aggregating Algorithm

One of the algorithms that can be used to solve the problem of prediction with expert advice is the Aggregating Algorithm (AA). We formulate the Aggregating Algorithm following Vovk (1990, 1998, 2001). The algorithm is based on the notion of *mixability*.

Consider a game $\mathfrak{G} = \langle \Omega, \Gamma, \lambda \rangle$. A constant $C > 0$ is *admissible* for a *learning rate* $\eta > 0$ if for every $\theta \in \Theta$, every set of predictions $\xi(\theta) \in \Gamma$, and every distribution $P(d\theta)$ (such that $\int_{\theta \in \Theta} P(d\theta) = 1$) there is $\gamma \in \Gamma$ ensuring for all outcomes $y \in \Omega$ the inequality

$$\lambda(y, \gamma) \le -\frac{C}{\eta} \ln \int_{\theta \in \Theta} e^{-\eta \lambda(y, \xi(\theta))} P(d\theta) \ . \tag{2.1}$$

The *mixability constant* C_η is the infimum of all $C > 0$ admissible for η. This infimum is usually achieved. For example, it is achieved for all $\eta > 0$ whenever Γ is compact and $e^{-\lambda(\gamma, y)}$ is continuous in γ.

An example of games $\mathfrak{G} = \langle \Omega, \Gamma, \lambda \rangle$ include *simple prediction game*:

$$\Omega = \Gamma = \{0, 1\},\ \lambda(y, \gamma) = \begin{cases} 0, & \text{if } y = \gamma, \\ 1, & \text{otherwise} \ . \end{cases}$$

Therefore, the learner tries to predict a binary sequence, 0 or 1; it suffers loss 1 if it makes a mistake. The *logarithmic loss game*:

$$\Omega = \{0, 1\},\ \Gamma = [0, 1],\ \lambda(y, \gamma) = \begin{cases} -\ln \gamma & \text{if } y = 1, \\ -\ln(1 - \gamma) & \text{if } y = 0, \end{cases}$$

and the *square-loss game*:

$$\Omega = [A, B],\ |A|, |B| < \infty,\ \Gamma = \mathbb{R},\ \lambda(y, \gamma) = (y - \gamma)^2$$

will be described in Section 2.2.2. More examples of games can be found in Section 2 of Vovk (2001).

The Aggregating Algorithm takes a learning rate $\eta > 0$, a constant C admissible for $\mathfrak{G}$ with η, and a prior distribution on experts $P_0(d\theta)$. After each step t it updates the experts' weights according to their losses:

$$P_t(d\theta) = e^{-\eta\lambda(y_t, \xi_t(\theta))} P_{t-1}(d\theta). \tag{2.2}$$

The weights of experts which suffer large loss at some step will have a smaller importance for making further predictions. The prior distribution $P_0(d\theta)$, which specifies the initial weights of experts, is an arbitrary distribution. Thus, any distribution can appear as $P_{t-1}(d\theta)$, and inequality (2.1) holds for any $P(d\theta)$.

First, we introduce the Aggregating Pseudo-Algorithm (APA). A *generalised prediction* is defined to be any function of the type $\Omega \to [0, \infty]$. The APA suffers loss $g_t(y_t)$ after choosing generalised prediction g_t at time t when the actual outcome is y_t. The APA outputs at time t the generalised prediction which is the weighted average of experts' predictions:

$$g_t(y) = -\frac{1}{\eta} \ln \int_\Theta e^{-\eta\lambda(y, \xi_t(\theta))} P^*_{t-1}(d\theta), \tag{2.3}$$

where $P^*_{t-1}(d\theta)$ are normalized weights:

$$P^*_{t-1}(d\theta) = \frac{P_{t-1}(d\theta)}{P_{t-1}(\Theta)},$$

where Θ is a *parameter space*, i.e. experts $\theta \in \Theta$ can output prediction $\xi_t(\theta) \in \Gamma$ at time t.

The AA is obtained from the APA by replacing each generalised prediction g_t by *a permitted prediction* $\Sigma(g_t)$, where the *substitution function* Σ maps every generalised prediction g into a permitted prediction $\Sigma(g) \in \Gamma$ satisfying

$$\forall y : \lambda(y, \Sigma(g)) \le C g(y). \tag{2.4}$$

By putting (2.3) in (2.4), we get (2.1). We also need to define a *superprediction*, which is a generalised prediction minorised by the loss of some prediction, i.e., a superprediction

is a function $f : \Omega \to [0, +\infty]$ such that for some $\gamma \in \Gamma$ we have $f(y) \geq \lambda(y, \gamma)$ for all $y \in \Omega$.

We will assume that the substitution function, which attains inequality (2.4), exists. The AA requires that a 'minimax' substitution function should be chosen (Vovk (1990))

$$\Sigma(g) \in \arg\min_{\gamma\in\Gamma} \sup_{y\in\Omega} \frac{\lambda(y,\gamma)}{g(y)}. \tag{2.5}$$

However, in some cases it is more computationally efficient to require that the substitution function follows

$$\Sigma(g) \in \arg\min_{\gamma\in\Gamma} \sup_{y\in\Omega} (\lambda(y,\gamma) - C_\eta g(y)) \tag{2.6}$$

and

$$(g_1(y) - g_2(y) \text{ does not depend on } y) \to (\Sigma(g_1) = \Sigma(g_2)), \tag{2.7}$$

where g_1, g_2 are the generalised predictions calculated with different weights distributions. Assumption (2.7) is always compatible with (2.6) but is typically incompatible with (2.5). A great advantage of (2.7) is that we do not need to normalise experts' weights at each step, and can calculate the pseudo-prediction from the unnormalised weights, since it will differ from (2.3) only by an additive constant.

The pseudo-code for the Aggregating Algorithm is given below.

The Aggregating Algorithm

Initialize prior distribution on experts $P_0(d\theta) = P_0^*(d\theta)$, $\theta \in \Theta$,
  a learning rate $\eta > 0$, an admissible constant C
for $t = 1, 2, \ldots$ do
  Get experts' predictions $\xi_t(\theta)$, $\theta \in \Theta$.
  Calculate generalised prediction $g : \Omega \to \mathbb{R}$,
    defined by $g_t(y) = -\frac{1}{\eta} \ln \int_\Theta e^{-\eta\lambda(y,\xi_t(\theta))} P_{t-1}^*(d\theta)$, for all $y \in \Omega$.
  Output prediction $\gamma_t := \Sigma(g_t) \in \Gamma$ using a substitution function
    $\Sigma : \mathbb{R}^\Omega \to \Gamma$, such that $\forall y : \lambda(y, \Sigma(g_t)) \leq C g_t(y)$.
  Read the outcome $y_t \in \Omega$.
  Update the weights $P_t(d\theta) = e^{-\eta\lambda(y_t,\xi_t(\theta))} P_{t-1}(d\theta)$, $\theta \in \Theta$.
  Normalize the weights $P_t^*(d\theta) = \frac{P_t(d\theta)}{\int_{\theta\in\Theta} P_t(d\theta)}$.
end for

Lemma 2.2.1. (Lemma 1 in Vovk (2001)) *For any learning rate $\eta > 0$, prior P_0,*

and $T = 1, 2, \ldots$, *the cumulative loss* L_T *of the APA with parameters* η *and* P_0 *follows*

$$L_T(\mathrm{APA}(\eta, P_0)) = -\frac{1}{\eta} \ln \int_{\theta \in \Theta} e^{-\eta L_T^\theta} P_0(d\theta). \tag{2.8}$$

Proof. The lemma is proven by induction. It is obvious from (2.3) that (2.8) holds for $T = 1$. Notice, that

$$g_T(y_T) = -\frac{1}{\eta} \ln \frac{\int_\Theta e^{-\eta \lambda(y_T, \xi_T(\theta))} P_{T-1}(d\theta)}{P_{T-1}(\Theta)} = -\frac{1}{\eta} \ln \frac{\int_\Theta e^{-\eta \lambda(y_T, \xi_T(\theta))} e^{-\eta L_{T-1}^\theta} P_0(d\theta)}{\int_\Theta e^{-\eta L_{T-1}^\theta} P_0(d\theta)}.$$

Assume that (2.8) holds for $T - 1$. Then

$$\begin{aligned} L_T(\mathrm{APA}) &= L_{T-1}(\mathrm{APA}) + g_T(y_T) \\ &= -\frac{1}{\eta} \ln \int_\Theta e^{-\eta L_{T-1}^\theta} P_0(d\theta) - \frac{1}{\eta} \ln \frac{\int_\Theta e^{-\eta L_T^\theta} P_0(d\theta)}{\int_\Theta e^{-\eta L_{T-1}^\theta} P_0(d\theta)} = -\frac{1}{\eta} \ln \int_\Theta e^{-\eta L_T^\theta} P_0(d\theta). \end{aligned}$$

□

The following lemma, which follows from Lemma 2.2.1 and (2.4), gives an upper bound on the cumulative loss of the Aggregating Algorithm.

Lemma 2.2.2. *For a game* $\mathfrak{G}$ *with an admissible constant* C*, a learning rate* η*, and any prior* P_0*, the following upper bound on the cumulative loss of the Aggregating Algorithm holds for* $T = 1, 2, \ldots$,

$$L_T(\mathrm{AA}) \le -\frac{C}{\eta} \ln \int_{\theta \in \Theta} e^{-\eta L_T^\theta} P_0(d\theta). \tag{2.9}$$

In case when the number of experts is finite we replace expert's index θ with index $i \in \Theta = \{1, 2, \ldots, N\}$. For the AA with a finite number of experts N we get the following lemma.

Lemma 2.2.3. *For a game* $\mathfrak{G}$ *with an admissible constant* C*, a learning rate* η*, and the initial distribution on experts* $p(i)$ *(such that* $\sum_{i=1}^N = 1$*), the following upper bound on the cumulative loss of the Aggregating Algorithm holds for* $T = 1, 2, \ldots$,

$$L_T(\mathrm{AA}) \le -\frac{C}{\eta} \ln \left(\sum_{i=1}^N p(i) e^{-\eta L_T^i} \right) \le -\frac{C}{\eta} \ln \left(p(i) e^{-\eta L_T^i} \right) = C L_T^i + \frac{C}{\eta} \ln \frac{1}{p(i)}. \tag{2.10}$$

An important class of the games are *mixable* games with C achieving 1. The natural choice of η is then the maximum value such that the mixability constant $C_\eta = 1$, which

is the infimum of all C admissible for η; it minimises the second term on the right-hand side of (2.10). In particular, for mixable games with finite number of experts N and the uniform initial distribution on experts, the loss of the AA satisfies

$$L_T(\mathrm{AA}) \leq L_T^i + \frac{1}{\eta} \ln N, \tag{2.11}$$

for $i = 1, \ldots, N$. For non-mixable games (such as the absolute loss game with $\lambda(y, \gamma) = |\gamma - y|$), bound (2.9) provides a trade-off. Optimising the bound is a more difficult task and may require assumptions on the behaviour of experts or the time horizon T.

2.2.2 Some examples of mixable games

In this section, we describe two important mixable games: the logarithmic loss game and the square-loss game. We show how to apply the AA for these games.

The logarithmic loss game

In this section, we discuss the logarithmic loss game $\mathfrak{G}$, where the outcome space $\Omega = \{0, 1\}$, the prediction space $\Gamma = [0, 1]$, and the loss function is defined as

$$\lambda(y, \gamma) = \begin{cases} -\ln \gamma & \text{if } y = 1, \\ -\ln(1 - \gamma) & \text{if } y = 0, \end{cases} \tag{2.12}$$

where $y \in \Omega$, $\gamma \in \Gamma$. We will show that the Aggregating Algorithm for the logarithmic loss function with $\eta = 1$ is the same as the Bayesian mixture.

The weight updates (2.2) becomes

$$P_t(d\theta) = \xi_t(\theta) P_{t-1}(d\theta) = P_0(d\theta) \prod_{i=1}^{t} \xi_i(\theta). \tag{2.13}$$

Therefore, the normalized weights: $P_t^*(d\theta) = \frac{P_t(d\theta)}{\int_\Theta P_t(d\theta)}$ are identical to the posterior distribution of θ after observing $y_1, y_2, \ldots, y_t$.

The generalised prediction (2.3) becomes the loss of the Bayesian mixture

$$g_t(y) = -\frac{1}{\eta} \ln \int \xi_t^y(\theta) P_{t-1}^*(d\theta), \tag{2.14}$$

where $\xi_t^y(\theta)$ is the prediction of the expert θ for the outcome y at the time t.

The log-loss game is mixable for $\eta \leq 1$ and the substitution function $\Sigma : \mathbb{R}^\Omega \to \Gamma$ is

simply $e^{-(\cdot)}$. The function x^η is concave for $\eta < 1$, and thus

$$\int_\Theta (\xi_t^y(\theta))^\eta Q(\theta) \le \Big(\int_\Theta \xi_t^y(\theta) Q(\theta)\Big)^\eta$$

for any y and $Q \in P(\Theta)$.

After taking negative logarithms of both parts and multiplying by $\frac{1}{\eta}$, we obtain

$$-\frac{1}{\eta}\ln\Big(\int_\Theta \xi_t^y(\theta) Q(\theta)\Big)^\eta \le -\frac{1}{\eta}\ln\int_\Theta (\xi_t^y(\theta))^\eta Q(\theta).$$

In other words, the loss of the prediction corresponding to $\eta = 1$ is less than the generalised prediction calculated with any other $\eta < 1$.

The pseudo-code for the Bayesian Algorithm is shown below.

The Bayesian Algorithm

Initialize prior distribution on experts $P_0(d\theta) = P_0^*(d\theta)$, $\theta \in \Theta$,
for $t = 1, 2, \ldots$ do
  Get experts' predictions $\xi_t(\theta)$, $\theta \in \Theta$.
  Output prediction $\gamma_t = \int_{\theta\in\Theta} \xi_t(\theta) P_{t-1}^*(d\theta)$.
  Read the outcome $y_t \in \Omega$.
  Update the weights $P_t(d\theta) = \xi_t(\theta) P_{t-1}(d\theta)$, $\theta \in \Theta$.
  Normalize the weights $P_t^*(d\theta) = \frac{P_t(d\theta)}{\int_{\theta\in\Theta} P_t(d\theta)}$.
end for

The square-loss game

In this section, we consider the square-loss game, where the outcome space $\Omega = [A, B]$, $|A|, |B| < \infty$, the prediction space $\Gamma = \mathbb{R}$ and the loss function $\lambda(y, \gamma) = (y - \gamma)^2$. The following lemma proves the mixability of the square-loss game with the restricted outcome space. The mixability for the outcome space $\Omega = [-Y, Y]$ is first proven in Vovk (2001), the case of the outcome space $\Omega = [A, B]$ with an arbitrary interval is first described in Zhdanov (2011).

Lemma 2.2.4. (Lemma 2.5 from Zhdanov (2011)). *The square-loss game is η-mixable if and only if $\eta \le \frac{2}{(B-A)^2}$.*

Proof. Consider the parametric curve $\{(\lambda(A, \gamma), \lambda(B, \gamma)) | \gamma \in \Gamma\}$. Each point on this curve corresponds to a prediction γ and the point's coordinates are the losses when $y = A$ or $y = B$ occur. Let $\Lambda = \{(x, y) |$ there is $\gamma \in \Gamma : x \ge \lambda(A, \gamma)$ and $y \ge \lambda(B, \gamma)\}$ be the set of superpredictions.

The game is η-mixable if $C_\eta = 1$ for some $\eta > 0$, i.e., according to (2.4), we can find $\gamma \in \Gamma$ such that

$$\begin{cases} \lambda(A,\gamma) \le g_t(A) \\ \lambda(B,\gamma) \le g_t(B) \end{cases}. \tag{2.15}$$

The system has a solution if $(g_t(A), g_t(B))$ falls into the set of superpredictions. We apply the transformation $\mathcal{B}_\eta : [0,+\infty]^2 \to [0,1]^2$ given by

$$\begin{pmatrix} x \\ y \end{pmatrix} \to \begin{pmatrix} e^{-\eta x} \\ e^{-\eta y} \end{pmatrix}.$$

Under the transformation $\mathcal{B}_\eta$ for the finite number of experts N, we need to solve the system:

$$\begin{cases} e^{-\eta\lambda(A,\gamma)} \ge e^{-\eta g_t(A)} = \sum_{i=1}^N p_{t-1}(i) e^{-\eta\lambda(A,\xi_t(i))} \\ e^{-\eta\lambda(B,\gamma)} \ge e^{-\eta g_t(B)} = \sum_{i=1}^N p_{t-1}(i) e^{-\eta\lambda(B,\xi_t(i))} \end{cases}.$$

The game is mixable if the system is solvable for all experts' predictions and all normalised weights. This means that the convex combination should always fall in the set $\mathcal{B}_\eta(\Lambda)$, which is possible if and only if the set $\mathcal{B}_\eta(\Lambda)$ is convex. It is equivalent to find the values of η for which the second derivative of the parametric curve

$$\{(u,v) = (e^{-\eta(\gamma-A)^2}, e^{-\eta(\gamma-B)^2}) |\ \gamma \in \Gamma\} \tag{2.16}$$

is less or equal to zero.

The first derivative of the curve (2.16) is

$$\frac{dv}{du} = \frac{dv/d\gamma}{du/d\gamma} = \frac{2\eta(B-\gamma)e^{-\eta(\gamma-B)^2}}{2\eta(A-\gamma)e^{-\eta(\gamma-A)^2}} = \frac{\gamma-B}{\gamma-A} e^{2\eta\gamma(B-A)-\eta(B^2-A^2)}.$$

And the second derivative of the curve is

$$\frac{d^2v}{du^2} = \frac{\frac{d^2v}{du d\gamma}}{\frac{du}{d\gamma}} = \frac{\left(\frac{B-A}{(\gamma-A)^2} + 2\eta(B-A)\frac{\gamma-B}{\gamma-A}\right) e^{2\eta\gamma(B-A)-\eta(B^2-A^2)}}{2\eta(A-\gamma)e^{-\eta(\gamma-A)^2}} \le 0.$$

This is equivalent to $\frac{B-A}{(\gamma-A)^2}\left(\frac{1}{2\eta(A-\gamma)} + (B-\gamma)\right) \le 0$. Thus, we have

$$\eta \le \frac{1}{2(\gamma-A)(B-\gamma)},$$

where $\gamma \in \mathbb{R}$. Since $\max_{\gamma\in\mathbb{R}}(\gamma-A)(B-\gamma) = \frac{1}{4}(B-A)^2$, the curve is concave if and only if $\eta \le \frac{2}{(B-A)^2}$. □

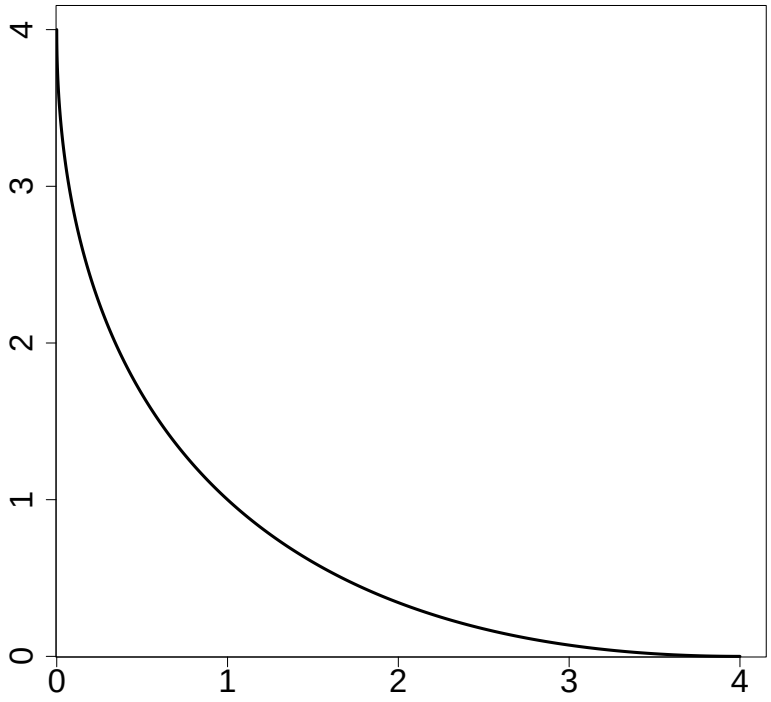


Figure 2.1: The parametric loss curve $((-1-\gamma)^2, (1-\gamma)^2),\ \gamma \in [-1,1]$.

The Figure 2.1 illustrates an example of the parametric loss curve for the restricted square-loss game with the outcome space $\Omega = \{-1,1\}$. Under the transformation $\mathcal{B}_\eta$, the set of permitted predictions is represented by the parametric curve $\left(e^{-\eta(-1-\gamma)^2}, e^{-\eta(1-\gamma)^2}\right),\ \gamma \in [-1,1]$, which is shown in Figure 2.2. The curve is convex for $\eta \leq \frac{1}{2}$. The grey area in the Figure 2.2 represents the set of superpredictions.

Now we find a substitution function for the square-loss game with the outcome space $\Omega = [A,B]$. Since we require the substitution function to follow (2.6), we have

$$(\gamma - B)^2 - g(B) = (\gamma - A)^2 - g(A).$$

Therefore, we find the formula for the substitution function of the square-loss game

$$\gamma = \frac{A+B}{2} - \frac{g(B)-g(A)}{2(B-A)}. \tag{2.17}$$

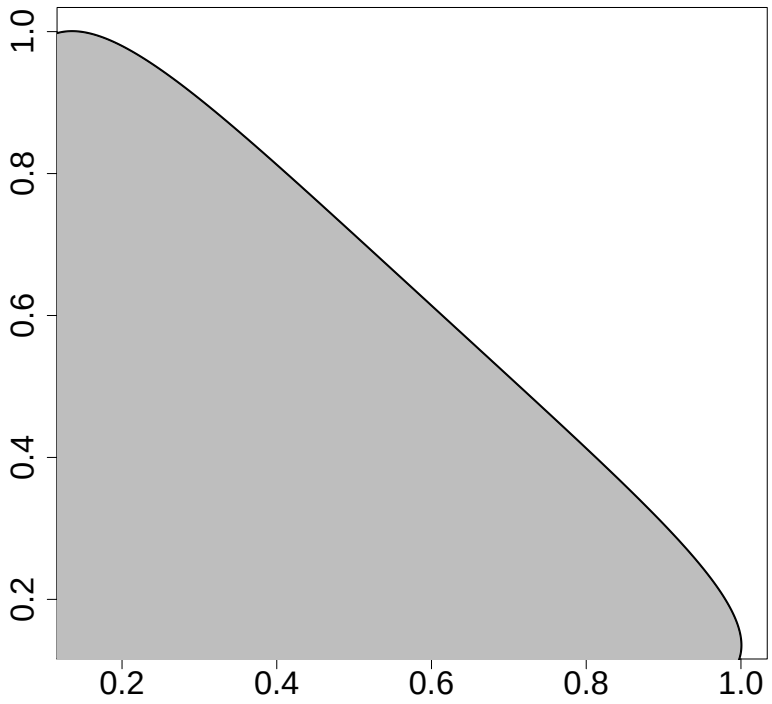


Figure 2.2: The parametric curve $\left(e^{-\frac{1}{2}(-1-\gamma)^2}, e^{-\frac{1}{2}(1-\gamma)^2}\right)$, $\gamma \in [-1, 1]$. The grey area represents the set of superpredictions.

2.3 Aggregating Algorithm for Regression

In this section, we describe an application of the Aggregating Algorithm to the problem of regression, where outcomes are continuous real numbers. In the framework of online regression, at each step nature announces a signal from a set X. Then the experts and the learner make their decisions based on the observed signal. The following protocol is the protocol of online regression:

Protocol 2.

for $t = 1, 2, \ldots$
    Nature outputs signal $x_t \in X$
    Experts output $\xi_t(\theta),\ \theta \in \Theta$
    Learner outputs $\gamma_t \in \Gamma$
    Nature announces $y_t \in \Omega$
end for

We consider the square-loss game with bounded outcome space $\Omega = [-Y, Y]$, prediction space $\Gamma = \mathbb{R}$ and loss function $\lambda(y, \gamma) = (y - \gamma)^2$. Our experts $\theta \in \mathbb{R}^n$ are linear regressions, which at the step t predict

$$\xi_t(\theta) = \theta' x_t, \tag{2.18}$$

where $x_t \in X$, and $X \subseteq \mathbb{R}^n$ is a set of input vectors.

Let $a > 0$ be an arbitrary constant. We set the prior distribution P_0 on parameters $\theta \in \Theta = \mathbb{R}^n$ to have the Gaussian distribution

$$P_0(d\theta) = \left(\frac{a\eta}{\pi}\right)^{n/2} e^{-a\eta\|\theta\|^2} d\theta. \tag{2.19}$$

After the step T the weight of the expert θ is updated according to (2.2)

$$P_T(d\theta) = e^{-\eta(y_T - \theta' x_T)^2} P_{T-1}(d\theta) = \left(\frac{a\eta}{\pi}\right)^{n/2} e^{-\eta(\sum_{t=1}^{T}(y_t - \theta' x_t)^2 + a\|\theta\|^2)} d\theta.$$

The generalised prediction (2.3) can be represented as follows

$$g_T(y) = -\frac{1}{\eta} \ln \left(\frac{1}{P_{T-1}(\Theta)} \int_{\theta \in \mathbb{R}^n} e^{-\eta((y - \theta' x_T)^2 + \sum_{t=1}^{T-1}(y_t - \theta' x_t)^2 + a\|\theta\|^2)} d\theta \right) =$$
$$-\frac{1}{\eta} \ln \left(\frac{1}{P_{T-1}(\Theta)} \int_{\theta \in \mathbb{R}^n} e^{-\eta\theta'(aI + \sum_{t=1}^{T} x_t x_t')\theta + 2\eta(\sum_{t=1}^{T-1} y_t x_t' + y x_T')\theta - \eta(\sum_{t=1}^{T-1} y_t^2 + y^2)} d\theta \right),$$

where $P_T(\Theta) = \int_{\theta \in \mathbb{R}^n} e^{-\eta(\sum_{t=1}^{T}(y_t - \theta' x_t)^2 + a\|\theta\|^2)} d\theta$ is the normalising constant for the experts' weights at the step T.

The formula for the prediction for the square-loss game is according to (2.17)

$$\begin{aligned}
\gamma &= \frac{g(-Y) - g(Y)}{4Y} \\
&= -\frac{1}{4\eta Y} \ln \frac{\int_{\theta \in \mathbb{R}^n} e^{-\eta\theta'(aI + \sum_{t=1}^{T} x_t x_t')\theta + 2\eta(\sum_{t=1}^{T-1} y_t x_t' - Y x_T')\theta - \eta(\sum_{t=1}^{T-1} y_t^2 + Y^2)} d\theta}{\int_{\theta \in \mathbb{R}^n} e^{-\eta\theta'(aI + \sum_{t=1}^{T} x_t x_t')\theta + 2\eta(\sum_{t=1}^{T-1} y_t x_t' + Y x_T')\theta - \eta(\sum_{t=1}^{T-1} y_t^2 + Y^2)} d\theta} \\
&= -\frac{1}{4\eta Y} \ln \frac{\int_{\theta \in \mathbb{R}^n} e^{-\eta\theta'(aI + \sum_{t=1}^{T} x_t x_t')\theta + 2\eta(\sum_{t=1}^{T-1} y_t x_t' - Y x_T')\theta} d\theta}{\int_{\theta \in \mathbb{R}^n} e^{-\eta\theta'(aI + \sum_{t=1}^{T} x_t x_t')\theta + 2\eta(\sum_{t=1}^{T-1} y_t x_t' + Y x_T')\theta} d\theta} \\
&= -\frac{1}{4\eta Y} \ln e^{-\eta F(aI + \sum_{t=1}^{T} x_t x_t', -2\sum_{t=1}^{T-1} y_t x_t', 2Y x_T')} \\
&= \frac{1}{4Y} F(aI + \sum_{t=1}^{T} x_t x_t', -2\sum_{t=1}^{T-1} y_t x_t', 2Y x_T') = \left(\sum_{t=1}^{T-1} y_t x_t'\right) \left(aI + \sum_{t=1}^{T} x_t x_t'\right)^{-1} x_T.
\end{aligned}$$

The fourth equality follows from Lemma A.1, the last equality follows from Lemma

A.2.

It is easy to check that the Aggregating Algorithm for Regression minimizes

$$a\|\theta\|^2 + (\theta' x_T)^2 + \sum_{t=1}^{T-1} (y_t - \theta' x_t)^2$$

in $\theta \in \mathbb{R}^n$ by taking the derivative of the quadratic form in θ.

We obtain the following algorithm

The Aggregating Algorithm for Regression

Initialize $A = aI, b = \mathbf{0}$.
for $t = 1, 2, \ldots$ do
  Read new signal $x_t \in X$.
  $A = A + x_t x_t'$
  Output prediction $\gamma_t = b' A^{-1} x_t$
  Read new outcome $y_t \in \Omega$
  $b = b + y_t x_t$
end for

Theorem 2.3.1. (Theorem 1 in Vovk (2001)) *For any positive integer n and any $a > 0$,*

$$L_T(AAR) \leq \inf_\theta (L_T^\theta + a\|\theta\|^2) + Y^2 \ln \det \left(I + \frac{1}{a} \sum_{t=1}^{T} x_t x_t' \right).$$

If, in addition, $\|x_t\|_\infty \leq B$, $\forall t$,

$$L_T(AAR) \leq \inf_\theta (L_T^\theta + a\|\theta\|^2) + nY^2 \ln \left(\frac{TB^2}{a} + 1 \right).$$

2.4 Aggregating Algorithm with Discounting

In this section, we formulate the Aggregating Algorithm for the case of discounted loss. It is essentially equivalent to the method in Chernov and Zhdanov (2010). The Aggregating Algorithm with Discounting (AAD) differs from the AA only by the use of the weights in the computation of generalised prediction g_t and the weights update.

In the standard framework of online learning the performance of learners is evaluated by means of cumulative loss. In this section, we consider the generalisation where we discount the previous losses with the discount factor which is announced at each time step.

The cumulative losses of the learner are discounted with a factor $\alpha_t \in (0, 1]$ at each step. If L_{T-1} is the discounted cumulative loss of the learner at step $T-1$, then the discounted cumulative loss of the learner at step T is defined by

$$L_T := \alpha_{T-1} L_{T-1} + \lambda_T(y_T, \gamma_T) = \sum_{t=1}^{T-1} \left(\prod_{j=t}^{T-1} \alpha_j \right) \lambda_t(y_t, \gamma_t) + \lambda_T(y_T, \gamma_T). \tag{2.20}$$

If L^θ_{T-1} is the discounted cumulative loss of the prediction strategy θ at the step $T-1$, then the discounted cumulative loss of the prediction strategy θ at the step T is defined by

$$L^\theta_T := \alpha_{T-1} L^\theta_{T-1} + \lambda_T(y_T, \xi_T(\theta)) = \sum_{t=1}^{T-1} \left(\prod_{j=t}^{T-1} \alpha_j \right) \lambda_t(y_t, \xi_t(\theta)) + \lambda_T(y_T, \xi_T(\theta)). \tag{2.21}$$

In the beginning, the losses L_0, L^θ_0 are initialized to zero. If all the discount factors are the same, i.e. $\alpha_1 = \cdots = \alpha_T = \alpha$, then we have a case of exponential smoothing. At each step the dependence on the loss at the previous steps exponentially decreases: the initial loss is discounted by α^{T-1} at the step T. Note that if $\alpha_t = 1$ at each time step t then we have the standard framework of undiscounted loss.

Learner and experts work according to the following protocol:

Protocol 3.

$L_0 := 0$
$L^\theta_0 := 0$
for $t = 1, 2, \ldots$
    Accountant announces $\alpha_{t-1} \in (0, 1]$
    Nature announces $x_t \in X \subseteq \mathbb{R}^n$
    Experts output $\xi_t(\theta),\ \theta \in \Theta$
    Learner outputs $\gamma_t \in \Gamma$
    Nature announces $y_t \in \Omega$
    $L^\theta_t := \alpha_{t-1} L^\theta_{t-1} + \lambda(y_t, \xi_t(\theta)),\ \theta \in \Theta$
    $L_t := \alpha_{t-1} L_{t-1} + \lambda(y_t, \gamma_t)$
end for

For the AAD we denote the discounted weight of expert θ as $\tilde{P}(\theta)$. We initialize a prior distribution on experts $P_0(d\theta)$, $\theta \in \Theta$ and initial discounted weights of experts $\tilde{P}_0(\theta) = 1$.

Instead of (2.2) the AAD updates weights according to

$$\tilde{P}_t(\theta) = \left(\tilde{P}_{t-1}(\theta)\right)^{\alpha_{t-1}} e^{-\eta\lambda(y_t,\xi_t(\theta))}. \tag{2.22}$$

The generalised prediction of the AAD is

$$g_t(y) = -\frac{1}{\eta}\ln\int_{\theta\in\Theta} P_0(d\theta)\left(\tilde{P}^*_{t-1}(\theta)\right)^{\alpha_{t-1}} e^{-\eta\lambda(y,\xi_t(\theta))}, \tag{2.23}$$

where

$$\tilde{P}^*_{t-1}(\theta) = \frac{\tilde{P}_{t-1}(\theta)}{\int_{\theta\in\Theta} P_0(d\theta)\tilde{P}_{t-1}(\theta)}. \tag{2.24}$$

First, we show that Lemma 2.2.1 holds for the discounted cumulative loss of the APA.

Lemma 2.4.1. *For any learning rate $\eta > 0$, prior P_0, any sequence $\alpha_T \in (0,1]$, and $T = 1, 2, \ldots$, the discounted cumulative loss L_T of the APA with parameters η and P_0 follows*

$$L_T(\mathrm{APA}(\eta, P_0)) = -\frac{1}{\eta}\ln\int_{\theta\in\Theta} e^{-\eta L_T^\theta} P_0(d\theta). \tag{2.25}$$

Proof. The lemma is proven by induction. It is obvious from (2.23) that (2.25) holds for $T = 1$. Notice, that

$$\begin{aligned} g_t(y) &= -\frac{1}{\eta}\ln\int_{\theta\in\Theta} P_0(d\theta)\left(\tilde{P}^*_{t-1}(\theta)\right)^{\alpha_{t-1}} e^{-\eta\lambda(y,\xi_t(\theta))} \\ &= -\frac{1}{\eta}\ln\frac{\int_{\theta\in\Theta} P_0(d\theta)\left(\tilde{P}_{t-1}(\theta)\right)^{\alpha_{t-1}} e^{-\eta\lambda(y,\xi_t(\theta))}}{\left(\int_{\theta\in\Theta} P_0(d\theta)\tilde{P}_{t-1}(\theta)\right)^{\alpha_{T-1}}} \\ &= -\frac{1}{\eta}\ln\frac{\int_{\theta\in\Theta} P_0(d\theta)e^{-\eta L_T^\theta}}{\left(\int_{\theta\in\Theta} P_0(d\theta)e^{-\eta L_{T-1}^\theta}\right)^{\alpha_{T-1}}}. \end{aligned}$$

Assume that (2.25) holds for $T-1$. According to (2.20) the discounted cumulative loss of the APA at time T

$$\begin{aligned} L_T(\mathrm{APA}) = \alpha_{T-1}L_{T-1}(\mathrm{APA}) + g_T(y_T) &= -\frac{1}{\eta}\ln\left(\int_{\theta\in\Theta} P_0(d\theta)e^{-\eta L_{T-1}^\theta}\right)^{\alpha_{T-1}} \\ -\frac{1}{\eta}\ln\frac{\int_{\theta\in\Theta} P_0(d\theta)e^{-\eta L_T^\theta}}{\left(\int_{\theta\in\Theta} P_0(d\theta)e^{-\eta L_{T-1}^\theta}\right)^{\alpha_{T-1}}} &= -\frac{1}{\eta}\ln\int_{\theta\in\Theta} P_0(d\theta)e^{-\eta L_T^\theta}. \end{aligned}$$

□

Now we prove the analogue of Lemma 2.2.2 for mixable games in the discounted case.

Lemma 2.4.2. *For any learning rate $\eta > 0$, initial prior P_0, every sequence $\alpha_T \in (0, 1]$, and $T = 1, 2, \ldots$, the discounted cumulative loss L_T of the AAD with parameters η and P_0 follows*

$$L_T(AAD(\eta, P_0)) \leq -\frac{1}{\eta} \ln \int_\Theta e^{-\eta L_T^\theta} P_0(d\theta). \tag{2.26}$$

Proof. The weights update for AAD is

$$\tilde{P}_t(\theta) = \left(\tilde{P}_{t-1}(\theta)\right)^{\alpha_{t-1}} e^{-\eta\lambda(y_t, \xi_t(\theta))} = e^{-\eta L_t^\theta}. \tag{2.27}$$

We will prove (2.26) by induction. At step $t+1$ we can re-write inequality (2.4) as follows

$$\begin{aligned} e^{-\eta\lambda(y_{t+1}, \gamma_{t+1})} &\geq \int_\Theta P_0(d\theta) \left(\tilde{P}_t^*(\theta)\right)^{\alpha_t} e^{-\eta\lambda(y_{t+1}, \xi_{t+1}(\theta))} \\ &= \int_\Theta P_0(d\theta) \frac{e^{-\eta\alpha_t L_t^\theta}}{\left(\int_\Theta P_0(d\theta) e^{-\eta L_t^\theta}\right)^{\alpha_t}} e^{-\eta\lambda(y_{t+1}, \xi_{t+1}(\theta))}. \end{aligned} \tag{2.28}$$

Suppose that (2.26) is true for the step t. By putting the inequality (2.26) for step t in the power $0 < \alpha_t \leq 1$ we obtain

$$e^{-\eta\alpha_t L_t} \geq \left(\int_\Theta P_0(d\theta) e^{-\eta L_t^\theta}\right)^{\alpha_t}.$$

By putting the last inequality in the denominator of (2.28) we obtain

$$e^{-\eta\lambda(y_{t+1}, \gamma_{t+1})} \geq \frac{\int_\Theta e^{-\eta\lambda(y_{t+1}, \xi_{t+1}(\theta)) - \eta\alpha_t L_t^\theta} P_0(d\theta)}{e^{-\eta L_t \alpha_t}}.$$

By multiplying by the denominator we have

$$e^{-\eta L_{t+1}} \geq \int_\Theta e^{-\eta L_{t+1}^\theta} P_0(d\theta).$$

By taking a natural logarithm of both parts and multiplying by $-\frac{1}{\eta}$ we obtain (2.26). □

The pseudo-code for AAD is given below

The Aggregating Algorithm with Discounting

Initialize prior distribution on experts $P_0(d\theta)$, $\theta \in \Theta$.
Initialize discounted weights of experts $\tilde{P}_0^*(\theta) = \tilde{P}_0(\theta) = 1$,
  a learning rate $\eta > 0$.
for $t = 1, 2, \ldots$ do
  Get discount $\alpha_{t-1} \in (0, 1]$.
  Get experts' predictions $\xi_t(\theta)$, $\theta \in \Theta$.
  Calculate generalised prediction $g : \Omega \to \mathbb{R}$, defined by
    $g_t(y) = -\frac{1}{\eta} \ln \int_{\theta \in \Theta} P_0(d\theta) \left(\tilde{P}_{t-1}^*(\theta)\right)^{\alpha_{t-1}} e^{-\eta\lambda(y,\xi_t(\theta))}$, for all $y \in \Omega$.
  Output prediction $\gamma_t := \Sigma(g_t) \in \Gamma$.
  Update the weights $\tilde{P}_t(\theta) = \left(\tilde{P}_{t-1}(\theta)\right)^{\alpha_{t-1}} e^{-\eta\lambda(y_t,\xi_t(\theta))}$, $\theta \in \Theta$.
  Normalize the weights $\tilde{P}_{t-1}^*(\theta) = \frac{\tilde{P}_{t-1}(\theta)}{\int_{\theta \in \Theta} P_0(d\theta)\tilde{P}_{t-1}(\theta)}$.
end for

2.5 Weak Aggregating Algorithm

There are interesting games that are not mixable, for example, the absolute loss game $\mathfrak{G}$, where the outcome space $\Omega = \{0, 1\}$, the prediction space $\Gamma = [0, 1]$ and the loss function $\lambda(y, \gamma) = |y - \gamma|$. The Aggregating Algorithm still works for some of such games, but it allows us to achieve only values of $C_\eta > 1$. In this section, we describe a different approach to non-mixable games. We fix $C_\eta = 1$ but consider the additive term that can grow when the time T increases. The approach is called the Weak Aggregating Algorithm (WAA) which solves the problem of predicting as well as the best expert up to an additive regret term of the order $\sqrt{T}$.

As in the standard framework of prediction with expert advice, the learner has an access to experts' predictions $\xi_t(\theta)$, $\theta \in \Theta$ at each time step t. The WAA maintains experts' weights $P_t(d\theta)$, $t = 1, \ldots, T$. After each step t the WAA updates the weights of the experts according to their losses:

$$P_t(d\theta) = \exp\left(-\frac{cL_{t-1}^\theta}{\sqrt{t}}\right) P_0(d\theta), \tag{2.29}$$

where $P_0(d\theta)$ is the initial weights of experts and c is a positive parameter.

The prediction of WAA is a weighted average of the experts' predictions:

$$\gamma_t = \int_\Theta \xi_t(\theta) P_{t-1}^*(d\theta), \tag{2.30}$$

where $P^*_{t-1}(d\theta)$ are normalized weights:

$$P^*_{t-1}(d\theta) = \frac{P_{t-1}(d\theta)}{P_{t-1}(\Theta)}.$$

In the finite case, an integral in (2.30) is replaced by a weighted sum of experts' predictions $\xi_t(i),\ i = 1, \dots, N$.

In particular, when there are finitely many experts $i \in \Theta = \{1, \dots, N\}$ for bounded games the following lemma holds.

Lemma 2.5.1. (Lemma 11 in Kalnishkan and Vyugin (2008)) *For every $L > 0$, every game $\langle \Omega, \Gamma, \lambda \rangle$ such that $|\Omega| < +\infty$ with $\lambda(y, \gamma) \le L$ for all $y \in \Omega$ and $\gamma \in \Gamma$ and every $N = 1, 2, \dots$ for every merging strategy for N experts that follows the WAA with initial weights $p(1), p(2), \dots, p(N) \in [0, 1]$ such that $\sum_{i=1}^N p(i) = 1$ and $c > 0$ the bound*

$$L_T \le L^i_T + \sqrt{T}\left(\frac{1}{c}\ln\frac{1}{p(i)} + cL^2\right),$$

is guaranteed for every $T = 1, 2, \dots$ and every $i = 1, 2, \dots, N$.

After taking equal initial weights $p(1) = p(2) = \dots = p(N) = 1/N$ in the WAA, the additive term reduces to $(cL^2 + (\ln N)/c)\sqrt{T}$. When $c = \sqrt{\ln N}/L$, this expression reaches its minimum. The following corollary shows that the WAA allows us to obtain additive regret of the order $\sqrt{T}$.

Corollary 1. (Corollary 14 in Kalnishkan and Vyugin (2008)) *Under the conditions of Lemma 2.5.1, there is a merging strategy such that the bound*

$$L_T \le L^i_T + 2L\sqrt{T \ln N}$$

is guaranteed.

Applying Lemma 2.5.1 for an infinite number of experts and taking a positive constant $c = 1$, we get the following Lemma.

Lemma 2.5.2. (Lemma 2 in Levina et al. (2010)) *Let $\lambda(y, \gamma) \le L$ for all $y \in \Omega$ and $\gamma \in \Gamma$. The WAA guarantees that, for all T*

$$L_T \le \sqrt{T}\left(-\ln \int_\Theta \exp\left(-\frac{L^\theta_T}{\sqrt{T}}\right) P_0(d\theta) + L^2\right).$$

Chapter 3

Aggregating Algorithm for prediction of packs

In this chapter, we formulate a protocol for prediction of vector-valued outcomes which we call 'packs'. It naturally applies to the situations when we need to provide the predictions of several outcomes beforehand. Under the prediction of packs protocol, the learner must make a few predictions without seeing the respective outcomes and then the outcomes are revealed in one go. We consider this protocol to be a special case of online prediction under delayed feedback. We develop the theory of prediction with expert advice for packs by generalising the concept of mixability for vector-valued outcomes. We propose a number of merging algorithms for prediction of packs with tight worst-case loss upper bounds similar to those for the Aggregating Algorithm. Unlike existing algorithms for delayed feedback settings, our algorithms do not depend on the order of outcomes in a pack. Empirical experiments on sports and house price datasets are carried out to study the performance of the new algorithms and compare them against an existing method.

3.1 Introduction

In the basic online prediction protocol, at step t the learner outputs a prediction γ_t and then immediately sees the true outcome y_t, which is the feedback. The quality of the prediction is assessed by a loss function $\lambda(y, \gamma)$ measuring the discrepancy between the prediction and outcome or, generally speaking, quantifying the (adverse) effect when a prediction γ confronts the outcome y. The performance of the learner is assessed by the cumulative loss over T trials $\sum_{t=1}^{T} \lambda(y_t, \gamma_t)$.

In this chapter, we are concerned with the problem of prediction with expert advice

with delayed feedback. In a protocol with delayed feedback, there may be a delay getting true outcomes y_t. The learner may need to make a few predictions before actually seeing the outcomes of past trials. We will consider a special case of that protocol when outcomes come in *packs*: the learner needs to make a few predictions, then all outcomes are revealed, and again a few predictions need to be made.

The similar framework was considered in bandit problem applied to clinical trials which are run in *batches*: groups of patients have to be treated simultaneously (Perchet et al. (2016)). It is shown that a very small number of batches gives close to minimax optimal regret bounds of the order $C\sqrt{T}$. On the contrary, in our task, the size of batches does not have to be small, and it can vary with time. Second, we show that by applying the framework of prediction with expert advice to the considered problem, we can achieve the constant regret term in the case of a finite number of experts.

A problem naturally fitting this framework is provided by aggregation of bookmakers' predictions. Vovk and Zhdanov (2009) predict the outcomes of sports matches on the basis of probabilities calculated from the odds quoted by bookmakers. If matches occur one by one, the problem naturally fits the basic prediction with expert advice framework. However, it is common, e.g., in the English Premier League, that a few matches occur on the same day. It would be natural to try and predict all the outcomes beforehand. All matches from the same day can be treated as a pack in our framework.

We develop a theory of prediction with expert advice for packs by extending the Aggregating Algorithm described in Section 2.2.1. In Section 3.3, we develop the theory of mixability for games with packs of outcomes. Theorem 3.3.3 shows that a game involving packs of K outcomes has the same profile of mixability constants as the original game with single outcomes, but the learning rate divides by K. This observation allows us to handle packs of constant size. However, as discussed above, in really interesting cases the size of the pack varies with time and thus the mixability of the environment varies from step to step. This problem can be approached in different ways resulting in different algorithms with different performance bounds. In Section 3.4, we introduce three *Aggregating Algorithms for Prediction of Packs*, AAP-max, AAP-incremental, and AAP-current and obtain worst-case upper bounds for their cumulative loss.

The general theory of the AA (Vovk, 1998) allows us to show in Section 3.5 that in some sense our bounds are optimal. In Section 3.5.1, we provide a standalone derivation of a lower bound for the mix-loss framework of Adamskiy et al. (2016). However, the question of optimality for packs is far from being fully resolved and requires further investigation.

As mentioned before, the problem of prediction of packs can be considered as a

special case of the delayed feedback problem. This problem has been studied mostly within the framework of online convex optimisation (Zinkevich, 2003; Joulani et al., 2013; Quanrud and Khashabi, 2015). The terminology and approach of online convex optimisation is different from ours, which go back to Littlestone and Warmuth (1994) and were surveyed by Cesa-Bianchi and Lugosi (2006).

The problem of prediction with expert advice for delayed feedback can be solved by running parallel copies of algorithms predicting single outcomes. In Section 3.2.2, we describe the algorithm Parallel Copies, which is essentially BOLD of Joulani et al. (2013) using the Aggregating Algorithm as a base algorithm for single outcomes. The theory of the Aggregating Algorithm implies a worst-case upper bound on the loss of Parallel Copies. We see that the regret term multiplies by the maximum delay or pack size as in the existing literature (Joulani et al., 2013; Weinberger and Ordentlich, 2002).

The bounds we obtain for our new algorithms are the same (AAP-max and AAP-incremental) or incompatible (AAP-current) with the bound for Parallel Copies. We discuss the bounds in Section 3.5 and then in Section 3.6 carry out an empirical comparison of the performance of the algorithms.

For experiments we predict outcomes of sports matches based on bookmakers' odds and work out house prices based on descriptions of houses. The sports datasets include football matches, which naturally contain packs, and tennis matches, where we introduce packs artificially in two different ways. The house price datasets contain records of property transactions in Ames in the US and the London area. The datasets only record the month of a transaction, so they are naturally organised in packs. The house price experiments follow the approach of Kalnishkan et al. (2015): prediction with expert advice can be used to find relevant past information. Predictors trained on different sections of past data can be combined in the online mode so that relevant past data is used for prediction.

The performance of the Parallel Copies algorithm depends on the order of outcomes in the packs, while our algorithms are order-independent. We compare the cumulative loss of our algorithms against the loss of Parallel Copies averaged over random permutations within packs. We conclude that while Parallel Copies can perform very well, especially if the order of outcomes in the packs carries useful information, the loss of our algorithms is always close to the average loss of Parallel Copies and some algorithms beat the average.

We then compare our algorithms to each other concluding that AAP-max is the worst and AAP-current outperforms AAP-incremental if the ratio of the maximum to the minimum pack size is small.

3.2 Preliminaries and Background

In this section, we introduce the framework of prediction of packs and review connections with the literature.

3.2.1 Protocols for Prediction of Packs

A game $\mathfrak{G} = \langle \Omega, \Gamma, \lambda \rangle$ contains an outcome space Ω, prediction space Γ, and loss function $\lambda : \Gamma \times \Omega \to [0, +\infty]$.

In the classical protocol, the learner makes a prediction (possibly upon using a signal) and then the outcome is immediately revealed. In this chapter, we consider an extension of this protocol and allow the outcomes to come in *packs* of possibly varying size. The learner must produce a pack of predictions before seeing the true outcomes. The following protocol summarises the framework.

Protocol 4 (Prediction of packs)**.**

FOR $t = 1, 2, \ldots$
  nature announces $x_{t,k} \in X$, $k = 1, 2, \ldots, K_t$
  learner outputs $\gamma_{t,k} \in \Gamma$, $k = 1, 2, \ldots, K_t$
  nature announces $y_{t,k} \in \Omega$, $k = 1, 2, \ldots, K_t$
  learner suffers losses $\lambda(y_{t,k}, \gamma_{t,k})$, $k = 1, 2, \ldots, K_t$
ENDFOR

At every trial t the learner needs to make K_t predictions rather than one. We will be speaking of a pack of the learner's predictions $\gamma_{t,k} \in \Gamma$, $k = 1, 2, \ldots, K_t$, a pack of outcomes $y_{t,k} \in \Omega$, $k = 1, 2, \ldots, K_t$ etc.

In this chapter, we assume a full information environment. The learner knows Ω, Γ, and λ. It sees all $y_{t,k}$ as they become available. On the other hand, we make no assumptions on the mechanism generating $y_{t,k}$ and will be interested in worst-case guarantees for the loss. The outcomes do not have to satisfy probabilistic assumptions such as i.i.d., and can behave maliciously.

Now let $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_N$ be experts working according to Protocol 4. Suppose that on each turn, their predictions are made available to a learner $\mathcal{S}$ as a special kind of side information. The learner then works according to the following protocol.

Protocol 5 (Prediction of packs with expert advice)**.**

FOR $t = 1, 2, \ldots$
  each expert $\mathcal{E}_i$, $i = 1, 2, \ldots, N$, announces
    predictions $\xi_{t,k}(i) \in \Gamma$, $k = 1, 2, \ldots, K_t$

```
    learner outputs predictions γ_{t,k} ∈ Γ, k = 1,2,...,K_t
    nature announces y_{t,k} ∈ Ω, k = 1,2,...,K_t
    each expert E_i, i = 1,2,...,N, suffers
      losses λ(y_{t,k}, ξ_{t,k}(i)), k = 1,2,...,K_t
    learner suffers losses λ(y_{t,k}, γ_{t,k}), k = 1,2,...,K_t
ENDFOR
```

The goal of the learner in this setup is to suffer loss close to the best expert in retrospect (in whatever formal sense that can be achieved). We look for merging strategies for the learner making sure that the learner achieves low cumulative loss as compared to the experts; we will see that one can quantify cumulative loss in different ways.

The merging strategies we are interested in are computable in some natural sense; we will not make exact statements about computability though. We do not impose any restrictions on experts. In what follows, the reader may substitute the clause 'for all predictions $\xi_{t,k}(i)$' appearing in Protocol 5 for the more intuitive clause 'for all experts'.

There can be subtle variations of this protocol. Instead of getting all K_t predictions from each expert at once, the learner may be getting predictions for each outcome one by one and making its own before seeing the next set of experts' predictions. For most of our analysis this does not matter, as we will see later. The only thing that matters is that the outcomes come in one go after the learner has finished predicting the pack.

3.2.2 Delayed Feedback Approach

The protocol of prediction with packs can be considered as a special case of the delayed feedback settings surveyed by Joulani et al. (2013).

In the delayed feedback prediction with expert advice protocol, on every step the learner gets just one round of predictions from each expert and must produce its own. However, the outcome corresponding to these predictions may become available later. If it is revealed on the same trial as in Section 2.2.1, we say that the delay is one. If it is revealed on the next trial, the delay equals two, etc. Prediction of packs of size not exceeding K can be considered as prediction with delays not exceeding K.

The algorithm BOLD (Joulani et al., 2013) for this protocol works as follows. Take an algorithm working with delays of 1 (or packs of size 1); we will call it the base algorithm. In order to merge experts' predictions, we will run several copies of the base algorithm. They are independent in the sense that they do not share information. Each copy will repeatedly receive experts' predictions for merging, output a prediction, and then wait for the outcome corresponding to the prediction. At every moment a

copy of the base algorithm either knows all outcomes for the predictions it has made or is expecting the outcome corresponding to the last prediction. In the former case we say that the copy is ready (to merge more experts' predictions) and in the later case we say that the copy is blocked (and cannot merge).

At each trial, when a new round of experts' predictions arrives, we pick a ready algorithm (say, one with the lowest number) and give the experts' predictions to it. It produces a prediction, which we pass on, and the algorithm becomes blocked until the outcome for that trial arrives. If all algorithms are currently blocked, we start a new copy of the base algorithm.

Suppose that we are playing a game $\mathfrak{G}$ and C is admissible for $\mathfrak{G}$ with a learning rate η. For the base algorithm take AA with C, η and initial weights $p(1), p(2), \ldots, p(N)$. If the delay never exceeds D, we never need more than D algorithms in the array and each of them suffers loss satisfying inequality (2.10). Summing the bounds up, we get that the loss of $\mathcal{S}$ using this strategy satisfies

$$L_T \le CL_T^i + \frac{CD}{\eta} \ln \frac{1}{p(i)} \tag{3.1}$$

for every expert $\mathcal{E}_i$, where the sum in L_T is taken over all outcomes revealed before step $T+1$. The value of D does not need to be known in advance; we can always expand the array as the delay increases. We will refer to the combination of BOLD and AA in the above fashion as the *Parallel Copies* algorithm.

For Protocol 5 described in Section 3.2.1 we can define plain cumulative loss

$$L_T = \sum_{t=1}^{T} \sum_{k=1}^{K_t} \lambda(y_{t,k}, \gamma_{t,k}) \ , \tag{3.2}$$

$$L_T^i = \sum_{t=1}^{T} \sum_{k=1}^{K_t} \lambda(y_{t,k}, \xi_{t,k}(i)) \ , i = 1, 2, \ldots, N. \tag{3.3}$$

Then (3.1) implies

$$L_T \le CL_T^i + \frac{CK}{\eta} \ln \frac{1}{p(i)} \ , \tag{3.4}$$

where $K = \max_{t=1,2,\ldots,T} K_t$, for $\mathcal{S}$ following Parallel Copies.

However, the Parallel Copies algorithm has two disadvantages. First, it requires us to maintain D arrays of experts' weights. Each copy of AA needs to maintain N weights, one for each expert. If packs of size D come up, we will need D such arrays. Secondly, and more importantly, the algorithm depends on the order of predictions in the pack. It matters what copy of the AA will pick a particular round of experts' predictions and the result is not invariant w.r.t. the order within the packs.

Below we will build algorithms that are order-independent and have loss bounds both similar (Section 3.4.1) and essentially different (Section 3.4.2) from (3.4). Our method is based on a generalisation of the concept of mixability and a direct application of AA to packs. The resulting algorithms will maintain one array of N weights (or losses).

3.3 Mixability

In this section, we extend the concept of mixability defined in Section 2.2.1 to packs of outcomes. This will be a key tool for the analysis of the algorithms we will construct. We need upper bound on admissible constants in order to get upper loss bounds and lower bounds in order to establish some form of optimality. As we cannot restrict ourselves to packs of constant size, we need to consider suboptimal constants too.

For a game $\mathfrak{G} = \langle \Omega, \Gamma, \lambda \rangle$ and a positive integer K consider the game $\mathfrak{G}^K$ with the outcome and prediction space given by the Cartesian products Ω^K and Γ^K and the loss function $\lambda^{(K)}((y_1, y_2, \dots, y_K), (\gamma_1, \gamma_2, \dots, \gamma_K)) = \sum_{k=1}^K \lambda(y_k, \gamma_k)$. What are the mixability constants for this game? Let C_η be the constants for $\mathfrak{G}$ and $C_\eta^{(K)}$ be the constants for $\mathfrak{G}^K$.

The following lemma provides an upper bound for $C_\eta^{(K)}$.

Lemma 3.3.1. *If $C > 0$ is admissible for a game $\mathfrak{G}$ with a learning rate $\eta > 0$, then C is admissible for the game $\mathfrak{G}^K$ with the learning rate η/K.*

Proof. Take N predictions in the game $\mathfrak{G}^K$, $\xi(1) = (\xi_1^1, \xi_2^1, \dots, \xi_K^1)$, $\dots$, $\xi(N) = (\xi_1^N, \xi_2^N, \dots, \xi_K^N)$ and weights $p(1), p(2), \dots, p(N)$. Since C is admissible for $\mathfrak{G}$, there are predictions $\gamma_1, \gamma_2, \dots, \gamma_K \in \Gamma$ such that

$$e^{-\eta\lambda(y_k,\gamma_k)/C} \geq \sum_{i=1}^N p(i) e^{-\eta\lambda(y_k,\xi_k^i)}$$

for every $y_k \in \Omega$. We will use $(\gamma_1, \gamma_2, \dots, \gamma_K) \in \Gamma^K$ to show that C is admissible for $\mathfrak{G}^K$. Multiplying the inequalities we get

$$e^{-\eta\sum_{k=1}^K \lambda(y_k,\gamma_k)/C} \geq \prod_{k=1}^K \sum_{i=1}^N p(i) e^{-\eta\lambda(y_k,\xi_k^i)} \ .$$

We will now apply the generalised Hölder inequality. On measure spaces (S, Σ, μ) formed by the space S, the σ-field Σ of measurable sets in this space, and the measure μ defined on this σ-field, the inequality states that for all measurable real- or

complex-valued functions $f_1, \dots, f_K$ defined on S: $\| \prod_{k=1}^K f_k \|_r \le \prod_{k=1}^K \| f_k \|_{r_k}$, where $\sum_{k=1}^K 1/r_k = 1/r$. This follows by induction from the version of the inequality given by (Loève, 1977, Section 9.3). Interpreting a vector $x_k = (x_k(1), x_k(2), \dots, x_k(N)) \in \mathbb{R}^N$ as a function on a discrete space $\{1, 2, \dots, N\}$ and introducing on this space a measure $p(i)$, $i = 1, 2, \dots, N$, we obtain

$$\left(\sum_{i=1}^N p(i) \left| \prod_{k=1}^K x_k(i) \right|^r \right)^{1/r} \le \prod_{k=1}^K \left(\sum_{i=1}^N p(i) \left| x_k(i) \right|^{r_k} \right)^{1/r_k} .$$

Letting $r_k = 1$ and $r = 1/K$ we get

$$\begin{aligned} e^{-\eta \sum_{k=1}^K \lambda(y_k, \gamma_k)/C} &\ge \prod_{k=1}^K \sum_{i=1}^N p(i) e^{-\eta \lambda(y_k, \xi_k^i)} \\ &\ge \left(\sum_{i=1}^N p(i) e^{-\sum_{k=1}^K \eta \lambda(y_k, \xi_k^i)/K} \right)^K . \end{aligned}$$

Raising the resulting inequality to the power $1/K$ completes the proof. □

Remark 1. *Note that the proof of the lemma offers a constructive way of solving (2.1) for $\mathfrak{G}^K$ provided we know how to solve (2.1) for $\mathfrak{G}$. Namely, to solve (2.1) for $\mathfrak{G}^K$ with the learning rate η/K, we solve K systems for $\mathfrak{G}$ with the learning rate η.*

We will now show that the admissible constants given by Lemma 3.3.1 cannot be decreased for a wide class of games.

Lemma 3.3.2. *Let a game $\mathfrak{G}$ have a convex set of superpredictions. If $C > 0$ is admissible for $\mathfrak{G}^K$ with a learning rate $\eta/K > 0$, then C is admissible for $\mathfrak{G}$ with the learning rate η.*

The requirement of convexity is not too restrictive. For a wide class of games the following implication holds. If the game is mixable (i.e., $C_\eta = 1$ for some $\eta > 0$), then its set of superpredictions is convex. (Kalnishkan et al., 2004, Lemma 7) essentially prove this for games with finite sets of outcomes.

Proof. Since $C > 0$ is admissible for $\mathfrak{G}^K$ with the learning rate $\eta/K > 0$, for every N arrays of predictions $\xi(1) = (\xi_1^1, \xi_2^1, \dots, \xi_K^1)$, $\dots$, $\xi(N) = (\xi_1^N, \xi_2^N, \dots, \xi_K^N)$ and weights $p(1), p(2), \dots, p(N)$ there are $\gamma_1, \gamma_2, \dots, \gamma_K \in \Gamma$ such that

$$\sum_{k=1}^K \lambda(y_k, \gamma_k) \le -\frac{C}{\eta/K} \ln \sum_{i=1}^N p(i) e^{-\eta \sum_{k=1}^K \lambda(y_k, \xi_k^i)/K}$$

for all $y_1, y_2, \ldots, y_K \in \Omega$.

Given N predictions $\xi_1, \xi_2, \ldots, \xi_N \in \Gamma$, we can turn them into predictions from Γ^K by considering N arrays $\xi(i) = (\xi_i, \ldots, \xi_i) \in \Gamma^K$, $i = 1, 2, \ldots, N$. By the above there are predictions $\gamma_1^*, \gamma_2^*, \ldots, \gamma_K^* \in \Gamma$ satisfying

$$\frac{1}{K}\sum_{k=1}^{K} \lambda(y, \gamma_k^*) \le -\frac{C}{\eta} \ln \sum_{i=1}^{N} p(i) e^{-\eta \lambda(y, \xi_i)}$$

for all $y \in \Omega$ (we let $y_1 = y_2 = \ldots = y_K = y$).

We have found a prediction from Γ^K, but we need one from Γ. The problem is that γ_k^* do not have to be equal. However, $\sum_{k=1}^{K} \lambda(y, \gamma_k^*)/K$ is a convex combination of superpredictions w.r.t. $\mathfrak{G}$. Since the set of superpredictions is convex, this expression is a superprediction and there is $\gamma \in \Gamma$ such that $\lambda(y, \gamma) \le \sum_{k=1}^{K} \lambda(y, \gamma_k^*)/K$ for all $y \in \Omega$. □

Since C_η and $C_{\eta/K}^{(K)}$ are the infimums of admissible values, Lemma 3.3.1 and Lemma 3.3.2 can be combined into the following theorem.

Theorem 3.3.3. *For a game $\mathfrak{G}$ with a convex set of superprediction, any positive integer K and learning rate $\eta > 0$, we have $C_{\eta/K}^{(K)} = C_\eta$.*

This theorem allows us to merge experts' predictions in an optimal way for the case when all packs are of the same size. In this case, we simply apply Proposition 2.10 and all the existing theory of the AA to the game $\mathfrak{G}^K$.

In order to analyse the case when pack sizes vary, we need to make a simple observation on the behaviour of $C_{\eta/K_2}^{(K_1)}$ for $K_1 \le K_2$.

Lemma 3.3.4. *For every game $\mathfrak{G}$, if $C > 0$ is admissible with a learning rate $\eta_1 > 0$, it is also admissible with every $\eta_2 \le \eta_1$. Hence the value of C_η is non-decreasing in η.*

Proof. Raising the inequality

$$e^{-\eta_1 \lambda(y,\gamma)/C} \ge \sum_{i=1}^{N} p(i) e^{-\eta_1 \lambda(y, \xi(i))}$$

to the power $\eta_2/\eta_1 \le 1$ and using Jensen's inequality we get

$$e^{-\eta_2 \lambda(y,\gamma)/C} \ge \left(\sum_{i=1}^{N} p(i) e^{-\eta_1 \lambda(y,\xi(i))} \right)^{\eta_2/\eta_1} \ge \sum_{i=1}^{N} p(i) e^{-\eta_2 \lambda(y,\xi(i))} \quad .$$

Thus as we decrease η, the infimum of admissible C can only go down. □

Corollary 2. *For every game $\mathfrak{G}$ and positive integers $K_1 \le K_2$, we have $C^{(K_1)}_{\eta/K_2} \le C^{(K_1)}_{\eta/K_1}$.*

Proof. The proof is by applying Lemma 3.3.4 to $\mathfrak{G}^{K_1}$. □

Remark 2. *The proofs of the lemma and corollary are again constructive in the following sense. If we know how to solve (2.1) for $\mathfrak{G}$ with a learning rate η_1 and an admissible C, we can solve (2.1) for $\eta_2 \le \eta_1$ and the same C.*

Suppose we play the game $\mathfrak{G}^{K_1}$ but have to use the learning rate η/K_2, where $K_2 \ge K_1$, with C admissible for $\mathfrak{G}$ with η. To solve (2.1), we can take K_1 solutions for (2.1) for $\mathfrak{G}$ with the learning rate η.

3.4 Algorithms for Prediction of Packs

In this section, we apply the theory we have developed to obtain prediction algorithms. This can be done in two essentially different ways leading to different types of bounds. In Section 3.4.1 we introduce AAP-max and AAP-incremental, and in Section 3.4.2 we introduce AAP-current.

3.4.1 Prediction with Plain Bounds

Consider a game $\mathfrak{G} = \{\Omega, \Gamma, \lambda\}$. The *Aggregating Algorithm for Packs with the Known Maximum* (AAP-max) and the *Aggregating Algorithm for Packs with an Unknown Maximum* (AAP-incremental) take as parameters a prior distribution $p(1), p(2), \dots, p(N)$ (such that $p(i) \ge 0$ and $\sum_{i=1}^N p(i) = 1$), a learning rate $\eta > 0$ and a constant C admissible for η. AAP-max also takes a constant $K > 0$. The intuitive meaning is that K is an upper bound on pack sizes, $K_t \le K$.

The algorithms follow very similar protocols and we will describe them in parallel. The algorithm AAP-max works as follows.

Protocol 6 (AAP-max)**.**

initialise losses $L_0^i = 0$, $i = 1, 2, \dots, N$
this step is skipped
set weights to $w_0(i) = p(i)$, $i = 1, 2, \dots, N$
FOR $t = 1, 2, \dots$
    normalise the weights $p_{t-1}(i) = w_{t-1}(i) / \sum_{i=1}^N w_{t-1}(i)$
    FOR $k = 1, 2, \dots, K_t$
        read the experts' predictions $\xi_{t,k}(i)$, $i = 1, 2, \dots, N$
        output $\gamma_{t,k} \in \Gamma$ satisfying for all $y \in \Omega$ the

inequality $\lambda(y, \gamma_{t,k}) \le -\frac{C}{\eta} \ln \sum_{i=1}^{N} p_{t-1}(i) e^{-\eta\lambda(y,\xi_{t,k}(i))}$

9 ENDFOR

10 observe the outcomes $y_{t,k}$, $k = 1, 2, \ldots, K_t$

11 update the losses $L_t^i = L_{t-1}^i + \sum_{k=1}^{K_t} \lambda(y_{t,k}, \xi_{t,k}(i))$, $i = 1, 2, \ldots, N$

12 let $K_{\max}^t = K$

13 update the experts' weights $w_t(i) = p(i) e^{-\eta L_t^i / K_{\max}^{t+1}}$, $i = 1, 2, \ldots, N$

14 END FOR

The algorithm AAP-incremental follows a protocol that is the same except for the following lines:

Protocol 7 (AAP-incremental)**.**

2 initialise $K_{\max}^0 = 1$

12 update $K_{\max}^t = \max(K_{\max}^{t-1}, K_t)$

As AAP-max always uses the same K for calculating the weights, line 13 can be replaced with an equivalent

$$w_t(i) = w_{t-1}(i) e^{-\eta \sum_{k=1}^{K_t} \lambda(y_{t,k}, \xi_{t,k}(i))/K}$$

and losses do not need to be maintained explicitly.

If C is admissible for $\mathfrak{G}$ with the learning rate η and $p_{t-1}(i)$, $i = 1, 2, \ldots, N$, the step on line 8 can always be performed and the (plain) cumulative losses (3.2) and (3.3) satisfy the following inequalities.

Theorem 3.4.1. *Let C be admissible for $\mathfrak{G}$ with the learning rate η. Then*

1. *The learner following AAP-max suffers loss satisfying*

$$L_T \le C L_T^i + \frac{KC}{\eta} \ln \frac{1}{p(i)}$$

for all outcomes and experts' predictions as long as the pack size does not exceed K, i.e., $K_t \le K$, $t = 1, 2, \ldots, T$.

2. *The learner following AAP-incremental suffers loss satisfying*

$$L_T \le C L_T^i + \frac{KC}{\eta} \ln \frac{1}{p(i)} \ ,$$

where K is the maximum pack size over T trials, $K = \max_{t=1,2,\dots,T+1} K_t$, for all outcomes and experts' predictions.

The theorem provides an alternative of Lemma 2.2.3 for the prediction of packs. The bounds are similar to the bound for the AA except that the regret term is multiplied by the maximum pack size K. For mixable games the theorem states that for a finite number of experts the AAP-max and AAP-incremental predict as well as the best expert up to an additive constant.

Proof. The proof essentially repeats that of Proposition 2.10. By induction one can show that

$$e^{-\eta L_T/(CK^T_{\max})} \geq \sum_{i=1}^{N} p(i) e^{-\eta L^i_T/K^T_{\max}} \quad . \tag{3.5}$$

Line 8 of Protocols 6 and 7 ensure that inequality (3.5) holds for $T = 1$.

Assume that (3.5) holds. We first raise inequality (3.5) to the power $K^T_{\max}/K^{T+1}_{\max} \leq 1$ and apply Jensen's inequality:

$$e^{-\eta L_T/(CK^{T+1}_{\max})} \geq \left(\sum_{i=1}^{N} p(i) e^{-\eta L^i_T/K^T_{\max}} \right)^{K^T_{\max}/K^{T+1}_{\max}} \geq \sum_{i=1}^{N} p(i) e^{-\eta L^i_T/K^{T+1}_{\max}} \quad . \tag{3.6}$$

According to line 8 of Protocols 6 and 7 we have that

$$e^{-\frac{\eta}{K^{T+1}_{\max}} \sum_{k=1}^{K_{T+1}} \lambda(y_{T+1,k},\gamma_{T+1,k})/C} \geq \sum_{i=1}^{N} p_T(i) e^{-\frac{\eta}{K^{T+1}_{\max}} \sum_{k=1}^{K_{T+1}} \lambda(y_{T+1,k},\xi_{T+1,k}(i))}$$
$$= \frac{\sum_{i=1}^{N} p(i) e^{-\eta L^i_{T+1}/K^{T+1}_{\max}}}{\sum_{i=1}^{N} p(i) e^{-\eta L^i_T/K^{T+1}_{\max}}}. \tag{3.7}$$

By multiplying inequalities (3.6) and (3.7), we get (3.5).

To complete the proof, it remains to drop all terms from the sum in inequality (3.5) except for one. □

3.4.2 Prediction with Bounds on Pack Averages

The *Aggregating Algorithm for Pack Averages* (AAP-current) takes as parameters a prior distribution $p(1), p(2), \dots, p(N)$ (such that $p(i) \geq 0$ and $\sum_{i=1}^{N} p(i) = 1$), a learning rate $\eta > 0$ and a constant C admissible for η.

Protocol 8 (AAP-current)**.**

`1 initialise weights` $w_0(i) = p(i)$, $i = 1, 2, \dots, N$

2 FOR $t = 1, 2, \ldots$
3 normalise the weights $p_{t-1}(i) = w_{t-1}(i) / \sum_{i=1}^{N} w_{t-1}(i)$
4 FOR $k = 1, 2, \ldots, K_t$
5 read the experts' predictions $\xi_{t,k}(i)$, $i = 1, 2, \ldots, N$
6 output $\gamma_{t,k} \in \Gamma$ satisfying for all $y \in \Omega$ the inequality $\lambda(y, \gamma_{t,k}) \leq -\frac{C}{\eta} \ln \sum_{i=1}^{N} p_{t-1}(i) e^{-\eta \lambda(y, \xi_{t,k}(i))}$
7 ENDFOR
8 observe the outcomes $y_{t,k}$, $k = 1, 2, \ldots, K_t$
9 update the experts' weights $w_t(i) = w_{t-1}(i) e^{-\eta \sum_{k=1}^{K_t} \lambda(y_{t,k}, \xi_{t,k}(i)) / K_t}$, $i = 1, 2, \ldots, N$
10 END FOR

In line 9 we divide by the size of the current pack.

Defining *cumulative average loss* of a strategy $\mathcal{S}$ and experts $\mathcal{E}_i$ working in the environment specified by Protocol 4 as

$$L_{\text{average},T} = \sum_{t=1}^{T} \frac{\sum_{k=1}^{K_t} \lambda(y_{t,k}, \gamma_{t,k})}{K_t} \quad ,$$

$$L^i_{\text{average},T} = \sum_{t=1}^{T} \frac{\sum_{k=1}^{K_t} \lambda(y_{t,k}, \xi_{t,k}(i))}{K_t} \quad , i = 1, 2, \ldots, N,$$

we get the following theorem.

Theorem 3.4.2. *If C is admissible for $\mathfrak{G}$ with the learning rate η, then the learner following AAP-current suffers loss satisfying*

$$L_{\text{average},T} \leq C L^i_{\text{average},T} + \frac{C}{\eta} \ln \frac{1}{p(i)} \tag{3.8}$$

for all outcomes and experts' predictions.

The bound (3.8) coincides with the bound for the AA in Lemma 2.2.3. However, note that the bound for the AAP-current is provided on the cumulative average loss instead of the cumulative loss. For mixable games the theorem states that for a finite number of experts the AAP-current predicts as well as the best expert up to an additive constant in terms of the cumulative average loss.

Proof. We again prove by induction that

$$e^{-\eta L_{\text{average},T}/C} \geq \sum_{i=1}^{N} p(i) e^{-\eta L^i_{\text{average},T}} \quad . \tag{3.9}$$

The step on line 6 of Protocol 8 ensures that inequality (3.9) holds for $T = 1$. For time $T + 1$ we have that

$$e^{-\frac{\eta}{K_{T+1}}\sum_{k=1}^{K_{T+1}}\lambda(y_{T+1,k},\gamma_{T+1,k})/C} \geq \sum_{i=1}^{N} p_T(i) e^{-\frac{\eta}{K_{T+1}}\sum_{k=1}^{K_{T+1}}\lambda(y_{T+1,k},\xi_{T+1,k}(i))}$$
$$= \frac{\sum_{i=1}^{N} p(i) e^{-\eta L^i_{\text{average},T+1}}}{\sum_{i=1}^{N} p(i) e^{-\eta L^i_{\text{average},T}}} \quad . \tag{3.10}$$

Assume that (3.9) holds. The induction step is by multiplying inequalities (3.9) and (3.10). □

3.5 Discussion and Optimality

The loss bounds from Theorem 3.4.1 do not improve on inequality (3.4), which holds for Parallel Copies (see Section 3.2.2 for details). However, the performance of AAP-max and AAP-incremental does not depend on the order of outcomes in packs. In Section 3.6.2 we describe numerical experiments comparing AAP-max, AAP-incremental, and AAP-current against the loss of Parallel Copies averaged over permutations within packs.

If all K_t are equal, $K_1 = K_2 = \ldots = K_T = K$, the algorithms AAP-max and AAP-incremental are identical and equivalent to applying the Aggregating Algorithm with the learning rate η/K to the game $\mathfrak{G}^K$. Under the conditions of Theorem 3.3.3, the optimality property of the Aggregating Algorithm proven by Vovk (1998). Thus the constants in the bounds of Theorem 3.4.1 cannot be improved without the loss of generality. However, if the pack size varies, AAP-max clearly uses a suboptimal learning rate η/K where η/K_t is needed. AAP-incremental does that if the pack size decreases. We compare AAP-incremental and AAP-max experimentally in Section 3.6.2.

The bound of Theorem 3.4.2 is, to our knowledge, novel and cannot be straightforwardly obtained using a parallel copies-type merging strategy. If the pack size is the same, the bound is optimal (and identical to those from Theorem 3.4.1). If the pack size varies, AAP-current always uses the optimal learning rate. However, technically it is not covered by the optimality results of Vovk (1998) as the game changes from step to step. We leave this as an open problem.

The bound of Theorem 3.4.2 involves cumulative average loss and does not imply good bounds for plain cumulative loss straightforwardly. If $K_{\min} \leq K_1, K_2, \ldots, K_T \leq$

$K_{\max}$, then $L_{\text{average},T} \geq L_T/K_{\max}$ and $L^i_{\text{average},T} \leq L^i_T/K_{\min}$. We get the bound

$$L_T \leq \frac{K_{\max}}{K_{\min}} C L^i_T + \frac{C K_{\max}}{\eta} \ln \frac{1}{p(i)} \tag{3.11}$$

for the cumulative loss of AAP-current, which appears inferior to those from Theorem 3.4.1. However, in experiments AAP-current shows good performance even in terms of the plain cumulative loss; see Section 3.6.2. Bound (3.11), loose it may be, provides an explanation to some phenomena we observe in Section 3.6.2.

3.5.1 A Mix Loss Lower Bound

In this section, we present a self-contained lower bound formulated for the mix loss protocol of Adamskiy et al. (2016). The proof sheds some further light on the extra term in the bound.

The mix loss protocol covers a number of learning settings including prediction with a mixable loss function (Adamskiy et al., 2016, Section 2). Consider the following protocol porting mix loss Protocol 1 of Adamskiy et al. (2016) to prediction of packs.

Protocol 9 (Mix loss)**.**

FOR $t = 1, 2, \ldots$
  nature announces K_t
  learner outputs K_t arrays of N probabilities
    $p_{t,k}(1), p_{t,k}(2), \ldots, p_{t,k}(N)$, $k = 1, 2, \ldots, K_t$, such that
    $p_{t,k}(i) \in [0,1]$ for all i and k and $\sum_{i=1}^N p_{t,k}(i) = 1$ for all k
  nature announces losses $\ell_{t,1}(i), \ell_{t,2}(i), \ldots, \ell_{t,K_t}(i) \in (-\infty, +\infty]$
  learner suffers loss $\ell_t = -\sum_{k=1}^{K_t} \ln \sum_{i=1}^N p_{t,k}(i) e^{-\ell_{t,k}(i)}$
ENDFOR

The total loss of the learner over T steps is $L_T = \sum_{t=1}^T \ell_t$. It should compare well against $L^i_T = \sum_{t=1}^T \ell_t(i)$, where $\ell_t(i) = \sum_{k=1}^{K_t} \ell_{t,k}(i)$. The values of L^i_T are the counterparts of experts' total losses. We shall propose a course of action for the nature leading to a high value of the regret $L_T - \min_{i=1,2,\ldots,N} L^i_T$.

Lemma 3.5.1. *For any K arrays of N probabilities $p_k(1), p_k(2), \ldots, p_k(N)$, $k = 1, 2, \ldots, K$ (where $p_k(i) \in [0,1]$ for all $i = 1, 2, \ldots, N$ and $k = 1, 2, \ldots, K$, and $\sum_{i=1}^N p_k(i) = 1$ for all k), there is i_0 such that*

$$\prod_{k=1}^K p_k(i_0) \leq \frac{1}{N^K} \ .$$

Proof. Assume this is not the case. Let $\prod_{k=1}^{K} p_k(i) > 1/N^K$ for all i. By the inequality of arithmetic and geometric means

$$\sum_{k=1}^{K} \frac{p_k(i)}{K} \geq \left(\prod_{k=1}^{K} p_k(i) \right)^{\frac{1}{K}}$$

for all $i = 1, 2, \ldots, N$. Summing the left-hand side over i we get

$$\sum_{i=1}^{N} \sum_{k=1}^{K} \frac{p_k(i)}{K} = \frac{1}{K} \sum_{k=1}^{K} \sum_{i=1}^{N} p_k(i) = 1 \ .$$

Summing the right-hand side over i and using the assumption on the products of $p_k(i)$, we get

$$\sum_{i=1}^{N} \left(\prod_{k=1}^{K} p_k(i) \right)^{\frac{1}{K}} > \sum_{i=1}^{N} \left(\frac{1}{N^K} \right)^{\frac{1}{K}} = \sum_{i=1}^{N} \frac{1}{N} = 1 \ .$$

The contradiction proves the lemma. □

Theorem 3.5.2. *Over a single pack of size K the regret R of the strategy with a mix loss should be lower-bounded by $K \ln N$.*

Proof. Here is the strategy for the nature. Upon getting the probability distributions from the learner, it finds i_0 such that $\prod_{k=1}^{K_t} p_{t,k}(i_0) \leq 1/N^{K_t}$ and sets $\ell_{t,1}(i_0) = \ell_{t,2}(i_0) = \ldots = \ell_{t,K_t}(i_0) = 0$ and $\ell_{t,k}(i) = +\infty$ for all other i and $k = 1, 2, \ldots, K_t$. The learner suffers loss

$$\ell_t = -\sum_{k=1}^{K_t} \ln p_{t,k}(i_0) = -\ln \prod_{k=1}^{K_t} p_{t,k}(i_0) \geq -\ln \frac{1}{N^{K_t}} = K_t \ln N$$

while $\ell_t(i_0) = 0$. We see that over a single pack of size K we can achieve the regret of $K_t \ln N$. Thus every upper bound of the form $L_t \leq L_t^i + R_t$ should have $R_t \geq K_t \ln N$, where K_t is the size of the t-th pack. □

3.6 Experiments

In this section, we present some empirical results[1]. We want to compare the behaviour of the AAP family algorithms against each other and against the Parallel Copies algorithm of Section 3.2.2. Sections 3.6.3 and 3.6.4 investigate related questions concerned with the power of online learning.

[1]The code written in R is available at https://github.com/RaisaDZ/AAP-.

3.6.1 Datasets and Experts

In our experiments we use two sports datasets and two datasets of house prices.

The idea of using odds output by bookmakers for testing prediction with expert advice algorithms goes back to Vovk and Zhdanov (2009). The bookmakers are treated as black boxes; we take the odds they quote from publicly available sources and do not look into techniques they use to work out the odds. This fits perfectly with the methodology of prediction with expert advice.

There is a tradition of using house prices as a benchmark for machine learning algorithms going back to the Boston housing dataset. However, batch learning protocols have hitherto been used in most studies. Recently extensive datasets with timestamps have become available. They call for online learning protocols. Property prices are prone to strong movements over time and the pattern of change may be complicated. Online algorithms should capture these patterns.

We train learning algorithms (regression and trees) on housing data and then use methods of prediction with expert advice to merge their predictions.

Sports datasets

In order to establish continuity with the existing empirical work, we use the tennis dataset[2] studied by Vovk and Zhdanov (2009). It contains historical information about tennis tournaments from 2004, 2005, 2006, and 2007, including Australian Open, French Open, US Open, and Wimbledon. The outcomes in the dataset are results of tennis matches coded as 0 or 1 according to which side wins (there can be no draws). The total number of outcomes is 10087. A prediction is $\gamma \in [0,1]$, which can be understood as the probability of the outcome 1. We use the quadratic loss function $\lambda(y,\gamma) = (\gamma - y)^2$. This falls under the definition of the general square-loss game described in Section 2.2.2. Note that the loss function used in this chapter equals one half of the one used by Vovk and Zhdanov (2009); we make this choice for consistency with regression experiments.

Four bookmakers are taken as experts, Bet365, Centrebet, Expekt, and Pinnacle Sports. What bookmakers output is odds and we need probabilities for the experiments. Vovk and Zhdanov (2009) give two methods for calculating the probabilities. For this dataset Khutsishvili's method (Vovk and Zhdanov, 2009, Section 3) was used.

The dataset does not contain packs, so we introduced them artificially. We did this in two ways. First, we grouped adjacent matches into packs of random size from 1 to 12. We refer to the resulting dataset as tennis with small packs. Secondly, we grouped adjacent matched into packs of random size from 5 to 16 and thus constructed the

[2] Available at http://vovk.net/ICML2008/.

tennis with large packs dataset. (The sizes were independently drawn from respective uniform distributions.)

The second sports dataset was compiled by us from historical information[3] on football matches and bookmakers' odds. The dataset covers four seasons, 2013/2014, 2014/2015, 2015/2016, and 2016/2017 of the English Premier League and totals 1520 matches. Each match can have three outcomes, 'home win', 'draw', or 'away win', interpreted as three unit vectors $(1,0,0)$, $(0,1,0)$, and $(0,0,1)$. A prediction is a vector $\gamma = (p_1, p_2, p_3)$ from the simplex, i.e., $p_i \geq 0$ and $p_1 + p_2 + p_3 = 1$ and the loss is the quadratic norm of the difference, $\lambda(y, \gamma) = \|\gamma - y\|_2^2$. This is a case of the multidimensional Brier game (Vovk and Zhdanov, 2009). The game is mixable and the maximum learning rate such that $C_\eta = 1$ is $\eta_0 = 1$; the substitution function is provided by (Vovk and Zhdanov, 2009, Proposition 2).

We recalculated experts' prediction probabilities from bookmakers' odds using the simpler method described by (Vovk and Zhdanov, 2009, Appendix B) for speed. We took Bet365, Bet&Win, Interwetten, Ladbrokes, Sportingbet, Will Hill, Stan James, VC Bet, and BetBrain.

The dataset is naturally organised in packs: from 1 to 10 matches occur on one day. We treat matches from the same day as a pack.

Ames House Prices

The Ames dataset describes the property sales that occurred in Ames, Iowa between 2006 and 2010. The dataset contains records of 2930 house sales transactions with 80 attributes, which are a mixture of nominal, ordinal, continuous, and discrete parameters (including physical property measurements) affecting the property value. The dataset was compiled by De Cock (2011) for use in statistics education as a modern substitute for the Boston Housing dataset. For the outcome we take the raw sales prices or their logarithms; these make two sets of experiments. We try to predict the outcomes measuring the deviation by the squared difference. This again falls under the definition of the general square-loss game of Section 2.2.2. The bounds A and B are taken from the first year of data, which is used for training.

There are timestamps in the dataset, but they contain only the month and the year of the purchase. The date is not available. We treat one month of transactions as a pack and interpret the problem as an online one falling under Protocol 4.

We create two pools of experts for experiments. In the first pool, our experts are linear regression models based on only two attributes: the neighbourhood and the total square footage of the dwelling. These simple models explain around 80% of the

[3] Available at http://football-Data.co.uk

variation in sales prices and they are very easy to train. Each expert has been trained on one month from the first year of the data. Hence there are 12 'monthly' experts. For the second pool we use random forests (RF) models after Bellotti (2017). 'Random forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest' (Breiman (2001)). A model was built for each quarter of the first year. Hence there are four 'quarterly' experts. They take longer to train but produce better results. Note that 'monthly' RF experts were not practical; training a tree requires a lot of data and 'monthly' experts returned very poor results. We apply the experts to predict the prices starting from year two.

London House Prices

Another dataset we used contains house prices in and around London over the period 2009 to 2014. This dataset was made publicly available by the Land Registry[4] in the UK and was originally sourced as part of a Kaggle competition. The Property Price data consists of details for property sales and contains around 1.38 million observations. This dataset was studied by Bellotti (2017) to provide reliable region predictions for Automated Valuation Models of house prices. Again, we try to predict sales prices and their logarithms.

As with the Ames dataset, we use linear regression models that were built for each month of the first year of the data as experts of AAP. The features that were used in regression models contain information about the property: property type, whether new build, whether free- or leasehold. Along with the information about the proximity to tube and railway stations, our models use the English indices of deprivation from 2010[5], which measures relative levels of deprivation. The following deprivation scores were used in models: income, employment, health and disability, education for children and skills for adults, barriers to housing and services with sub-domains wider barriers and geographical barriers, crime, living environment score with sub-domains for indoor and outdoor living (i.e., quality of housing and external environment, respectively). In addition to the general income score, separate scores for income deprivation affecting children and the older population were used.

In the second set of experiments on London house price dataset, we use RF models built for each month of the first year as experts. Unlike the Ames dataset, London

[4]See HM Land Registry Monthly Property Transaction Data on http://data.gov.uk, https://data.gov.uk/dataset/7d866093-2af5-4076-896a-2d19ca2708bb/hm-land-registry-monthly-property-transaction-data

[5]See https://www.gov.uk/government/statistics/english-indices-of-deprivation- 2010

dataset contains enough observations to train RF models on one month of data. Hence we get 12 'monthly' experts.

3.6.2 Comparison of Merging Algorithms

Comparison of AAP with Parallel Copies of AA

We start by comparing the family of AAP merging algorithms against parallel copies of AA. While for AAP algorithms the order of examples in the pack makes no difference, for Parallel Copies it is important. To analyse the dependency on the order we ran Parallel Copies 500 times randomly shuffling each pack each time. The experiments were only carried out on sports and Ames data, as on London data they would take too long to complete. Figures 3.1 and 3.2 show histograms of total losses of Parallel Copies, total losses of AAP family algorithms, and the total loss of Parallel Copies with one particular order, as in the database.

We see that while the performance of Parallel Copies *can* be better for particular orderings, order-independent performance loss of AAP family algorithms is always close to the average loss of Parallel Copies and some algorithms from the family beat it. AAP-current is always better than the average. In experiments with Ames data and tennis data with large packs AAP-current is the best while AAP-incremental is the best on tennis data with small packs and football data.

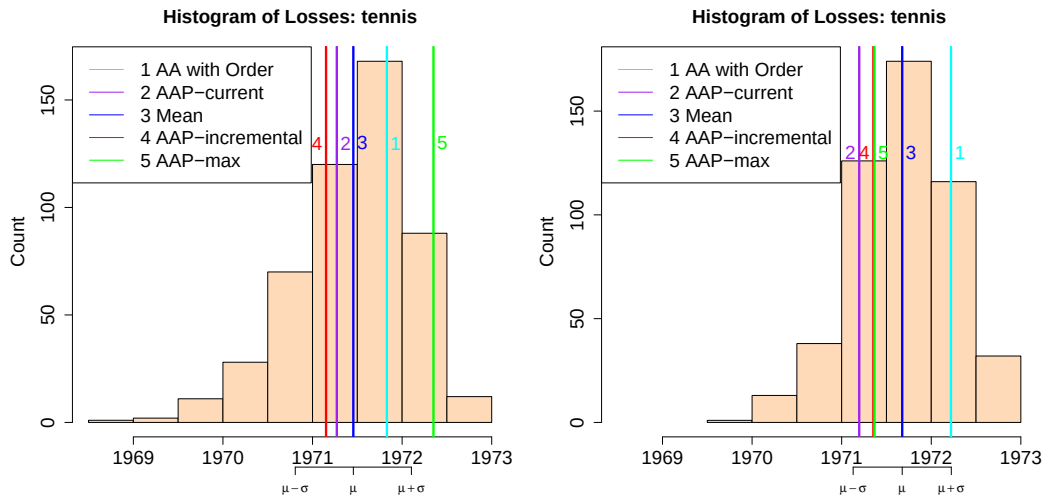
There is one remarkable ordering where Parallel Copies show greatly superior performance on Ames data. If packs are ordered by PID (i.e., as in the database), Parallel Copies suffer substantially lower loss. PID (Parcel identification ID) is assigned to each property by the tax assessor. It is related to the geographical location. When the packs are ordered by PID, Parallel Copies benefit from geographical proximity of the houses; each copy happens to get similar houses.

This effect is not observed on sports datasets as the order in the dataset does not convey any particular information.

Comparison of AAP-incremental and AAP-max

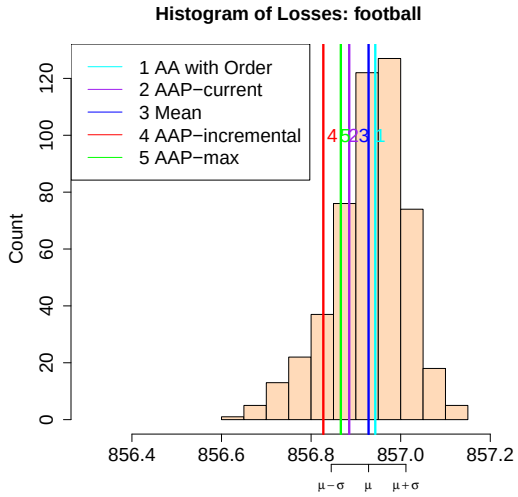
As one can see from Figures 3.1 and 3.2, AAP-max is usually the worst among the AAP bunch. In this section, we check this by comparing AAP-max against AAP-incremental. Here AAP-max receives the maximum pack size calculated retrospectively from the start and AAP-incremental uses the current maximum.

For a detailed comparison of two online learning algorithms, $\mathcal{S}_1$ and $\mathcal{S}_2$, it is not enough to consider the two values of their total losses. We need to see how these losses were accumulated. So, following Vovk and Zhdanov (2009) and Kalnishkan et al.



(a) Tennis data, small packs

(b) Tennis data, large packs

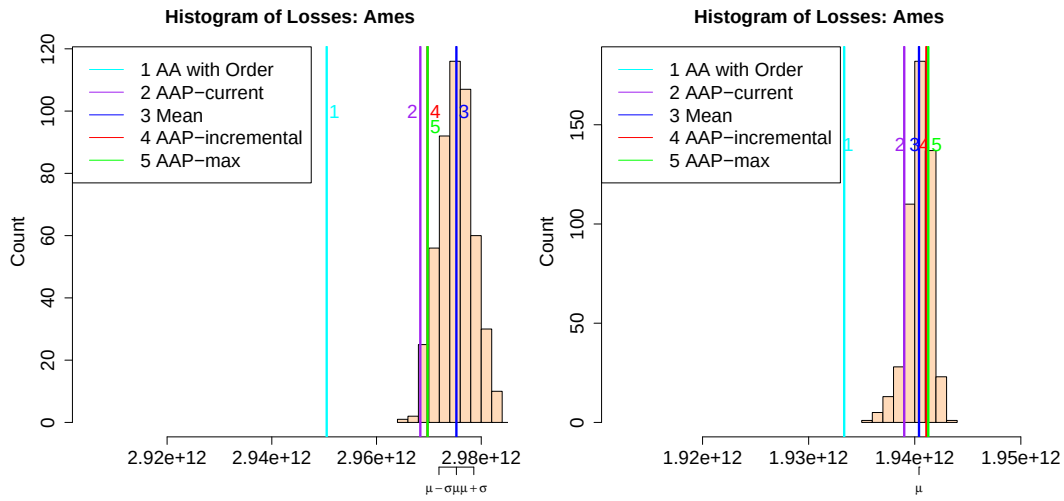


(c) Football data

Figure 3.1: Histogram of total losses of Parallel Copies with total losses of AAP algorithms on sports datasets

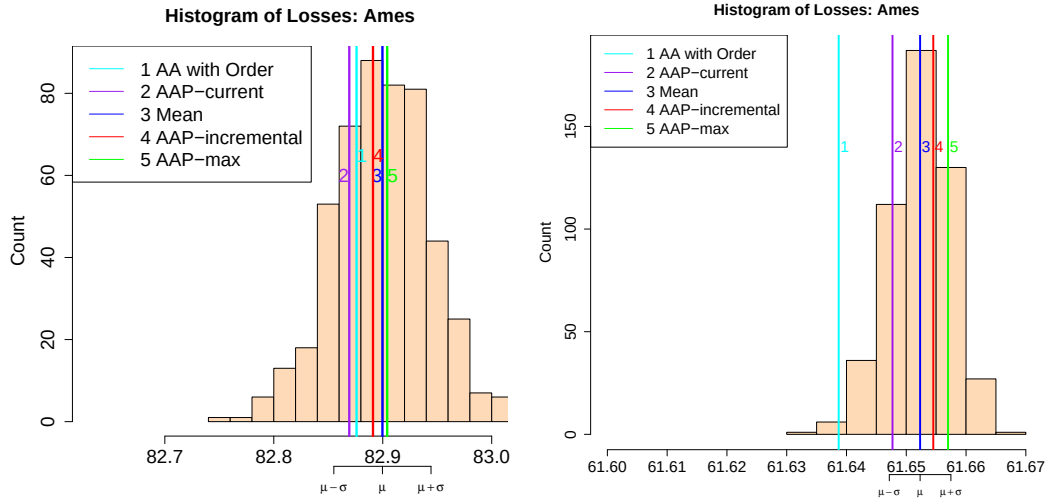
(2015), we plot the difference of their cumulative losses vs time. If the difference steadily decreases, then $\mathcal{S}_1$ consistently outperforms $\mathcal{S}_2$.

Figures 3.3, 3.4, and 3.5 plot the differences in total losses of AAP-incremental and AAP-max on sports datasets, house prices, and logarithms of house prices, respectively. Over the graphs of the difference of losses, the values of $K^t_{\max} = \max_{s=1,2,\dots,t} K^s_{\max}$, the current maximum pack size, are superimposed.



(a) Regression on Ames house prices

(b) RF on Ames house prices

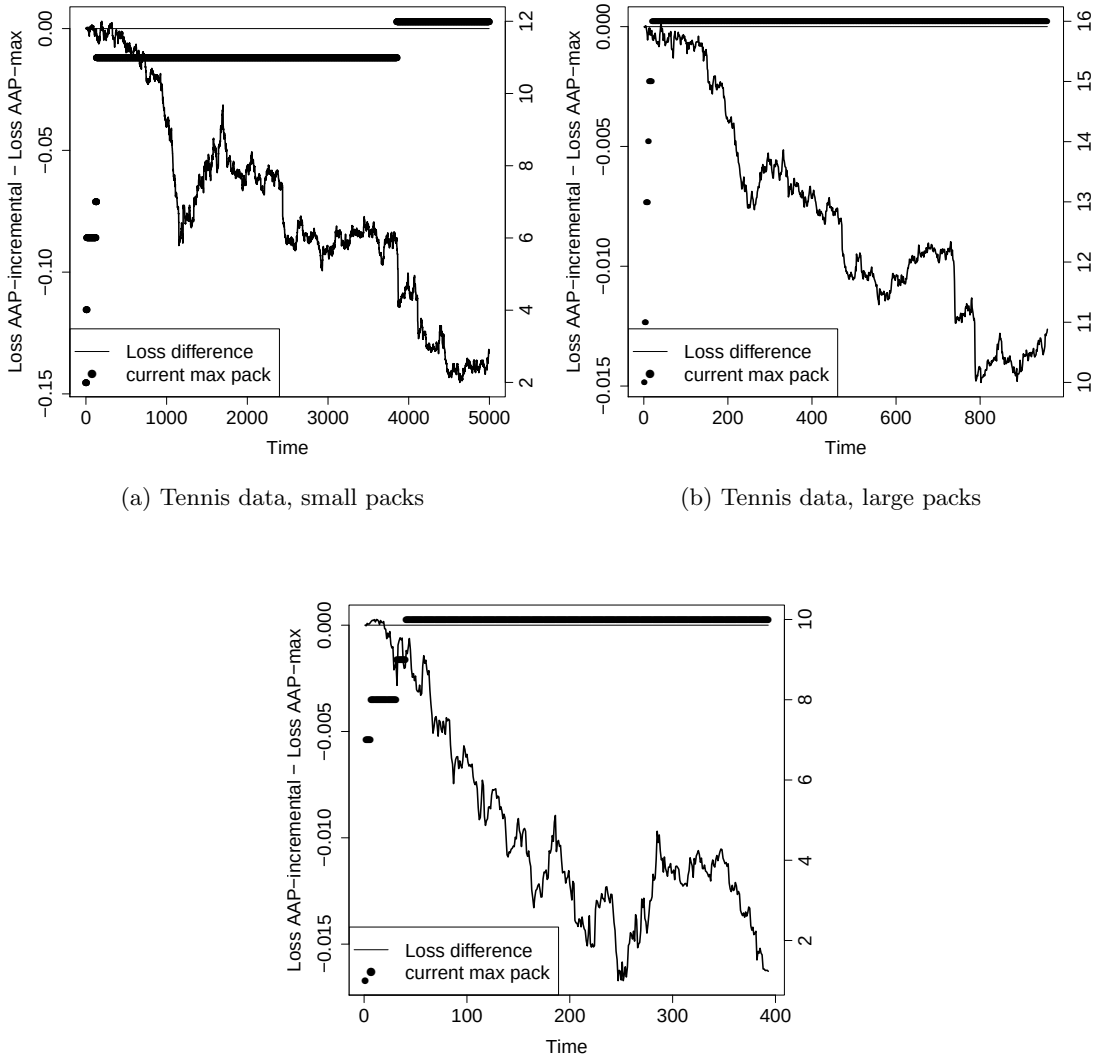


(c) Regression on Ames house log-prices

(d) RF on of Ames house log-prices

Figure 3.2: Histogram of total losses of Parallel Copies with total losses of AAP algorithms on house price datasets

We see that AAP-incremental generally performs better at the beginning of the period when the current maximum size of the pack is much lower than the maximum pack of the whole period. The difference of the losses then goes down in the figure. As the current maximum reaches the overall maximum, the difference of losses may level out or even go up sometimes. This means that the performance of AAP-incremental is no longer superior to the performance of AAP-max.



(a) Tennis data, small packs

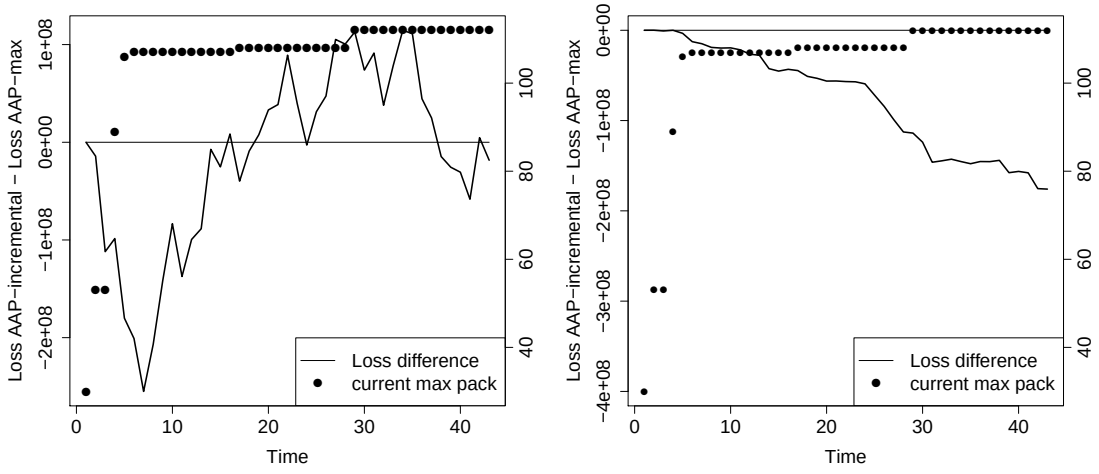
(b) Tennis data, large packs

(c) Football data

Figure 3.3: Difference of cumulative losses of AAP-incremental and AAP-max vs time on sports data with cumulative maximum pack sizes superimposed

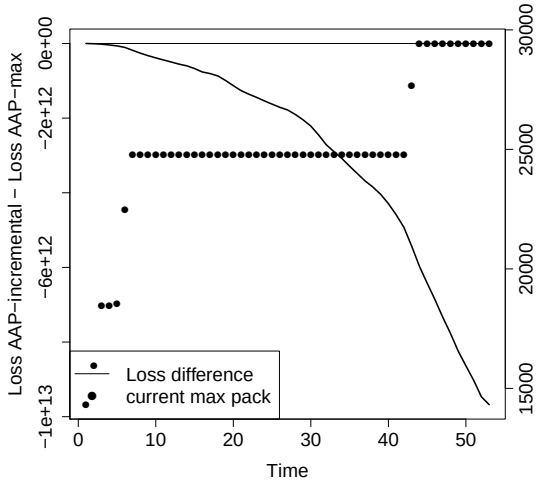
These observations are consistent with the discussion in Section 3.5: AAP-max uses a suboptimal learning rate before the maximum pack size is achieved.

On London house prices (and their logarithms), where the maximum pack size is achieved very late, AAP-incremental outperforms AAP-max in a steady fashion. After the maximum pack size has been reached, the effect lingers. A possible explanation is that AAP-incremental was learning from its feedback in a more effective way through-

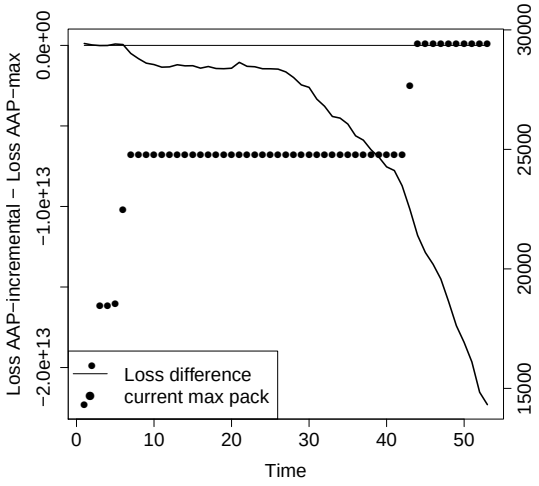


(a) Regression on Ames house prices

(b) RF on Ames house prices



(c) Regression on London house prices



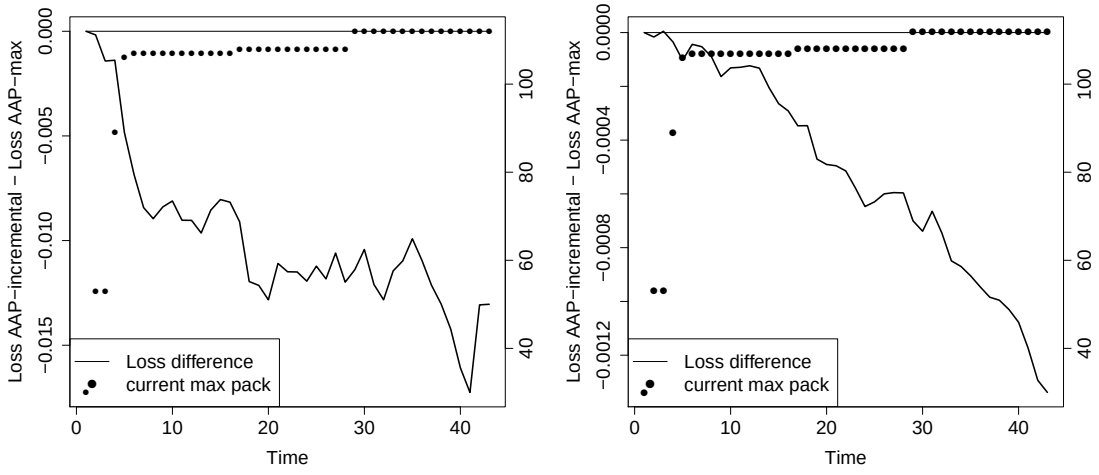
(d) RF on London house prices

Figure 3.4: Difference of cumulative losses of AAP-incremental and AAP-max vs time on house price datasets with cumulative maximum pack sizes superimposed

out the most of the dataset.

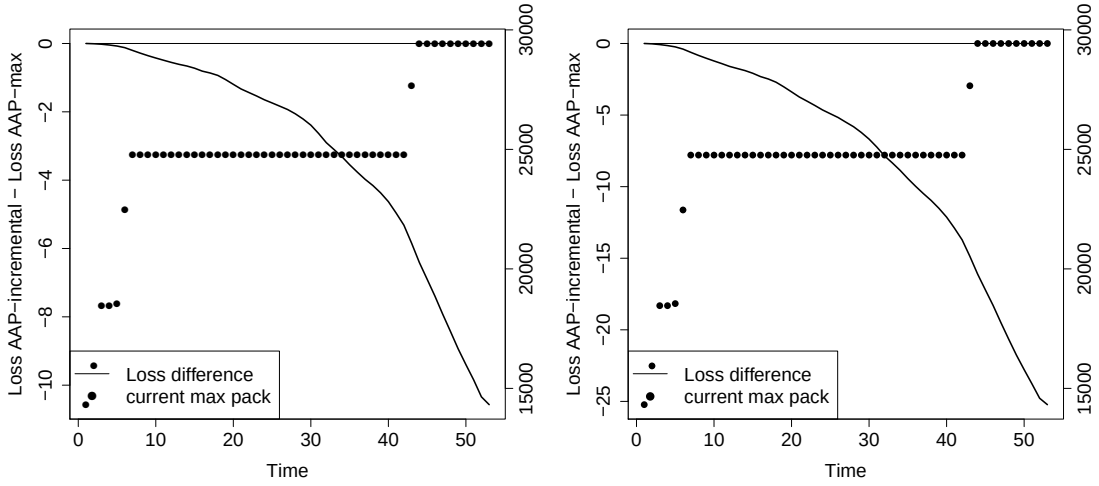
Comparison of AAP-current and AAP-incremental

The comparison of AAP-current and AAP-incremental provides a more challenging problem: sometimes one performs better and sometimes the other. Recall that we assess AAP-current by the plain cumulative loss (3.2) for comparison purposes.



(a) Regression on Ames house prices

(b) RF on Ames house prices



(c) Regression on London house prices

(d) RF on London house prices

Figure 3.5: Difference of cumulative losses of AAP-incremental and AAP-max vs time on logarithms of house prices with cumulative maximum pack sizes superimposed

Figures 3.6, 3.7, and 3.8 show the difference in plain cumulative losses of AAP-current and AAP-incremental for sports dataset, house prices and logarithms of house prices, respectively.

We see that AAP-current outperforms AAP-incremental on house prices and tennis data with large packs. The performance of AAP-current is remarkable because by design it is not optimised to minimise the total loss; see the discussion in Section 3.5.

In a way, here we assess AAP-current with a measure it is not good at. Still, optimal decisions of AAP-current produce superior performance.

Poor performance of AAP-current on tennis data with small packs and football data calls for an explanation. We attempt to explain this using upper bound (3.11). By design, the two tennis datasets differ in the ratio of the maximum and the minimum pack size: for the dataset with small packs it is $12/1 = 12$ and for the dataset with large packs it is $16/5 = 3.2$ (note that the differences are the same).

For the football and housing datasets we do not control the ratio of the maximum and minimum pack sizes. For the football dataset, where AAP-current performs poorly, the ratio is $10/1 = 10$ and for the London house prices, where it performs well, the ratio is much less and equals $29431/8900 = 3.3$.

The Ames dataset apparently does not fit the pattern with a large ratio of $112/8 = 14$. However, one can see from the histogram shown on Figure 3.9 that packs of small size are relatively rare; if we ignore them, the ratio immediately goes down. The same argument does not apply to the football dataset with plenty of small packs.

3.6.3 Comparison of AAP with Batch Models

In this section, we compare AAP-current with two straightforward ways of prediction, which are essentially batch. One goal we have here is to do a sanity check and verify whether we are not studying properties of very bad algorithms. Secondly, we want to show that prediction with expert advice may yield better ways of handling the available historical information as suggested by Kalnishkan et al. (2015).

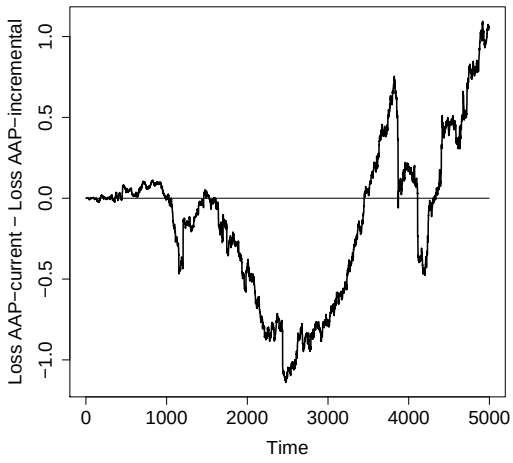
In AAP we use linear regression models that have been trained on each month of the first year of data. Is the performance of these models affected by straightforward seasonality? What if we always predict January with the January model, February with the February model etc.?

The first batch model we compare our online algorithm to is the seasonal model that predicts January with the linear regression model trained on January of the first year, February with the linear model trained on February of the first year, etc.

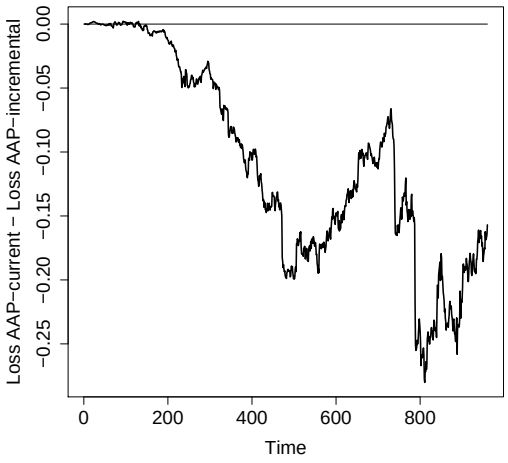
In the case of 'quarterly' RF experts, we compete with a seasonal model that predicts the first quarter with the RF model trained on the first quarter, second quarter with the RF model trained on the second quarter, etc.

Secondly, what if we train a model on the whole of the first year? This may be more expensive than training smaller models, but what do we gain in performance? The second batch model is the linear model trained on the whole first year of data. In case of RF experts, we compete with RF model trained on the first year of data.

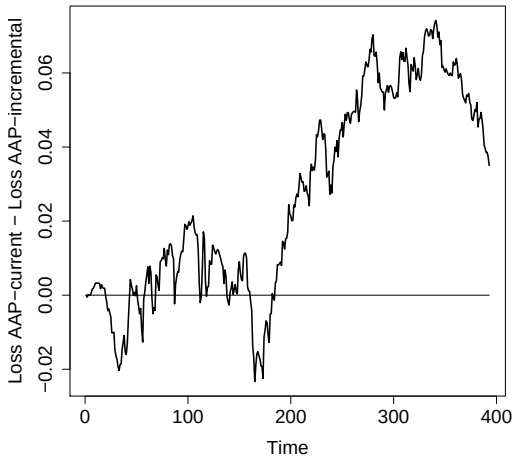
Figures 3.10, and 3.11 show the comparison of total losses of AAP-current and



(a) Tennis data, small packs



(b) Tennis data, large packs

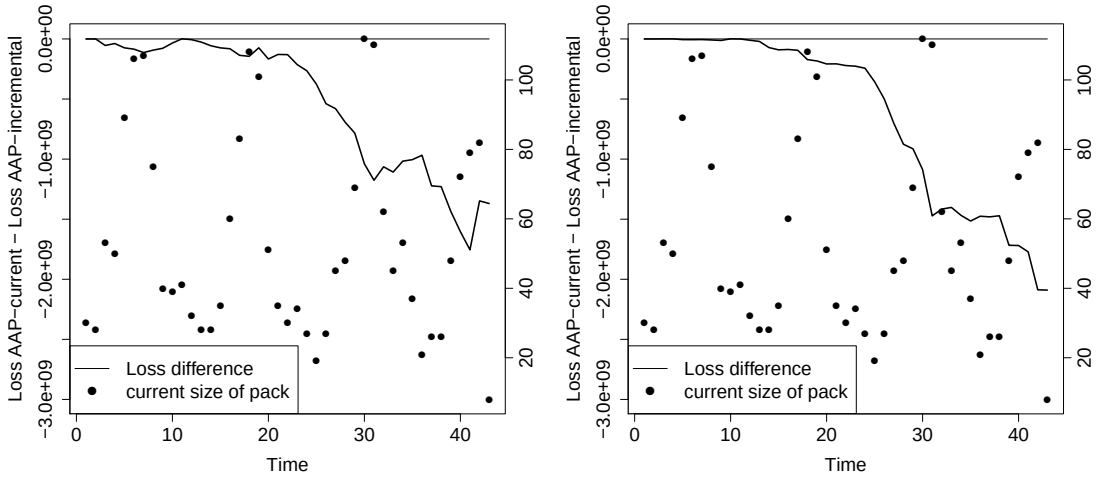


(c) Football data

Figure 3.6: Difference of cumulative losses of AAP-current and AAP-incremental vs time on sports data

batch linear regression models for Ames house dataset for prices and logarithmic prices respectively. AAP-current consistently performs better than the seasonal batch model. Thus the straightforward utilisation of seasonality does not help.

When compared to the linear regression model of the first year, AAP-current initially has higher losses but it becomes better towards the end. It could be explained as follows. AAP-current needs time to train until it becomes good in prediction. These



(a) Regression on Ames house prices

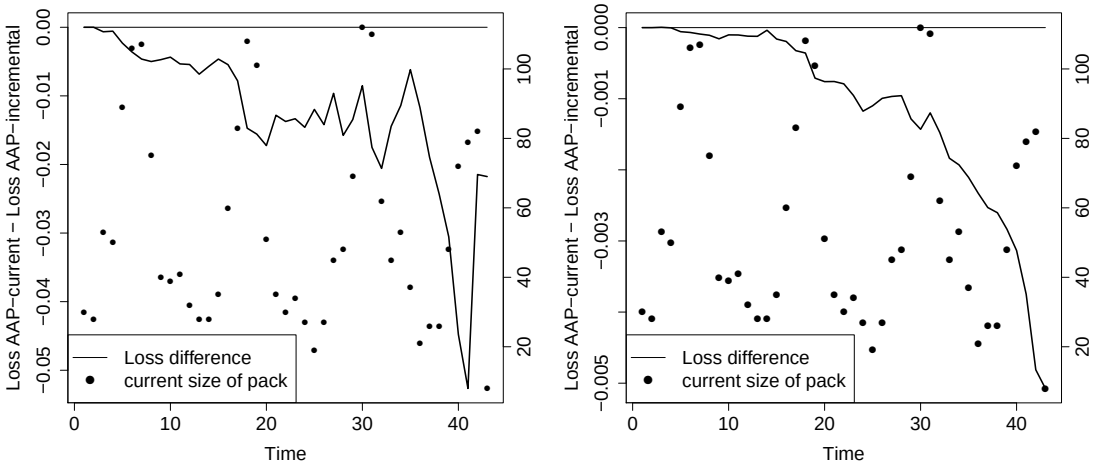
(b) RF on Ames house prices

(c) Regression on London house prices

(d) RF on London house prices

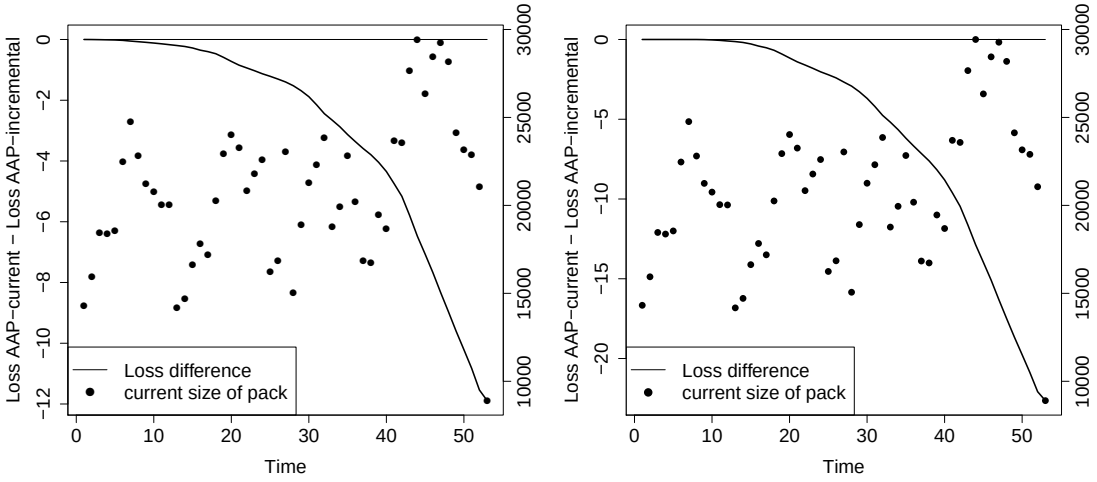
Figure 3.7: Difference of cumulative losses of AAP-current and AAP-incremental vs time on house price data with current pack sizes superimposed

results show that we can make a better use of the past data with prediction with expert advice than with models trained in the batch mode. However, these results do not hold for logarithmic prices where the linear regression model of the first year outperforms AAP-current almost on the whole period of the dataset.



(a) Regression on Ames house prices

(b) RF on Ames house prices



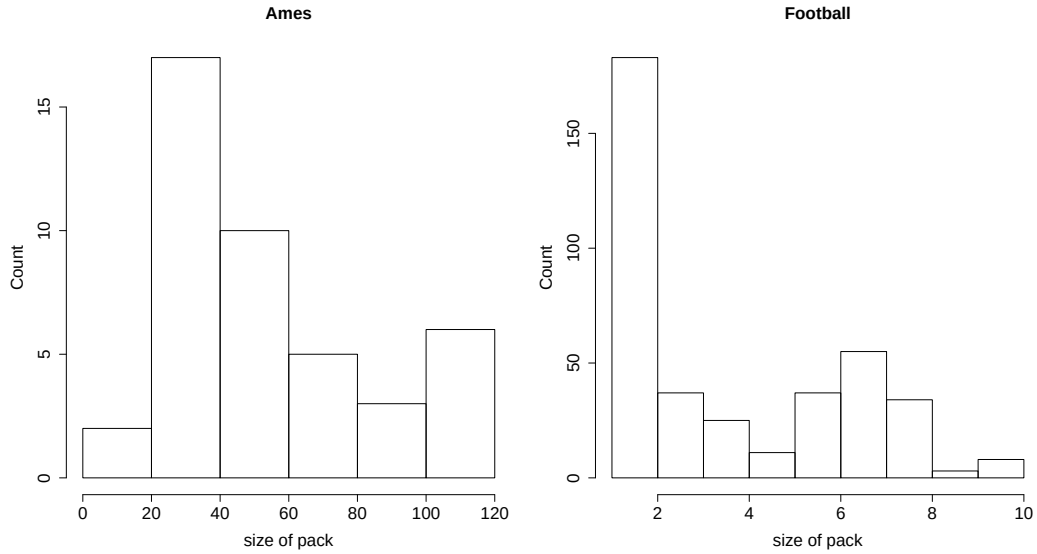
(c) Regression on London house prices

(d) RF on London house prices

Figure 3.8: Difference of cumulative losses of AAP-current and AAP-incremental vs time on logarithms of house price datasets with current pack sizes superimposed

3.6.4 Improving Predictions with Inflation Data

In the evolution of house prices a significant role is played by inflation. While on Ames data the overall trend is hardly visible, London house prices show a clear upward trend. One may wonder to what extent taking inflation information into account improves the quality of predictions and whether the effects we observed still stand if inflation is considered.



(a) Ames pack sizes

(b) Football pack sizes

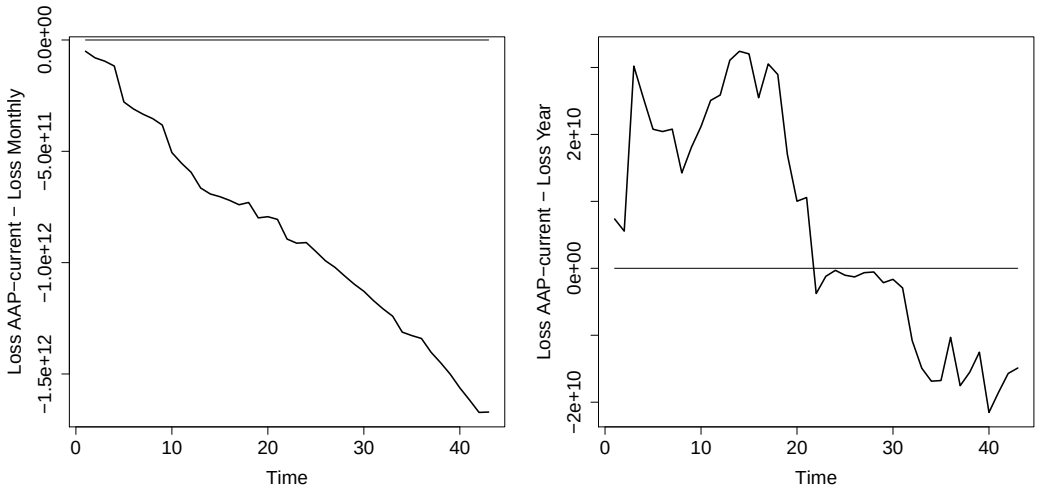
Figure 3.9: Histograms of pack sizes

We used Acadata House Price Index (HPI) data[6] to improve the quality of our prediction. Every expert was adjusted on the basis of inflation data. For every month passed since the expert had been trained, we added to the log price it predicted the value of $\ln(1+r)$, where r is the monthly index calculated by Acadata. (The index for the month when transactions occurred was not used; we assumed this information is only available afterwards.)

Figure 3.12 shows the comparison of cumulative losses of AAP-current with and without inflation. It is clear from the graph that taking inflation into account improves both linear regression and random forests experts. As original experts were built on the first year of the dataset, they consistently under-estimate house prices for more recent data.

Figure 3.13 illustrates the comparison of total losses of AAP-incremental and AAP-max on log prices with experts adjusted for inflation. Figure 3.14 illustrates the comparison of total losses of AAP-current and AAP-incremental. The patterns are similar to what we previously observed: AAP-current consistently outperforms AAP-incremental, whereas AAP-incremental is better than AAP-max on the whole period of data.

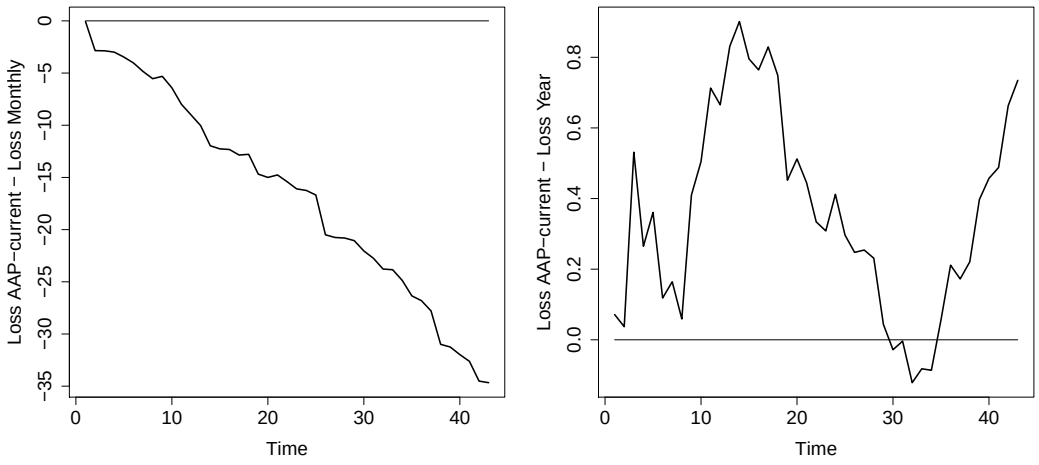
[6]Available at http://www.acadata.co.uk/acadataHousePrices.php.



(a) Loss difference of AAP-current and monthly batch

(b) Loss difference of AAP-current and year batch

Figure 3.10: Difference of cumulative losses of AAP and batch models vs time on house price data



(a) Loss difference of AAP-current and monthly batch

(b) Loss difference of AAP-current and year batch

Figure 3.11: Difference of cumulative losses of AAP and batch models vs time on log prices

3.6.5 Conclusions

This section summarises the conclusions from empirical experiments.

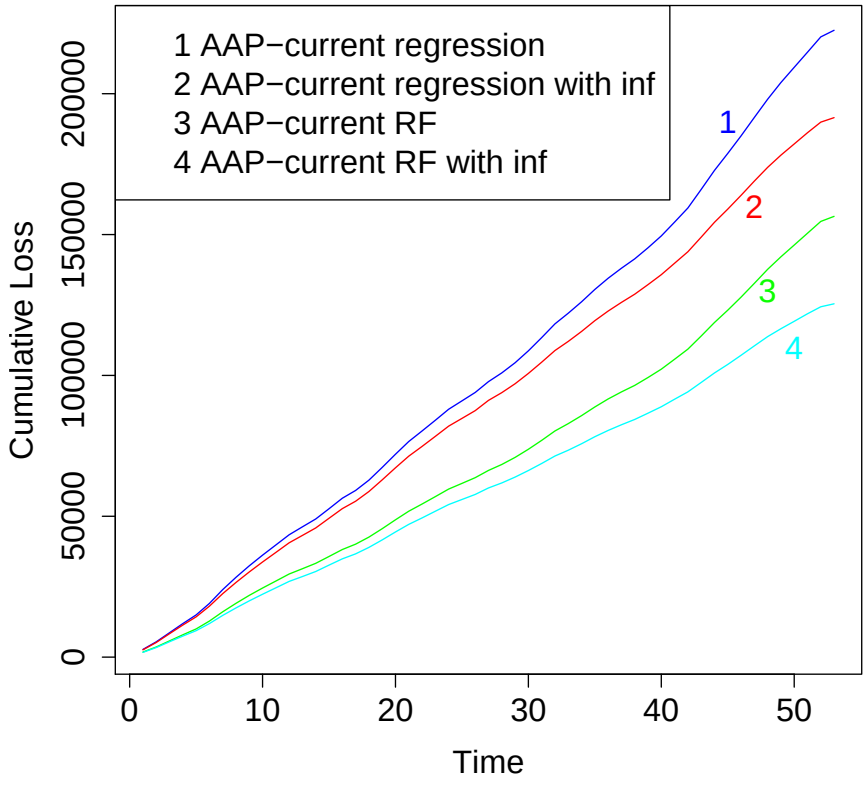
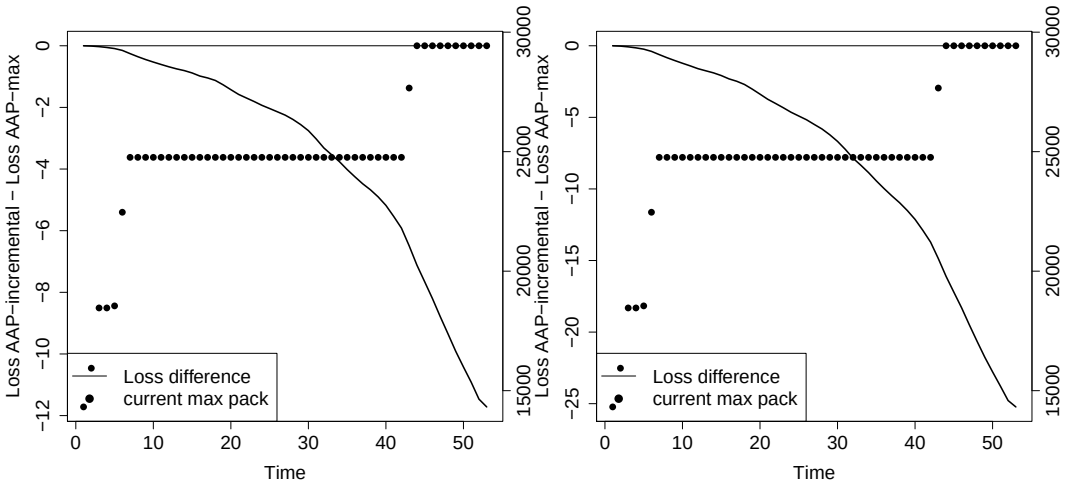


Figure 3.12: Cumulative losses of AAP-current with experts adjusted and not adjusted for inflation

We have found that the average performance of Parallel Copies of AA is close to the performance of the AAP family. Some members of the family (especially AAP-incremental and AAP-current) often perform better than the average. However, Parallel Copies may be able to benefit from extra information contained in the order.

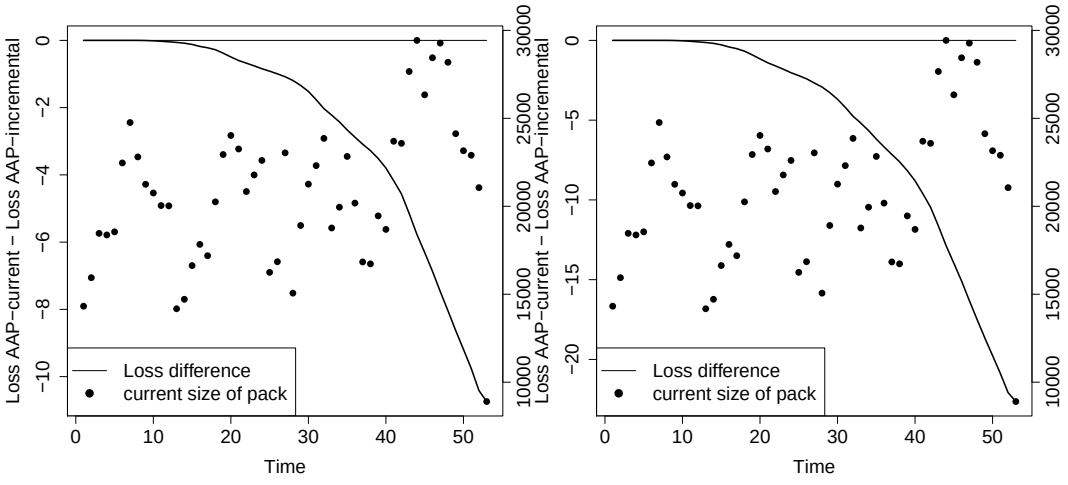
We have also found that AAP-incremental typically outperforms AAP-max, especially before the pack size has reached the maximum.

AAP-current may outperform AAP-incremental in terms of the plain loss, especially if the ratio of maximum and minimum pack sizes is small.



(a) Regression on London house prices (b) RF on London house prices

Figure 3.13: Difference of cumulative losses of AAP-incremental and AAP-max vs time on log prices with inflation



(a) Regression on London house prices (b) RF on London house prices

Figure 3.14: Difference of cumulative losses of AAP-current and AAP-incremental vs time on log prices with inflation

Chapter 4

Weak Aggregating Algorithm for Value at Risk

In this chapter, we propose to apply the method of online prediction with expert advice for estimation of Value at Risk. We show that in some cases the combination of different methods can produce better results compared to a single model.

4.1 Introduction

In the history of finance, there have been a lot of crises that deeply influenced the global economy. Examples of these crises are the Wall Street crash in 1987, the Japan financial crisis in 1989, the Asian financial crisis in 1997, the sub-prime mortgage crisis of 2007-2008 and the European debt crisis in 2010. Financial crises and the rise of uncertainty in the financial market emphasize the need of effective risk calculation.

Value at Risk (VaR) measure is one of the most important methods of risk management. The VaR method was introduced in 1994 by J. P. Morgan and became widely used by most financial institutions (Guldimann (1995)). J. P. Morgan (Morgan (1996)) defines VaR as 'a measure of the maximum possible change in the value of a portfolio of financial instruments over a pre-set horizon'.

There are several conventional methods that are widely used for measuring VaR. Historical Simulation is one of the non-parametric methods for measuring VaR, which assumes that all possible future variations have been experienced in the past and will be repeated in the future (Butler and Schachter (1996)). Another approach, known as parametric, is when one estimates volatility of assets' returns in turn to obtain their VaR. Across parametric approaches the conventional methods include Variance–Covariance, Exponential Weighted Moving Average (EWMA) (Chapter 22 in

Hull (2006)) and Generalized Autoregressive Conditional Heteroskedasticity (GARCH) (Bollerslev (1986)).

Some of the procedures to estimate VaR propose the use of quantile regression. The quantile regression approach suggested by Koenker and Basset (Koenker and Bassett (1978)) is one of the methods which models a quantile of the response variable conditional on the explanatory variables. 'It is natural to evaluate a VaR model by a quantile regression method due to its capability of conditional distribution exploration with distribution-free assumption, also allowing for serial correlation and conditional heteroskedasticity' (Gaglianone et al. (2008)). In Taylor (1999) a procedure to estimate a conditional quantile model to calculate VaRs for portfolios is presented; this method is found to be comparable with conventional methods in forecasting VaR.

In this chapter, we use the same pinball loss function as is used in optimising parameters of the quantile regression model. However, we do not try to optimize parameters of some model. Instead our approach combines predictions of different models based on the method of online prediction with expert advice. Contrary to batch mode, where the algorithm is trained on a training set and gives predictions on a test set, in online setting we learn as soon as new observations become available. In addition, previous research shows that combining predictions of multiple regressors often produces better results compared to a single model (Rokach (2010)).

In this chapter, we propose to apply the method of prediction with expert advice to estimate VaR. Our approach is based on the Weak Aggregating Algorithm (WAA), described in Section 2.5. The first approach is to apply WAA to combine predictions of normal distribution experts, where each expert has particular parameters of standard deviation. We choose to evaluate performance of proposed strategies using stocks' adjusted closing prices of Walmart, WPP inc. and Apple. The experiments show that loss of the WAA is close to or better than the loss of the retrospectively best normal distribution expert. We compare WAA with the model of quantile regression, and experimental results show that in most cases WAA outperforms quantile regression.

The second approach is to combine predictions of several conventional methods for estimating VaR, such as Historical Simulation, Variance–Covariance, EWMA and GARCH. The experiments illustrate that combining predictions of different experts sometimes could provide better results compared to the single retrospectively best model.

We run backtesting of all methods by using the Kupiec unconditional coverage test (Kupiec (1995)) and the Christoffersen conditional coverage test (Christoffersen (1998)) to do the Backtesting on VaR. WAA with normal distribution experts is the only method that fails to reject the null hypothesis for both unconditional and conditional

coverage tests.

4.2 Framework

We consider a game $\mathfrak{G}$, where space of outcomes $\Omega = \mathbb{R}$ and decision space $\Gamma = \mathbb{R}$, where for any $y \in \Omega$ and $\gamma \in \Gamma$ we define the pinball loss for $\alpha \in (0, 1)$

$$\lambda(y, \gamma) = \begin{cases} \alpha(y - \gamma), & \text{if } y \geq \gamma \\ (1 - \alpha)(\gamma - y), & \text{if } y < \gamma \end{cases}. \tag{4.1}$$

When N days is the time horizon and $1 - \alpha$ is the confidence level, $\mathrm{VaR}_{1-\alpha}$ is the loss corresponding to the α-quantile of the distribution of the gain in the value of the portfolio over the next N days (Chapter 21.1 in Hull (2006)). We consider outcomes to be returns of some stocks or portfolios. Let outcomes have a cumulative distribution function $F_Y(z) = \Pr(Y \leq z)$. Because VaR is conventionally reported as a positive number, we define

$$\mathrm{VaR}_{1-\alpha} = -\inf\{z : F_Y(z) \geq \alpha\} \tag{4.2}$$

as the negative α-quantile of Y. Then the problem of VaR estimation is equivalent to the problem of prediction of α-quantile of returns. This problem can be solved by applying the quantile regression.

Letting x_t denote a sequence of signals, suppose y_t is a random sample on the regression process $u_t = y_t - x_t\beta$. The α-th quantile regression, $0 < \alpha < 1$, is defined as any solution to the minimization problem:

$$\min_{b \in \mathbb{R}^n} \sum_{t: y_t > x_t b} \alpha |y_t - x_t b| + \sum_{t: y_t < x_t b} (1 - \alpha) |y_t - x_t b|.$$

The least absolute error estimator is the regression median, i.e., the quantile regression for $\alpha = 1/2$ (Koenker and Bassett (1978)). The loss function (4.2) is appropriate for quantile regression because on average it is minimized by the α-th quantile. Namely, if Y is a real-valued random variable with a cumulative distribution function $F_Y(z) = \Pr(Y \leq z)$, then the expectation $\mathbb{E}\lambda(Y, \gamma)$ is minimized by $\gamma = \inf\{z : F_Y(z) \geq \alpha\}$ (see Section 1.3 in Koenker (2005) for a discussion).

In the framework of prediction with expert advice the learner has access to predictions $\xi_t(1),\ \xi_t(2), \ldots, \xi_t(N)$ at time t generated by experts $\mathcal{E}_1,\ \mathcal{E}_2, \ldots, \mathcal{E}_N$ that try to predict elements of the same sequence.

Learner works according to the Protocol 1.

The performance of a learner is measured by the cumulative loss.

Let us denote L_T^i the cumulative loss of expert $\mathcal{E}_i$ at step T:

$$L_T^i := \sum_{t=1}^{T} \lambda(y_t, \xi_t(i)) = \sum_{\substack{t=1,\dots,T:\\ y_t > \xi_t(i)}} \alpha|y_t - \xi_t(i)| + \sum_{\substack{t=1,\dots,T:\\ y_t < \xi_t(i)}} (1-\alpha)|y_t - \xi_t(i)|. \tag{4.3}$$

The cumulative loss of the learner at step T is:

$$L_T := \sum_{t=1}^{T} \lambda(y_t, \gamma_t) = \sum_{\substack{t=1,\dots,T:\\ y_t > \gamma_t}} \alpha|y_t - \gamma_t| + \sum_{\substack{t=1,\dots,T:\\ y_t < \gamma_t}} (1-\alpha)|y_t - \gamma_t|. \tag{4.4}$$

4.3 Experiments

In this section, we apply WAA to the problem of prediction of VaR using three stocks of Walmart, Apple and WPP inc. We use daily adjusted closing prices from January 2011 to December 2018 that are downloaded from Yahoo Finance.[1]

4.3.1 WAA for normal distributions

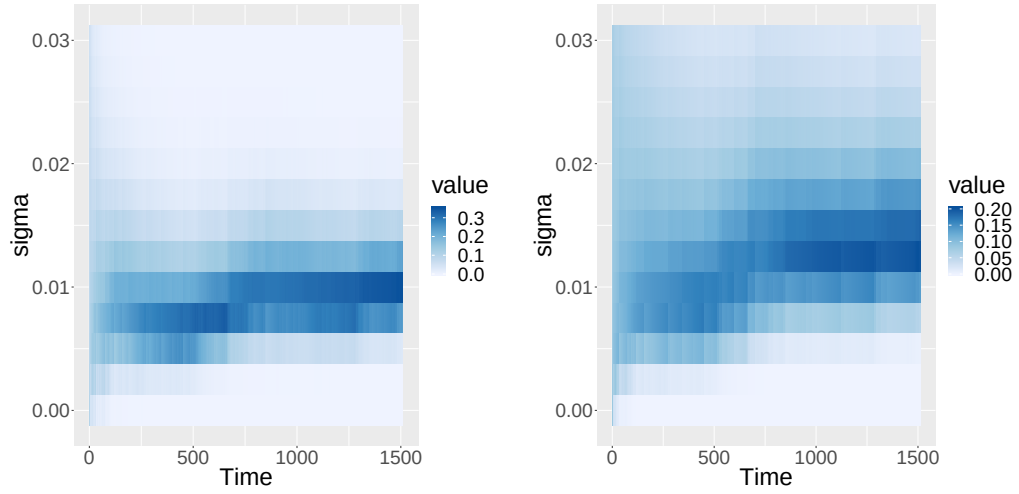
First, as a proof of concept, we apply WAA to normal distribution experts. We assume that stock investment's returns are normally distributed around the mean of a normal probability distribution. The volatility σ of a stock is a measure of our uncertainty about the returns provided by the stock. Each expert $\mathcal{E}_i$ predicts according to $\mathcal{N}(0, \sigma_i^2)$, $i = 1, \dots, N$. We pick σ_i to be in a range from 0 to 0.03 with a step 0.0025. We take the constant of WAA $c = 200$ using the historical information about maximum losses on the first 500 observations. We test the performance of WAA using dataset without the first 500 observations. Figure 4.1 shows the weights update for experts $\mathcal{E}_i$, $i = 1, \dots, N$ for Walmart. We can see from the graph that, for significance level $\alpha = 0.05$, expert with $\sigma = 0.01$ has the largest weights at the end of the period. It corresponds with Figure 4.4, where the same expert has the lowest total pinball loss. It shows that WAA converges to the best expert by updating weights of experts online based on their performance. For significance level $\alpha = 0.01$ expert with $\sigma = 0.0125$ has the largest weight at the end of the period and the lowest total loss. However, for $\alpha = 0.01$ losses of several experts are close to the loss of best expert, and as a result, their weights are also close to each other. A similar picture can be seen for WPP inc. at Figures 4.2, 4.5, and for Apple at Figures 4.3, 4.6. Tables 4.1, 4.2 summarise losses of normal distribution experts and WAA for $\alpha = 0.05$ and $\alpha = 0.01$ respectively. We

[1] The code written in R is available at https://github.com/RaisaDZ/VaR.

can see from the table that losses of WAA are very close to the loss of best normal distribution expert. For example, for WPP inc. losses of WAA are lower than losses of best experts.

We also compare the performance of WAA with quantile regression model (QR). QR was trained in online mode using sliding window of the length 500. We can see from Tables 4.1, 4.2 that in most cases losses of WAA are lower than losses of QR. Tables 4.3, 4.4 show the actual exceptions of VaR, i.e., the number of times when stock's losses exceed VaR, for each method for $\alpha = 0.05$ and $\alpha = 0.01$ respectively. It seems that WAA tends to underestimate VaR a little, while QR overestimates VaR.

This approach shows that it is reasonable to apply WAA in the considered setting. WAA converges to the best expert by updating experts' weights online. In addition, best experts might be different for different significance levels. It shows that the single retrospectively best model might not perform well in the future, and it is reasonable to apply the mixture of models instead. The performance of WAA is close to or better than the best normal distribution expert, and in most cases it outperforms the model of QR.

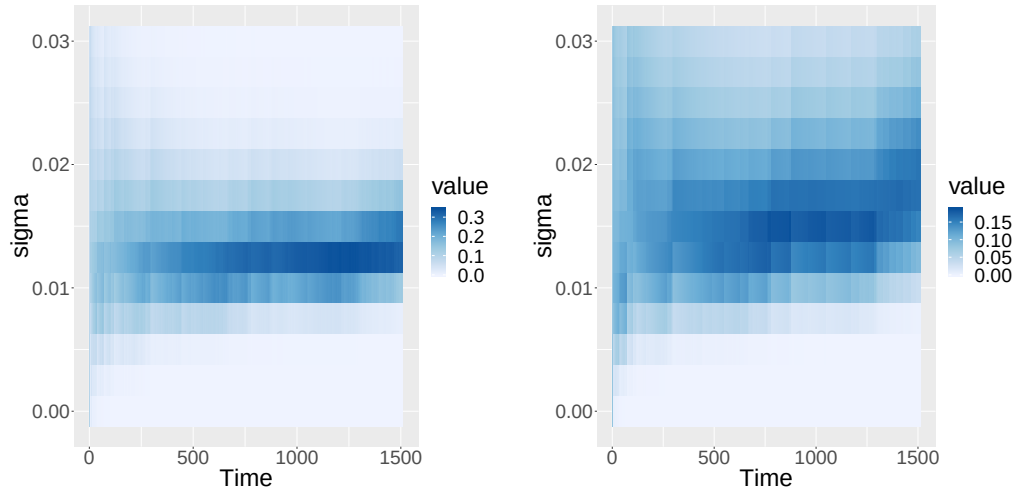


(a) $\alpha = 0.05$ (b) $\alpha = 0.01$

Figure 4.1: Weights update for Walmart

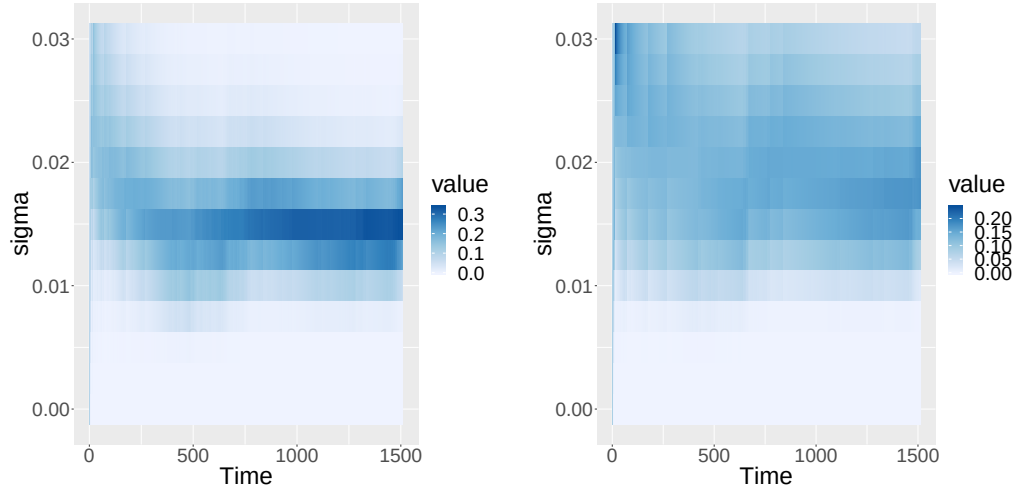
4.3.2 WAA for conventional models

In this section, we use WAA with four conventional models that are widely used to calculate VaR: Historical Simulation, Variance–Covariance, EWMA and GARCH. 'Historical simulation is one popular way of estimating VaR. It involves using past data as



(a) $\alpha = 0.05$ (b) $\alpha = 0.01$

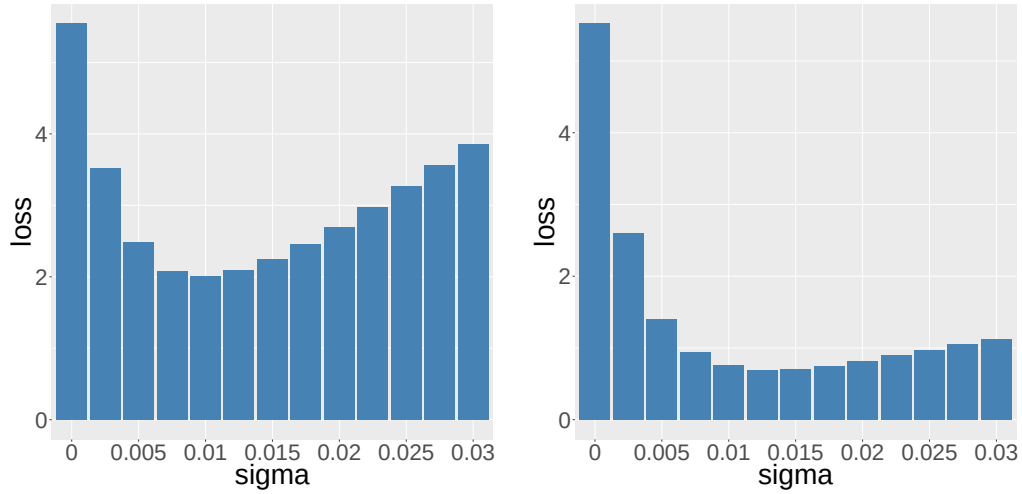
Figure 4.2: Weights update for WPP inc.



(a) $\alpha = 0.05$ (b) $\alpha = 0.01$

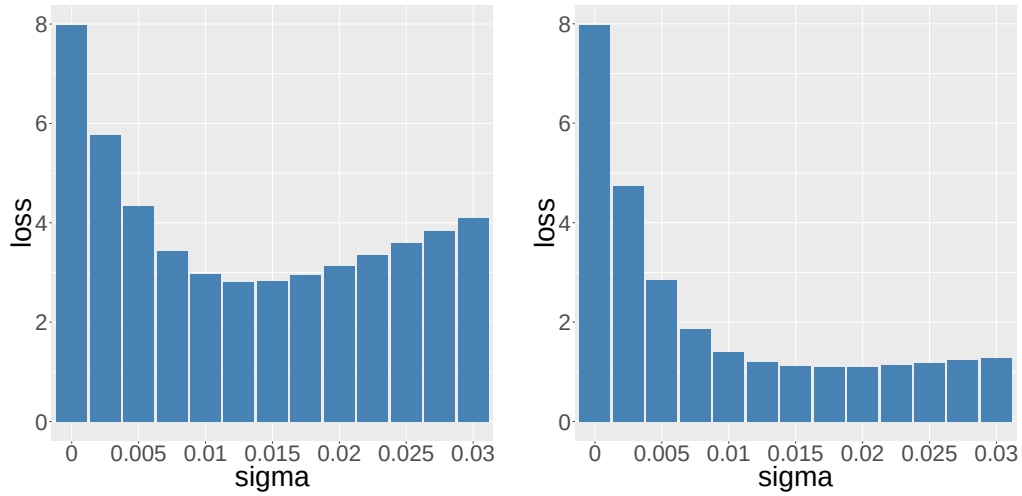
Figure 4.3: Weights update for Apple

a guide to what will happen in the future' (Section 21.2 in Hull (2006)). Suppose that we want to calculate $\text{VaR}_{1-\alpha}$ for a stock, and data are collected on movements in the market variables over the most recent N days. This provides $N-1$ alternative scenarios for what can happen between today and tomorrow. The estimate of $\text{VaR}_{1-\alpha}$ is the negative α-quantile (4.2) of returns based on $N-1$ historical scenarios. The Variance-Covariance method is one of the parametric methods which estimates the volatility of returns based on the normal distribution assumption. Then VaR is calculated as the



(a) $\alpha = 0.05$ (b) $\alpha = 0.01$

Figure 4.4: Losses of normal distribution experts for Walmart

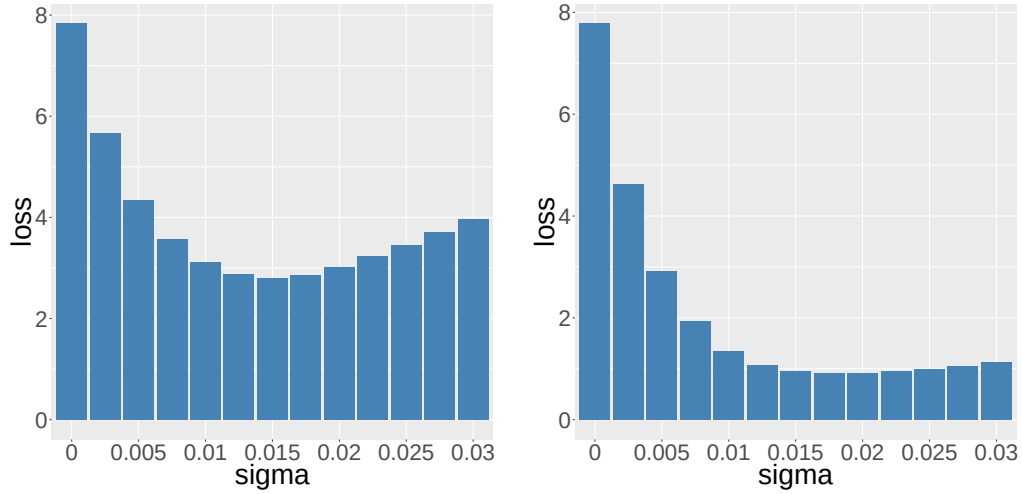


(a) $\alpha = 0.05$ (b) $\alpha = 0.01$

Figure 4.5: Losses of normal distribution experts for WPP inc.

α-quantile of the normal distribution with zero mean and the estimated volatility. The exponentially weighted moving average (EWMA) is another parametric method, where the estimate of the volatility σ_t for day t is given by the formula

$$\sigma_t^2 = \lambda\sigma_{t-1}^2 + (1-\lambda)u_{t-1}^2,$$



(a) $\alpha = 0.05$ (b) $\alpha = 0.01$

Figure 4.6: Losses of normal distribution experts for Apple

Table 4.1: Total losses of normal distribution experts for $\alpha = 0.05$.

sigma	WMT	WPP	AAPL
0	5.545	7.974	7.834
0.0025	3.515	5.775	5.655
0.005	2.478	4.329	4.337
0.0075	2.083	3.427	3.561
0.01	**2.007**	2.975	3.113
0.0125	2.088	**2.811**	2.876
0.015	2.252	2.828	**2.788**
0.0175	2.450	2.948	2.865
0.02	2.700	3.130	3.023
0.0225	2.975	3.346	3.228
0.025	3.262	3.587	3.453
0.0275	3.556	3.838	3.702
0.03	3.857	4.094	3.968
WAA	2.013	2.806	2.834
QR	2.089	2.851	2.761

where $0 < \lambda < 1$ is a constant, σ_{t-1} is the volatility estimate at the end of day $t-2$ of the volatility for day $t-1$ and u_{t-1} is the most recent daily percentage change in returns (Section 22.2 in Hull (2006)). Finally, the GARCH(p, q) estimates the volatility σ_t for day t as

$$\sigma_t^2 = \alpha_0 + \sum_{i=1}^{q} \alpha_i u_{t-i}^2 + \sum_{i=1}^{p} \beta_i \sigma_{t-i}^2.$$

Table 4.2: Total losses of normal distribution experts for $\alpha = 0.01$.

sigma	WMT	WPP	AAPL
0	5.523	7.970	7.782
0.0025	2.604	4.745	4.619
0.005	1.397	2.838	2.910
0.0075	0.939	1.854	1.935
0.01	0.763	1.397	1.344
0.0125	**0.688**	1.199	1.066
0.015	0.702	1.117	0.959
0.0175	0.751	**1.099**	**0.919**
0.02	0.817	1.103	0.923
0.0225	0.894	1.128	0.953
0.025	0.970	1.177	1.001
0.0275	1.046	1.230	1.059
0.03	1.122	1.283	1.131
WAA	0.705	1.085	0.930
QR	0.796	1.181	1.080

Table 4.3: Actual exceptions of normal distribution experts for $\alpha = 0.05$.

	expected = 75.5		
sigma	WMT	WPP	AAPL
0	711	720	721
0.0025	439	501	492
0.005	227	360	320
0.0075	123	234	219
0.01	74	143	155
0.0125	43	90	115
0.015	31	58	79
0.0175	20	37	42
0.02	10	28	29
0.0225	7	19	23
0.025	5	15	18
0.0275	3	14	12
0.03	2	11	10
WAA	72	73	63
QR	92	86	85

In our experiments we use GARCH(1, 1) which is based on the most recent volatility estimates and the most recent returns' changes.

We train these models using a sliding window of length 500, and then apply WAA using forecasts of these models to predict a one-step ahead forecast. We re-train all models except GARCH(1, 1) after each new observation becomes available, for GARCH(1, 1)

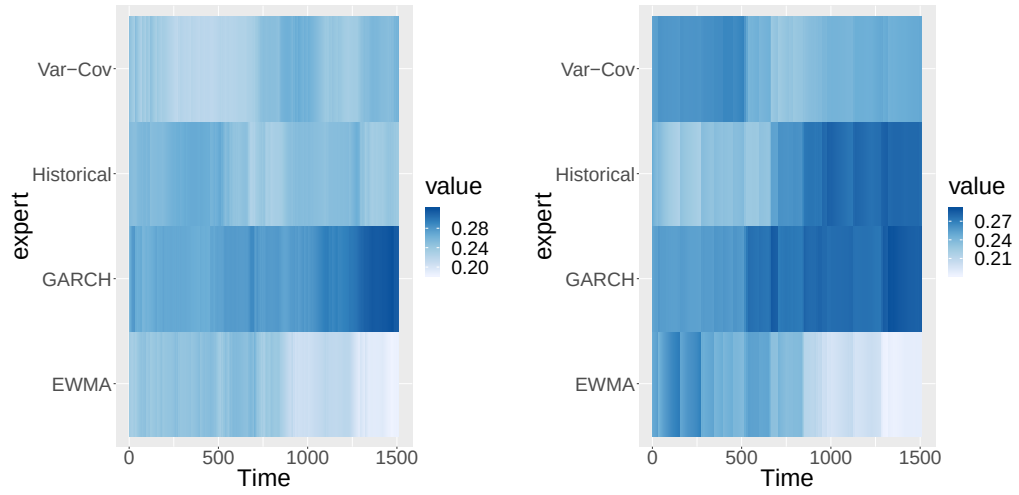
Table 4.4: Actual exceptions of normal distribution experts for $\alpha = 0.01$.

	expected = 15.1		
sigma	WMT	WPP	AAPL
0	711	720	721
0.0025	339	434	407
0.005	140	251	226
0.0075	63	129	144
0.01	33	64	91
0.0125	20	36	42
0.015	9	23	26
0.0175	5	15	19
0.02	2	14	11
0.0225	2	8	8
0.025	2	6	6
0.0275	2	6	4
0.03	2	6	2
WAA	9	14	12
QR	22	32	28

we do it after each 50 steps due to computational complexity of this method. We start with equal initial weights of each model and then update their weights according to their current performance.

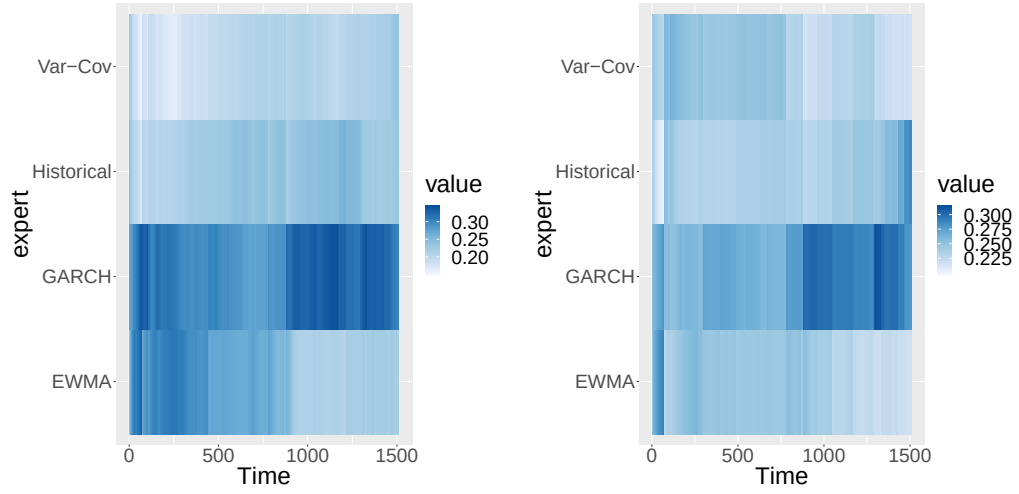
Figures 4.7, 4.8, 4.9 illustrate weights of each model depending on the current time step. Figure 4.10 with the corresponding Tables 4.5, 4.6 show total losses of each model and WAA for $\alpha = 0.05$ and $\alpha = 0.01$ respectively. We can see from the graphs that in most cases GARCH(1, 1) obtains the largest weights which indicates that it suffers smaller losses compared to other models. However, it changes for $\alpha = 0.01$ for WPP inc., where the largest weights are acquired by Historical Simulation model. It shows that sometimes we cannot use the past information to evaluate the best model. The retrospectively best model can perform worse in the future as an underlying nature of data generating can change. In addition, different models can perform better for different significance levels of VaR.

Similar to the previous experiments, losses of WAA are very close to the loss of the best performing expert. In most of the cases the best expert is GARCH(1, 1), and WAA follows its predictions. However, for $\alpha = 0.01$ for WPP inc. the best expert changes. It again illustrates that the retrospectively best model could change with time, and one should be cautious about choosing the single retrospectively best model for future forecasts.



(a) $\alpha = 0.05$ (b) $\alpha = 0.01$

Figure 4.7: Weights update for Walmart

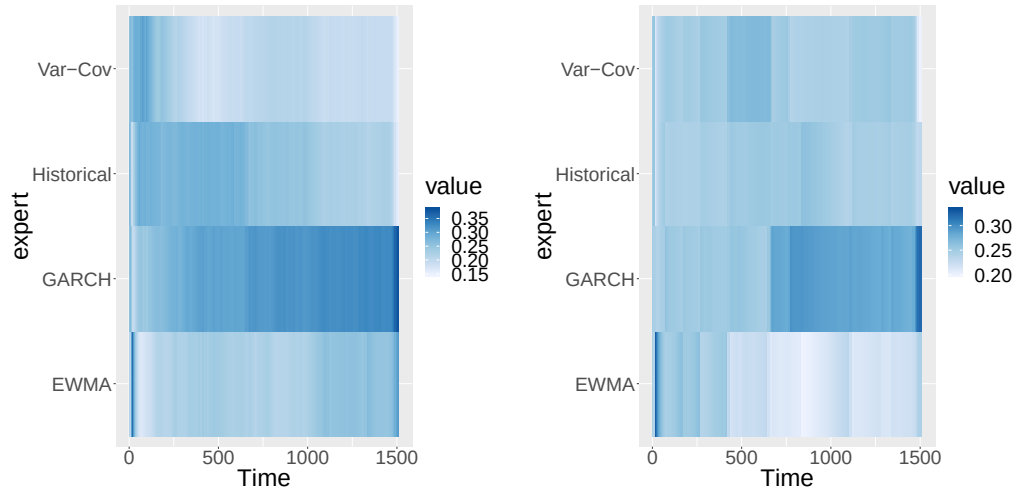


(a) $\alpha = 0.05$ (b) $\alpha = 0.01$

Figure 4.8: Weights update for WPP inc.

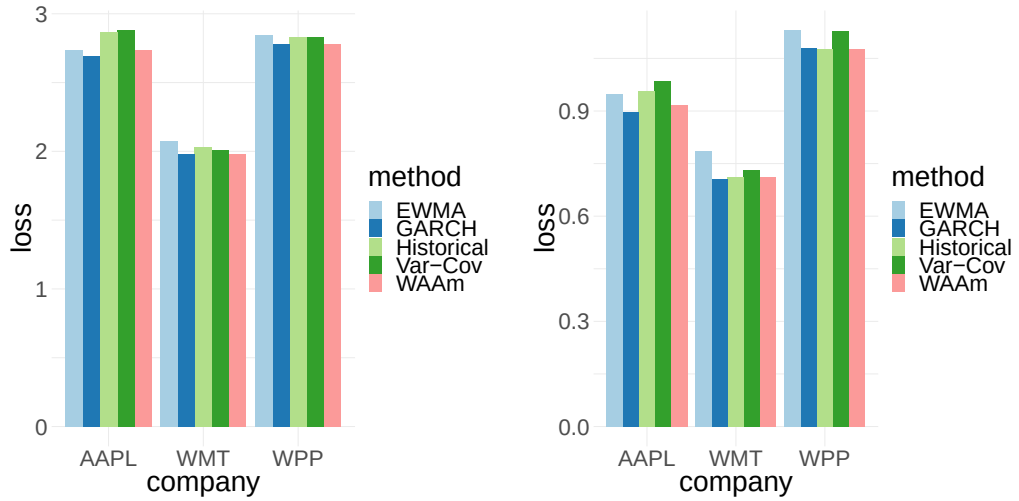
4.3.3 Backtesting

First, we introduce the Kupiec unconditional coverage test, which is also known as the proportion of failures test. The most common way to test the performance of VaR models is to count the number of exceptions (failures), i.e., the number of times when stock's losses exceed VaR. Denoting m to be the number of exceptions, we define the failure rate during the time horizon T as m/T. The Kupiec unconditional coverage test measures whether the number of exceptions is consistent with the confidence level.



(a) $\alpha = 0.05$ (b) $\alpha = 0.01$

Figure 4.9: Weights update for Apple



(a) $\alpha = 0.05$ (b) $\alpha = 0.01$

Figure 4.10: Total losses of methods

The null hypothesis H_0 is

$$H_0 : \alpha = \hat{\alpha} = m/T,$$

where $\hat{\alpha}$ is the observed failure rate and α is the significance level of $\mathrm{VaR}_{1-\alpha}$. According to Kupiec (1995) the test statistics takes the form of a likelihood ratio test:

$$\mathrm{LR}_{\mathrm{UC}} = -2 \ln \left(\frac{(1-p)^{T-m} p^m}{(1-m/T)^{T-m} (m/T)^m} \right).$$

Table 4.5: Total losses of methods for $\alpha = 0.05$.

Method	WMT	WPP	AAPL
Historical	2.031	2.829	2.867
Var-Cov	2.012	2.827	2.880
EWMA	2.077	2.845	2.734
GARCH(1, 1)	**1.978**	**2.781**	**2.695**
WAAm	1.983	2.782	2.733

Table 4.6: Total losses of methods for $\alpha = 0.01$.

Method	WMT	WPP	AAPL
Historical	0.711	1.076	0.956
Var-Cov	0.731	1.129	0.986
EWMA	0.786	1.130	0.948
GARCH(1, 1)	**0.706**	1.081	**0.896**
WAAm	0.713	**1.075**	0.917

This statistic is asymptotically distributed as a chi-square variable with 1 degree of freedom. The VaR model fails the test if this likelihood ratio exceeds a critical value. The critical value depends on the test confidence level.

The Kupiec unconditional coverage test focuses only on the number of exceptions. However, we would like to test whether these exceptions were evenly spread over time. The null hypothesis H_0 for Christoffersen conditional coverage test is that the probability of observing an exception on a particular day does not depend on whether an exception occurred. The test statistic for independence is given by

$$\mathrm{LR}_{\mathrm{CCI}} = -2\ln\left(\frac{(1-\pi)^{n_{00}+n_{10}}\pi^{n_{01}+n_{11}}}{(1-\pi_0)^{n_{00}}\pi_0^{n_{01}}(1-\pi_1)^{n_{10}}\pi_1^{n_{11}}}\right),$$

where n_{00} is the number of periods with no failures followed by a period with no failures,
n_{10} is the number of periods with failures followed by a period with no failures,
n_{01} is the number of periods with no failures followed by a period with failures,
n_{11} is the number of periods with failures followed by a period with failures,
and π_i is the probability of having a failure conditional on the previous period:

$$\pi_0 = \frac{n_{01}}{n_{00}+n_{01}},\ \pi_1 = \frac{n_{11}}{n_{10}+n_{11}} \text{ and } \pi = \frac{n_{01}+n_{11}}{n_{00}+n_{01}+n_{10}+n_{11}}.$$

This statistic is asymptotically distributed as a chi-square with 1 degree of freedom. The Christoffersen conditional coverage test is a combination of this statistic with the

frequency unconditional coverage test:

$$\mathrm{LR}_{\mathrm{CC}} = \mathrm{LR}_{\mathrm{CCI}} + \mathrm{LR}_{\mathrm{UC}}.$$

This test is asymptotically distributed as a chi-square variable with 2 degrees of freedom.

In this section, we perform backtesting of all considered methods by running Kupiec unconditional coverage test and Christoffersen conditional coverage test. Tables 4.7, 4.8 and 4.9 show results for $\alpha = 0.05$ for Walmart, WPP inc. and Apple respectively. Tables 4.10, 4.11 and 4.12 illustrate results for $\alpha = 0.01$. UCD and CCD denotes decisions for unconditional and conditional coverage tests respectively. We denote WAAn the method considered in Section 4.3.1, and WAAm is the method from Section 4.3.2. We can see from the tables that WAA for normal distribution experts (WAAn) is the only method that fails to reject the null hypothesis H_0. The second best performing model seems to be GARCH(1, 1) as it rejects the only test case for WPP inc. with significance level $\alpha = 0.01$. In Table 4.10 we can see that all methods reject the null hypothesis except WAAn. WAA for conventional model experts (WAAm) sometimes rejects the null hypothesis. It happens in situations when most of models that are used in WAAm reject the null hypothesis

Figures 4.11, 4.12 and 4.13 illustrate returns of each company and VaR for WAAn and WAAm. The behavior of VaR for WAAn is smooth because WAAn uses predictions of constant normal distribution experts. VaR of WAAm has more spikes because it uses predictions of methods such as Historical Simulation, Variance-Covariance, EWMA and GARCH(1, 1) which have more fluctuations in their predictions.

Table 4.7: Walmart, $\alpha = 0.05$, expected = 75.5.

Method	Actual	Luc	Lcc	UCD	CCD
Historical	95	0.0266	0.0398	Reject H0	Reject H0
Var-Cov	58	0.0315	0.0869	Reject H0	Fail to Reject H0
EWMA	69	0.4364	0.4433	Fail to Reject H0	Fail to Reject H0
GARCH(1, 1)	69	0.4364	0.6582	Fail to Reject H0	Fail to Reject H0
QR	92	0.0592	0.0618	Fail to Reject H0	Fail to Reject H0
WAAn	72	0.6772	0.8733	Fail to Reject H0	Fail to Reject H0
WAAm	64	0.1637	0.1609	Fail to Reject H0	Fail to Reject H0

Table 4.8: WPP inc., $\alpha = 0.05$, expected = 75.5.

Method	Actual	Luc	Lcc	UCD	CCD
Historical	84	0.3238	0.0056	Fail to Reject H0	Reject H0
Var-Cov	60	0.0580	0.0192	Fail to Reject H0	Reject H0
EWMA	74	0.8590	0.0387	Fail to Reject H0	Reject H0
GARCH(1, 1)	78	0.7690	0.3462	Fail to Reject H0	Fail to Reject H0
QR	86	0.2247	0.0978	Fail to Reject H0	Fail to Reject H0
WAAn	73	0.7667	0.0891	Fail to Reject H0	Fail to Reject H0
WAAm	67	0.3066	0.0218	Fail to Reject H0	Reject H0

Table 4.9: Apple, $\alpha = 0.05$, expected = 75.5.

Method	Actual	Luc	Lcc	UCD	CCD
Historical	85	0.2711	0.0005	Fail to Reject H0	Reject H0
Var-Cov	72	0.6772	0.0020	Fail to Reject H0	Reject H0
EWMA	66	0.2521	0.0160	Fail to Reject H0	Reject H0
GARCH(1, 1)	82	0.4488	0.3711	Fail to Reject H0	Fail to Reject H0
QR	85	0.2711	0.3277	Fail to Reject H0	Fail to Reject H0
WAAn	63	0.1291	0.0536	Fail to Reject H0	Fail to Reject H0
WAAm	72	0.6772	0.1831	Fail to Reject H0	Fail to Reject H0

Table 4.10: Walmart, $\alpha = 0.01$, expected = 15.1.

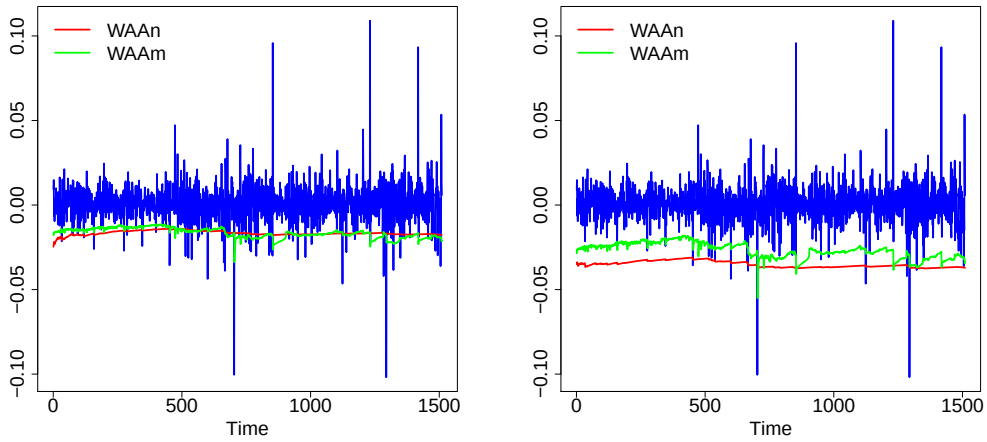
Method	Actual	Luc	Lcc	UCD	CCD
Historical	17	0.6300	0.7336	Fail to Reject H0	Fail to Reject H0
Var-Cov	30	0.0007	0.0028	Reject H0	Reject H0
EWMA	35	0.0000	0.0001	Reject H0	Reject H0
GARCH(1, 1)	20	0.2273	0.2594	Fail to Reject H0	Fail to Reject H0
QR	22	0.0948	0.1789	Fail to Reject H0	Fail to Reject H0
WAAn	9	0.0880	0.2211	Fail to Reject H0	Fail to Reject H0
WAAm	24	0.0340	0.0737	Reject H0	Fail to Reject H0

Table 4.11: WPP inc., $\alpha = 0.01$, expected = 15.1.

Method	Actual	Luc	Lcc	UCD	CCD
Historical	18	0.4666	0.0437	Fail to Reject H0	Reject H0
Var-Cov	26	0.0106	0.0082	Reject H0	Reject H0
EWMA	30	0.0007	0.0028	Reject H0	Reject H0
GARCH(1, 1)	26	0.0106	0.0241	Reject H0	Reject H0
QR	32	0.0001	0.0001	Reject H0	Reject H0
WAAn	14	0.7733	0.2813	Fail to Reject H0	Fail to Reject H0
WAAm	24	0.0340	0.0737	Reject H0	Fail to Reject H0

Table 4.12: Apple, $\alpha = 0.01$, expected = 15.1.

Method	Actual	Luc	Lcc	UCD	CCD
Historical	21	0.1496	0.2049	Fail to Reject H0	Fail to Reject H0
Var-Cov	24	0.0340	0.0737	Reject H0	Fail to Reject H0
EWMA	22	0.0948	0.1789	Fail to Reject H0	Fail to Reject H0
GARCH(1, 1)	15	0.9793	0.8599	Fail to Reject H0	Fail to Reject H0
QR	28	0.0029	0.0032	Reject H0	Reject H0
WAAn	12	0.4057	0.6428	Fail to Reject H0	Fail to Reject H0
WAAm	19	0.3322	0.4904	Fail to Reject H0	Fail to Reject H0



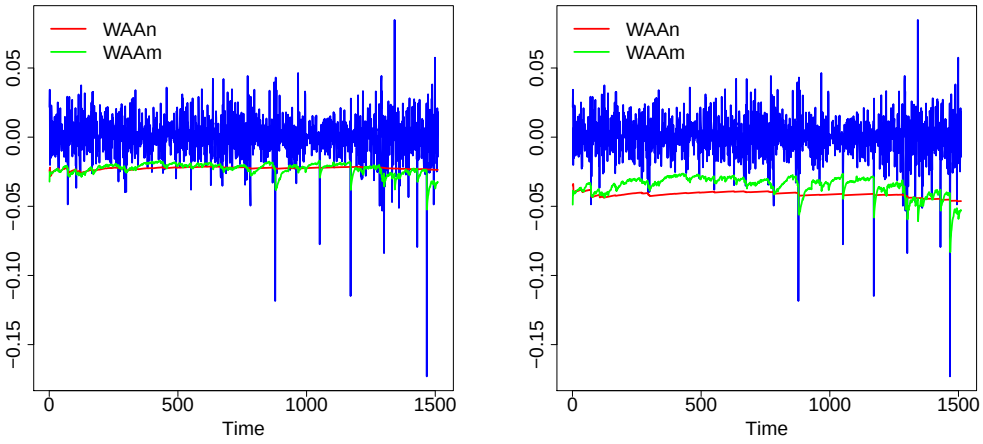
(a) $\alpha = 0.05$ (b) $\alpha = 0.01$

Figure 4.11: VaR for Walmart

4.4 Conclusions

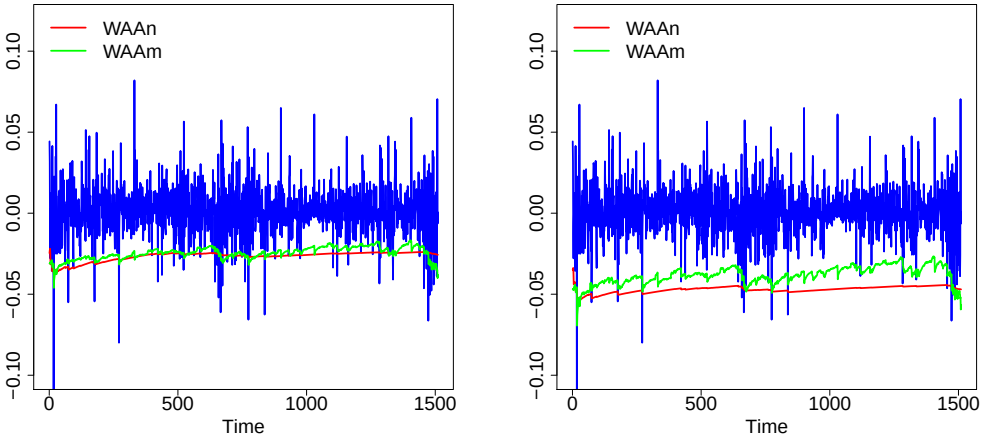
We proposed two ways of applying the framework of prediction with expert advice for calculating VaR. The first approach is to apply WAA with normal distribution experts. The experiments show that WAA converges to the best expert by updating weights of experts online based on their current performance, and its loss is close to or better than the loss of the best expert. WAA also outperforms the quantile regression model that is built using sliding window.

The second approach is to combine predictions of different methods: Historical Simulation, Variance–Covariance, EWMA and GARCH(1, 1). Similar to the previous experiments, losses of WAA are very close to the loss of best performing model, and sometimes WAA shows a better performance. The experiments illustrate that the retrospectively best model could change with time, and combining predictions of different



(a) $\alpha = 0.05$ (b) $\alpha = 0.01$

Figure 4.12: VaR for WPP inc.



(a) $\alpha = 0.05$ (b) $\alpha = 0.01$

Figure 4.13: VaR for Apple

experts could provide better results.

We compare performances of all different methods of prediction of VaR by running Kupiec unconditional coverage test and Christoffersen conditional coverage test. WAA for normal distribution experts is the only method which fails to reject the null hypothesis for both unconditional and conditional coverage tests.

Chapter 5

Universal algorithms for quantile regression

In this chapter, we construct universal algorithms for quantile regression. First, we propose to apply the framework of prediction with expert advice for the prediction of quantiles. Second, we propose a new universal algorithm Weak Aggregating Algorithm for Quantile Regression (WAAQR) and prove a theoretical bound on the cumulative loss of the proposed strategy. The theoretical bound ensures that WAAQR is asymptotically as good as any quantile regression.

5.1 Introduction

Probabilistic forecasting attracts an increasing attention in sports, finance, weather and energy fields. While an initial focus has been on deterministic forecasting, probabilistic prediction provides a more useful information which is essential for optimal planning and management in these fields. Probabilistic forecasts serve to quantify the uncertainty in a prediction, and they are an essential ingredient of optimal decision making (Gneiting and Katzfuss (2014)). An overview of the state of the art methods and scoring rules in probabilistic forecasting can be found in Gneiting and Katzfuss (2014). Quantile regression is one of the methods which models a quantile of the response variable conditional on the explanatory variables (Koenker (2005)).

Due to its ability to provide interval predictions, quantile regression found its niche in the renewable energy forecasting area. Wind power is one of the fastest growing renewable energy sources (Barton and Infield (2004)). As there is no efficient way to store wind power, producing accurate wind power forecasts are essential for reliable operation of wind turbines. Due to the uncertainty in wind power generation, there

have been studies for improving the reliability of power forecasts to ensure the balance between supply and demand at electricity market. Quantile regression has been extensively used to produce wind power quantile forecasts, using a variety of explanatory variables such as wind speed, temperature and atmospheric pressure (Koenker and Bassett (1978)).

The Global Energy Forecasting Competition 2014 showed that combining predictions of several regressors can produce better results compared to a single model. It is shown in Nagya et al. (2016) that a voted ensemble of several quantile predictors could produce good results in probabilistic solar and wind power forecasting. In Alessandrini et al. (2015) the analogue ensemble technique is applied for prediction of solar power which slightly outperforms the quantile regression model.

In this chapter, we apply a different approach to combine predictions of several models based on the Weak Aggregating Algorithm. It is possible to apply the WAA to combine predictions of an infinite pool of experts. In Levina et al. (2010) the WAA was applied to the multi-period, distribution-free perishable inventory problem, and it was shown that the asymptotic average performance of the proposed method was as good as any time-dependent stocking rule up to an additive term of the form $C\sqrt{T}\ln T$.

We propose two methods to solve the problem of prediction of quantiles. First, as a proof of concept, we apply the WAA to a finite pool of experts to show that this method is applicable for this problem. As our experts we pick several models that provide quantile forecasts and then combine their predictions using the WAA. To the best of our knowledge, prediction with expert advice was not applied before for the prediction of quantiles. Second, we propose a new universal algorithm Weak Aggregating Algorithm for Quantile Regression (WAAQR), which is as good as any quantile regression up to an additive term of the form $C\sqrt{T}\ln T$. For this purpose, we apply the WAA to an infinite pool of quantile regressions. While the bound for the finite case can be straightforwardly applied to finite or countable sets of experts, every case of a continuous pool needs to be dealt with separately as there is no generic procedure for deriving a theoretical bound for the cumulative loss of the algorithm. WAAQR can be implemented by using Markov chain Monte Carlo (MCMC) method in a way which is similar to the algorithm introduced in Zhdanov and Vovk (2010), where the AAR was applied to generalised linear regression class of function for making a prediction in a fixed interval. We derive a theoretical bound on the cumulative loss of our algorithm which is approximate (in the number of MCMC steps). MCMC is only a method for evaluating the integral and it can be replaced by a different numerical method. Theoretical convergence of the Metropolis-Hastings method in this case follows from Theorems 1 and 3 in Roberts and Smith (1993). Estimating the convergence speed

is more difficult. With the experiments provided we show that by tuning parameters online, our algorithm moves fast to the area of high values of the probability function and gives a good approximation of the prediction.

We apply both methods to the problem of probabilistic forecasting of wind and solar power. Experimental results show a good performance of both methods. WAA applied to a finite set of models performs close or better than the retrospectively best model, whereas WAAQR outperforms the best quantile regression model that was trained on the historical data.

5.2 Framework

We consider a game $\mathfrak{G}$ with the space of outcomes $\Omega = [A, B]$ and decision space $\Gamma = \mathbb{R}$, and as a loss function we take the pinball loss, defined in (4.1): for $\alpha \in (0, 1)$

$$\lambda(y, \gamma) = \begin{cases} \alpha(y - \gamma), & \text{if } y \geq \gamma \\ (1 - \alpha)(\gamma - y), & \text{if } y < \gamma \end{cases}.$$

The game $\mathfrak{G}$ is the same as defined in Section 4.2, except that the outcome space is bounded. In many tasks predicted outcomes are bounded. For example, wind and solar power cannot reach infinity. Therefore, it is possible to have a sensible estimate for the outcome space Ω based on the historical information.

Learner works according to the Protocol 1 defined in Section 2.1.

The cumulative loss of the learner at the step T is defined in (4.4):

$$L_T := \sum_{t=1}^{T} \lambda(y_t, \gamma_t) = \sum_{\substack{t=1,\dots,T: \\ y_t > \gamma_t}} \alpha|y_t - \gamma_t| + \sum_{\substack{t=1,\dots,T: \\ y_t < \gamma_t}} (1 - \alpha)|y_t - \gamma_t|.$$

Let us denote L_T^θ the cumulative loss of expert $\mathcal{E}_\theta$ at the step T, defined in (4.3):

$$L_T^\theta := \sum_{t=1}^{T} \lambda(y_t, \xi_t(\theta)) = \sum_{\substack{t=1,\dots,T: \\ y_t > \xi_t(\theta)}} \alpha|y_t - \xi_t(\theta)| + \sum_{\substack{t=1,\dots,T: \\ y_t < \xi_t(\theta)}} (1 - \alpha)|y_t - \xi_t(\theta)|.$$

In this chapter, we use expert's index θ as we deal with an infinite pool of prediction strategies $\Theta = \mathbb{R}^n$.

We want to find a strategy which is capable of competing in terms of cumulative

loss with all prediction strategies $\mathcal{E}_\theta$, $\theta \in \mathbb{R}^n$ which at step t outputs

$$\xi_t(\theta) = x_t'\theta, \tag{5.1}$$

where x_t is a signal at time t. In other terms, we want to combine a class of quantile regressions in a way that allows us to be asymptotically the same as the best quantile regression. As discussed in Section 4.2, the α-th quantile regression, $0 < \alpha < 1$, is defined as any solution to the minimization problem:

$$\min_{b\in\mathbb{R}^n} \sum_{t:y_t>x_tb} \alpha|y_t - x_tb| + \sum_{t:y_t<x_tb} (1-\alpha)|y_t - x_tb|.$$

5.3 Theoretical Bounds for WAAQR

In this section, we formulate and prove the theoretical bounds of our algorithm.

Theorem 5.3.1. *Let $a > 0$, $y \in \Omega = [A, B]$ and $\gamma \in \Gamma$. There exists a prediction strategy for Learner such that for every positive integer T, every sequence of outcomes of length T, and every $\theta \in \mathbb{R}^n$ the cumulative loss L_T of Learner satisfies*

$$L_T \le L_T^\theta + \sqrt{T}a\|\theta\|_1 + \sqrt{T}\left(n\ln\left(1 + \frac{\sqrt{T}}{a}\max_{t=1,\dots,T}\|x_t\|_\infty\right) + (B-A)^2\right).$$

The theorem states that the algorithm predicts as well as the best quantile regression, defined in (5.1), up to an additive regret of the order $\sqrt{T}\ln T$. The choice of the regularisation parameter a is important as it affects the behaviour of the theoretical bound of our algorithm. Large parameters of regularisation increase the bound by an additive term $\sqrt{T}a\|\theta\|_1$, however the regret term has a smaller growth rate as time increases. As the maximum time T is usually not known in advance, the regularisation parameter a cannot be optimised, and its choice depends on the particular task.

Proof. We choose an initial distribution of parameters

$$P_0(d\theta) = \left(\frac{a}{2}\right)^n e^{-a\|\theta\|_1} d\theta, \tag{5.2}$$

for some $a > 0$.

We consider that outcomes come from the interval $[A, B]$, and it is known in ad-

vance. Let us define the truncated expert $\tilde{\mathcal{E}}_\theta$ which at step t outputs:

$$\tilde{\xi}_t(\theta) = \begin{cases} A, & \text{if } x_t'\theta < A \\ x_t'\theta, & \text{if } A \le x_t'\theta \le B \\ B, & \text{if } x_t'\theta > B \end{cases}. \tag{5.3}$$

Let us denote $\tilde{L}_T^\theta$ the cumulative loss of expert $\tilde{\mathcal{E}}_\theta$ at the step T:

$$\tilde{L}_T^\theta := \sum_{t=1}^{T} \lambda(y_t, \tilde{\xi}_t(\theta)). \tag{5.4}$$

We apply WAA for truncated experts $\tilde{\mathcal{E}}_\theta$. As experts $\tilde{\mathcal{E}}_\theta$ output predictions inside the interval $[A, B]$, and predictions of WAA is a weighted average of experts' predictions (2.30), then γ lies in the interval $[A, B]$.

We can bound the maximum loss at each time step:

$$L := \max_{y\in[A,B],\ \gamma\in[A,B]} \lambda(y,\gamma) \le (B-A)\max(\alpha, 1-\alpha) \le B - A. \tag{5.5}$$

Applying Lemma 2.5.2 for initial distribution (5.2) and putting the maximum loss (5.5) we obtain:

$$L_T \le \sqrt{T}\left(-\ln\left(\left(\frac{a}{2}\right)^n \int_{\mathbb{R}^n} e^{-\tilde{J}(\theta)} d\theta\right) + (B-A)^2\right), \tag{5.6}$$

where

$$\tilde{J}(\theta) := \frac{\tilde{L}_T^\theta}{\sqrt{T}} + a\|\theta\|_1. \tag{5.7}$$

For all $\theta, \theta_0 \in \mathbb{R}^n$ we have:

$$\begin{aligned} \sum_{\substack{t=1,\dots,T:\\ y_t < x_t'\theta}} |x_t'\theta - y_t| &\le \sum_{\substack{t=1,\dots,T:\\ y_t < x_t'\theta}} |x_t'\theta_0 - y_t| + \sum_{\substack{t=1,\dots,T:\\ y_t < x_t'\theta}} |x_t'\theta - x_t'\theta_0| \\ &\le \sum_{\substack{t=1,\dots,T:\\ y_t < x_t'\theta}} |x_t'\theta_0 - y_t| + \sum_{\substack{t=1,\dots,T:\\ y_t < x_t'\theta}} \max_{t=1,\dots,T} \|x_t\|_\infty \|\theta - \theta_0\|_1 \\ &\le \sum_{\substack{t=1,\dots,T:\\ y_t < x_t'\theta}} |x_t'\theta_0 - y_t| + T \max_{t=1,\dots,T} \|x_t\|_\infty \|\theta - \theta_0\|_1. \end{aligned} \tag{5.8}$$

Analogously, we have:

$$\sum_{\substack{t=1,\dots,T:\\ y_t>x_t'\theta}} |x_t'\theta - y_t| \le \sum_{\substack{t=1,\dots,T:\\ y_t>x_t'\theta}} |x_t'\theta_0 - y_t| + T \max_{t=1,\dots,T} \|x_t\|_\infty \|\theta-\theta_0\|_1. \tag{5.9}$$

By multiplying inequality (5.8) by $(1-\alpha)$, inequality (5.9) by α and summing them, we have:

$$L_T^\theta \le L_T^{\theta_0} + T \max_{t=1,\dots,T} \|x_t\|_\infty \|\theta-\theta_0\|_1. \tag{5.10}$$

The cumulative loss of truncated expert $\tilde{\mathcal{E}}_\theta$ cannot exceed the cumulative loss of non-truncated expert $\mathcal{E}_\theta$ for all $\theta \in \mathbb{R}^n$:

$$\tilde{L}_T^\theta \le L_T^\theta. \tag{5.11}$$

By dividing (5.10) by $\sqrt{T}$ and adding $a\|\theta\|_1$ to both parts, we have:

$$\begin{aligned}\tilde{J}(\theta) \le J(\theta) &\le J(\theta_0) + \sqrt{T} \max_{t=1,\dots,T} \|x_t\|_\infty \|\theta-\theta_0\|_1 + a(\|\theta\|_1 - \|\theta_0\|_1) \\ &\le J(\theta_0) + (\sqrt{T} \max_{t=1,\dots,T} \|x_t\|_\infty + a)\|\theta-\theta_0\|_1, \end{aligned} \tag{5.12}$$

where

$$J(\theta) := \frac{L_T^\theta}{\sqrt{T}} + a\|\theta\|_1. \tag{5.13}$$

Let us denote $b_T = \sqrt{T}\max_{t=1,\dots,T}\|x_t\|_\infty + a$. We evaluate the integral:

$$\begin{aligned}\int_{\mathbb{R}^n} e^{-\tilde{J}(\theta)} d\theta &\ge \int_{\mathbb{R}^n} e^{-(J(\theta_0)+b_T\|\theta-\theta_0\|_1)} d\theta \\ &= e^{-J(\theta_0)} \int_{\mathbb{R}} \dots \int_{\mathbb{R}} e^{-b_T \sum_{i=1}^n |\theta_i-\theta_{i,0}|} d\theta_i \\ &= e^{-J(\theta_0)} \int_{\mathbb{R}} \dots \int_{\mathbb{R}} \prod_{i=1}^n e^{-b_T|\theta_i-\theta_{i,0}|} d\theta_i \\ &= e^{-J(\theta_0)} \prod_{i=1}^n \int_{\mathbb{R}} e^{-b_T|\theta_i-\theta_{i,0}|} d\theta_i = e^{-J(\theta_0)} \left(\frac{2}{b_T}\right)^n. \end{aligned}$$

By putting this expression in (5.6) we obtain the theoretical bound. □

Note that even though we apply WAA for truncated experts (5.3), we achieve the theoretical bound for prediction strategy that competes with a class of experts (5.1).

5.4 Prediction Strategy

A prediction of the WAA (2.30) can be re-written as follows:

$$\gamma_T = \int_\Theta \tilde{\xi}_T(\theta) q^*_{T-1}(\theta) d\theta, \tag{5.14}$$

where

$$q^*_T(\theta) = Z q_T(\theta) = Z \exp\Big(-\frac{1}{\sqrt{T}}\Big(\sum_{\substack{t=1,\dots,T:\\ y_t<\tilde{\xi}_t(\theta)}} (1-\alpha)|y_t - \tilde{\xi}_t(\theta)| + \sum_{\substack{t=1,\dots,T:\\ y_t>\tilde{\xi}_t(\theta)}} \alpha|y_t - \tilde{\xi}_t(\theta)|\Big) - a\|\theta\|_1\Big). \tag{5.15}$$

and Z is the normalising constant ensuring that $\int_\Theta q^*_T(\theta)d\theta = 1$.

Integral (5.14) is a Bayesian mixture, where function $\xi_T(\theta)$ needs to be integrated with respect to the normalized distribution $q^*_T(\theta)$. In order to calculate the integral (5.14), it is possible to use MCMC algorithms. A good introduction of MCMC for Machine Learning is in Andrieu et al. (2003).

We use Metropolis-Hastings algorithm for sampling parameters θ from the posterior distribution $\mathcal{P}$. As a proposal distribution we choose Gaussian distribution $\mathcal{N}(0, \sigma^2)$ with some parameter σ. We start with some initial parameter θ^0 and at each step m we update:

$$\theta^m = \theta^{m-1} + \mathcal{N}(0, \sigma^2),\ m = 1, \dots, M,$$

where M is a maximum number of iterations in MCMC method.

The update parameter θ^m at step m is accepted with probability $\min\left(1, \frac{f_\mathcal{P}(\theta^m)}{f_\mathcal{P}(\theta^{m-1})}\right)$, where $f_\mathcal{P}(\theta)$ is the density function for the distribution $\mathcal{P}$ at point θ. At each step we try to maximize the density function by either accepting or rejecting the new parameters θ. It is common not to calculate the integral until high values of density function $f_\mathcal{P}$ are reached. It is known as 'burn-in' stage of the algorithm. Some values of θ are accepted even when the calculated probability is less than 1, it allows the algorithm to move away from local minimum of the density function. Because Metropolis-Hastings algorithm uses only the ratio of density functions of sampling parameters, it is possible to avoid the calculation of normalising constant Z. We can generate new parameters θ from the unnormalized posterior distribution $q_T(\theta)$ and skip the weights normalization at each step which is more computationally efficient.

At time $t = 0$ the algorithm starts with the initial estimate of the parameters $\theta_0 = 0$.

At each iteration $t > 0$ we start with parameter θ_{t-1}^M calculated at the previous step $t-1$. It allows the algorithm to converge faster to the correct location of the main mass of the distribution.

WAAQR

Parameters: number $M > 0$ of MCMC iterations,
standard deviation $\sigma > 0$,
regularization coefficient $a > 0$

initialize $\theta_0^M := 0 \in \Theta$
define $q_0(\theta) := \exp(-a\|\theta\|_1)$
for $t = 1, 2, \ldots$ do
  $\gamma_t := 0$
  define $q_{t-1}(\theta)$ by (5.15) if $t > 1$
  read $x_t \in \mathbb{R}^n$
  initialize $\theta_t^0 = \theta_{t-1}^M$
  for $m = 1, 2, \ldots, M$ do
    $\theta^* := \theta_t^{m-1} + \mathcal{N}(0, \sigma^2 I)$
    flip a coin with success probability
      $\min\left(1, q_{t-1}(\theta^*)/q_{t-1}(\theta_t^{m-1})\right)$
    if success then
      $\theta_t^m := \theta^*$
    else
      $\theta_t^m := \theta_{t-1}^m$
    end if
    $\gamma_t := \gamma_t + \tilde{\xi}_t(\theta_t^m)$
  end for
  output predictions $\gamma_t = \gamma_t/M$
end for

5.5 Experiments

In this section, we apply the WAA and the WAAQR for prediction of wind and solar power and compare their performance with other predictive models. The dataset is downloaded from Open Power System Data, which provides free and open data platform for power system modelling. The platform contains hourly measurements of geographically aggregated weather data across Europe and time-series of wind and so-

lar power. Our training data are measurements in Austria from January to December 2015, test set contains data from January to July 2016. [1]

5.5.1 WAA

We apply the WAA for three models: Quantile Regression (QR), Quantile Random Forests (QRF), Gradient Boosting Decision Trees (GBDT). QRF gives a non-parametric and accurate way of estimating conditional quantiles instead of the conditional mean for high-dimensional predictor variables (Meinshausen (2006)). Similar to random forests, QRF grows a large number of trees, but takes into account all observations for each leaf of each tree, not just the average. GBDT uses a combination of weak decision trees, which were built iteratively using the negative gradient of a loss function. The final predictor is the weighted combination of these predictors. These models were used in GEFCom 2014 energy forecasting competition on the final leaderboard (Nagya et al. (2016)). The authors of the paper argue that using multiple regressors is often better than using only one, and therefore combine multiple model outputs. They noted that voting was found to be particularly useful for averaging the quantile forecasts of different models.

We propose an alternative approach to combine different models' predictions by using the WAA. We work according to Protocol 1: at each step t before seeing outcome y_t, we output our prediction γ_t according to (2.30). After observing outcome y_t, we update experts' weights according to (2.29).

To build models for wind power forecasting we use wind speed and temperature as explanatory variables. These variables have been extensively used to produce wind power quantile forecasts (Koenker and Bassett (1978)). We train three models QR, QRF and GBDT on training dataset, and then apply the WAA using forecasts of these models on test dataset. We start with equal initial weights of each model and then update their weights according to their current performance. We estimate the constant of the WAA $c = 0.01$ using information about maximum losses on training set.

Figure 5.1 shows weights of each model for different quantiles depending on the current time step. We can see from the graph that for most of quantiles GBDT obtains the largest weights which indicates that it suffers smaller losses compared to other models. However, it changes for $q = 0.95$, where the largest weights are acquired by QR. It shows that sometimes we cannot use the past information to evaluate the best model. The retrospectively best model can perform worse in the future as an underlying nature of data generating can change. In addition, different models can perform better on different quantiles.

[1] The code written in R is available at https://github.com/RaisaDZ/Quantile-Regression.

Tables 5.1, 5.2, 5.3, 5.4 show monthly losses of QR, QRF, GBDT, WAA and Average methods, where Average is a simple average of QR, QRF and GBDT. Figure 5.2 illustrates total losses of these methods. We can see from the tables that performance of models might vary for different months, however the performance of WAA is always close to the performance of the best model. For $q = 0.25$ and $q = 0.5$ the total loss of the WAA is slightly higher than the total loss of GBDT, whereas for $q = 0.75$ and $q = 0.95$ the WAA has the smallest loss. In most cases, the WAA outperforms Average method.

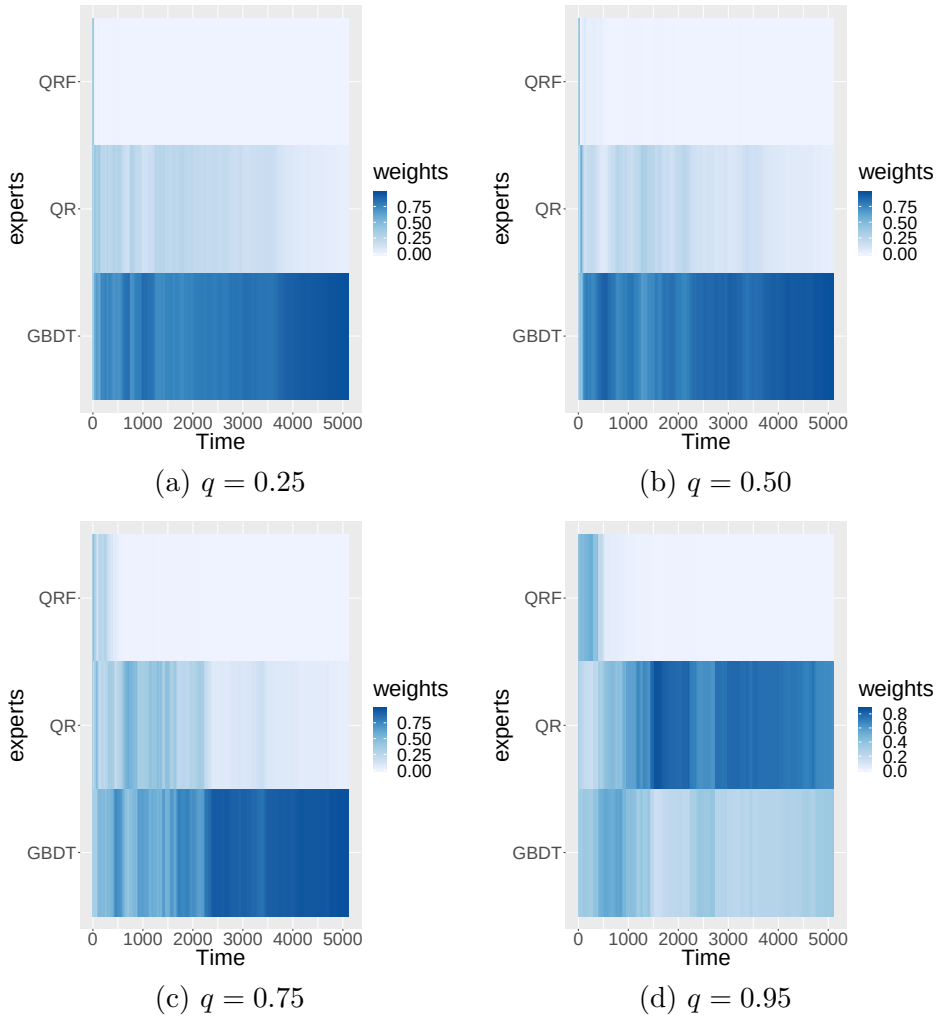
We perform similar experiments for prediction of solar power. We choose measurements of direct and diffuse radiations to be our explanatory variables. In a similar way, QR, QRF and GBDT are trained on training set, and the WAA is applied on test data. Figure 5.3 illustrates weights of models depending on the current step. Opposite to the previous experiments, GBDT has smaller weights compared to other models for $q = 0.25$ and $q = 0.5$. However, for $q = 0.75$ and $q = 0.95$ weights of experts become very close to each other. Therefore, predictions of the WAA should become close to Average method. Figure 5.4 shows total losses of the methods. For $q = 0.25$ and $q = 0.5$ both QR and QRF have small losses compared to GBDT, and the WAA follows their predictions. However, for $q = 0.75$ and $q = 0.95$ it is not clear which model performs better, and predictions of the WAA almost coincide with Average method. It again illustrates that the retrospectively best model could change with time, and one should be cautious about choosing the single retrospectively best model for future forecasts.

Table 5.1: Monthly losses ($\times 10^3$) for wind power, $q = 0.25$.

	QRF	GBDT	QR	Average	WAA
1	87.4	**64.9**	69.3	70.7	65.8
2	92.4	85.8	85.7	**84.8**	85.4
3	90.3	**83.7**	86.3	84.6	84.0
4	81.7	81.2	83.7	**80.7**	81.4
5	88.1	**83.1**	84.7	83.5	83.3
6	41.6	**36.4**	43.9	38.7	36.9
7	57.0	**56.1**	63.0	57.3	56.3
	538.5	**491.2**	516.6	500.3	493.0

5.5.2 WAAQR

In this section, we demonstrate the performance of our algorithm and compare it with quantile regression model. We train QR on training dataset, and apply WAAQR on test set. First, we use training set to choose the parameters of our algorithm. Table



(a) $q = 0.25$ (b) $q = 0.50$

(c) $q = 0.75$ (d) $q = 0.95$

Figure 5.1: Weights update for wind power

5.6 illustrates the acceptance ratio of new sampling parameters of our algorithm for $q = 0.5$. Increasing values of σ results in decreasing acceptance ratios of new sampling parameters θ. With large values of σ we move faster to the area of high values of density function while smaller values of σ can lead to more expensive computations as our algorithm would require more iterations to find the optimal parameters. Figure 5.5 illustrates logarithm of parameters likelihood $w(\theta)$ defined in (5.15) for $a = 0.1$ and $\sigma = 0.5$ and 3.0. We can see from the graphs that for $\sigma = 3.0$ the algorithm reaches maximum value of log-likelihood after around 800 iterations while for $\sigma = 0.5$ it still tries to find maximum value after 1500 iterations. Table 5.7 shows the total losses of WAAQR for different parameters a and σ. We can see that choosing the right parameters is very important as it notably affects the performance of WAAQR. It is

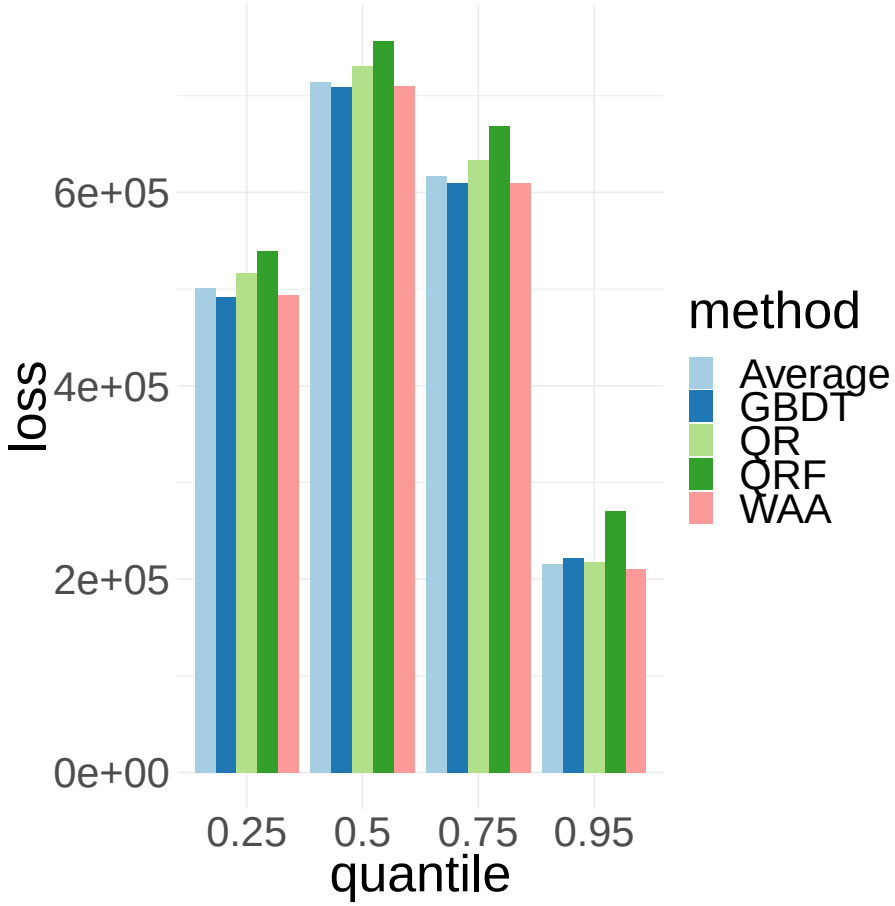


Figure 5.2: Total losses of methods for wind power

important to keep track of acceptance ratio of the algorithm, as high acceptance ratio means that we move too slowly and need more iterations and larger 'burn-in' period to find the optimal parameters.

Now we compare performances of our WAAQR and quantile regression. The parameters of our algorithm are number of iterations $M = 1500$, 'burn-in' stage $M_0 = 300$, regularization parameter $a = 0.1$, and standard deviation $\sigma = 3$. Note that even though we use the prior knowledge to choose the parameters of WAAQR, we start with initial $\theta_0 = 0$ and train our algorithm only on the test set. Figure 5.6 illustrates a difference between cumulative losses of QR and WAAQR. If the difference is greater than zero, our algorithm shows better results compared to QR. For $q = 0.25$ WAAQR shows better performance at the beginning, but after around 1000 iterations its performance becomes worse, and by the end of the period cumulative losses of QR and WAAQR are almost the same. We observe a different picture for $q = 0.5$ and $q = 0.75$: most of the time a difference between cumulative losses is positive, which indicates that WAAQR

Table 5.2: Monthly losses ($\times 10^3$) for wind power, $q = 0.50$.

	QRF	GBDT	QR	Average	WAA
1	113.1	**97.1**	100.3	101.2	98.0
2	123.3	119.1	119.5	**117.8**	118.9
3	121.7	**114.0**	115.9	114.0	114.1
4	119.7	**116.4**	121.6	116.9	117.0
5	127.0	121.2	121.4	121.2	**121.0**
6	65.6	**56.2**	62.5	58.9	56.3
7	86.6	**83.5**	89.7	84.0	83.7
	757.0	**707.5**	730.7	714.0	709.0

Table 5.3: Monthly losses ($\times 10^3$) for wind power, $q = 0.75$.

	QRF	GBDT	QR	Average	WAA
1	95.8	83.2	83.1	84.8	**82.5**
2	103.8	98.8	102.2	**97.8**	98.6
3	101.8	93.7	94.8	**92.8**	93.0
4	108.6	**99.7**	108.6	103.1	100.8
5	109.4	103.0	104.0	**102.0**	102.9
6	60.1	54.4	57.9	55.6	**54.4**
7	88.9	**77.9**	83.2	80.5	78.0
	668.3	610.7	633.9	616.6	**610.1**

performs better than QR.

Figure 5.7 shows predictions of WAAQR and QR with $[25\%, 75\%]$ confidence interval for the first and last 100 steps. We can see from the graph, that initially predictions of WAAQR are very different from predictions of QR. However, by the end of period, predictions of both methods become very close to each other.

One of the disadvantages of WAAQR is that it might perform much worse with non-optimal input parameters of regularization a and standard deviation σ. If no prior knowledge is available, one can start with some reasonable values of input parameters and keep track of the acceptance ratio of new generated θ. If the acceptance ratio is too high it might indicate that the algorithm moves too slowly to the area of high values of the probability function of θ, and standard deviation σ should be increased. Another option is to take very large number of steps and larger 'burn-in' period.

Table 5.4: Monthly losses ($\times 10^3$) for wind power, $q = 0.95$.

	QRF	GBDT	QR	Average	WAA
1	36.5	28.9	30.0	29.7	**28.5**
2	40.7	35.8	32.5	33.1	**32.2**
3	38.3	32.8	28.6	29.3	**28.2**
4	48.3	40.1	40.4	39.0	**38.8**
5	41.2	34.3	33.2	33.5	**32.6**
6	24.5	20.2	20.5	20.3	**20.2**
7	41.0	**29.9**	32.4	31.0	30.6
	270.5	222.1	217.5	216.0	**211.0**

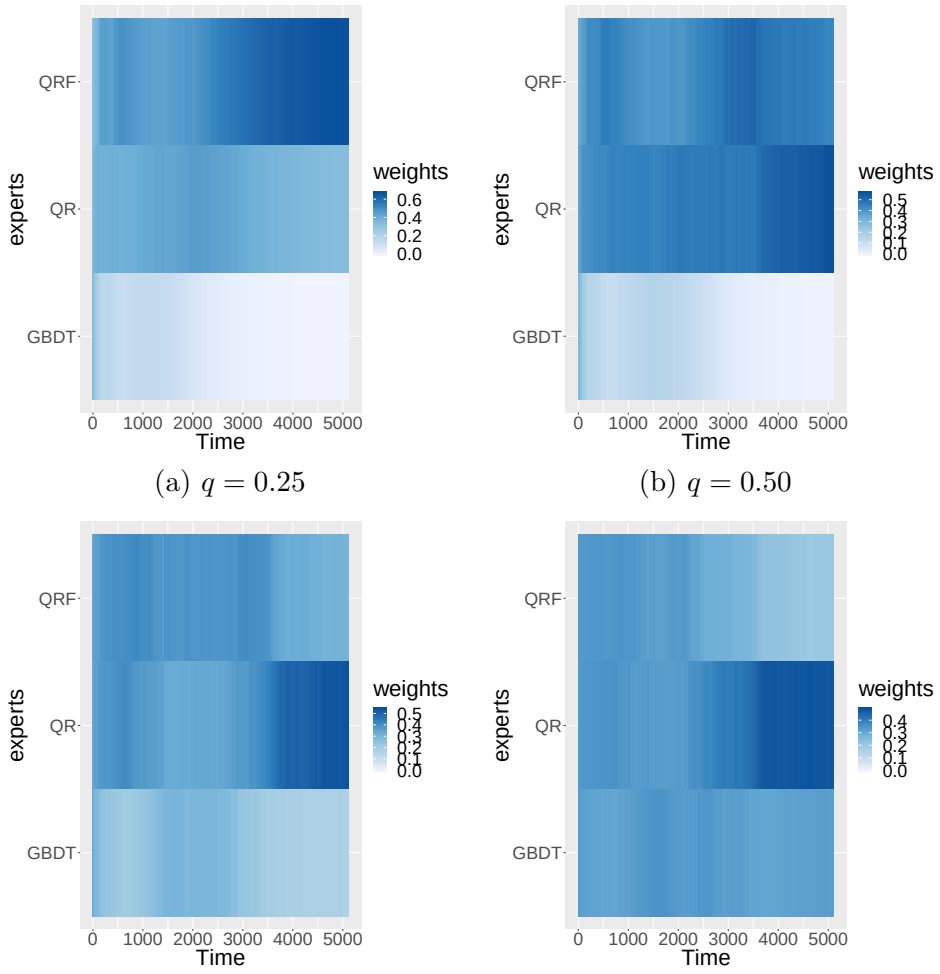
Table 5.5: Total losses ($\times 10^3$) for solar power

q	QRF	GBDT	QR	Average	WAA
0.25	**48.6**	98.3	53.1	63.8	50.1
0.5	70.5	110.7	**68.8**	79.1	69.2
0.75	63.5	67.6	59.3	58.7	**58.0**
0.95	29.2	26.1	23.2	21.0	**20.8**

5.6 Conclusions

We proposed two ways of applying the framework of prediction with expert advice to the problem of probabilistic forecasting of renewable energy. The first approach is to apply WAA with a finite number of models and combine their predictions by updating weights of each model online based on their performance. Experimental results show that WAA performs close or better than the best model in terms of cumulative pinball loss function. It also outperforms the simple average of predictions of models. With this approach we show that it is reasonable to apply WAA for the prediction of quantiles.

Second, we propose a new universal algorithm WAAQR which combines predictions of an infinite pool of quantile regressions. We derive the theoretical bound which guarantees that WAAQR asymptotically performs as well as any quantile regression up to an additive term of the form $C\sqrt{T}\ln T$. Experimental results show that WAAQR can outperform the best quantile regression model that was trained on the historical data.



(a) $q = 0.25$

(b) $q = 0.50$

(c) $q = 0.75$

(d) $q = 0.95$

Figure 5.3: Weights update for solar power

Table 5.6: Acceptance ratio of WAAQR on training set

$a \setminus \sigma$	0.5	1.0	2.0	3.0
0.1	0.533	0.550	0.482	0.375
0.3	0.554	0.545	0.516	0.371
0.5	0.549	0.542	0.510	0.352
1.0	0.548	0.538	0.502	0.343

Table 5.7: Total loss of WAAQR ($\times 10^3$) on training set

$a \setminus \sigma$	0.5	1.0	2.0	3.0
0.1	1821.8	823.5	216.3	28.8
0.3	1806.2	844.9	265.3	62.7
0.5	1815.7	878.5	272.7	92.1
1.0	1810.4	877.5	379.3	116.9

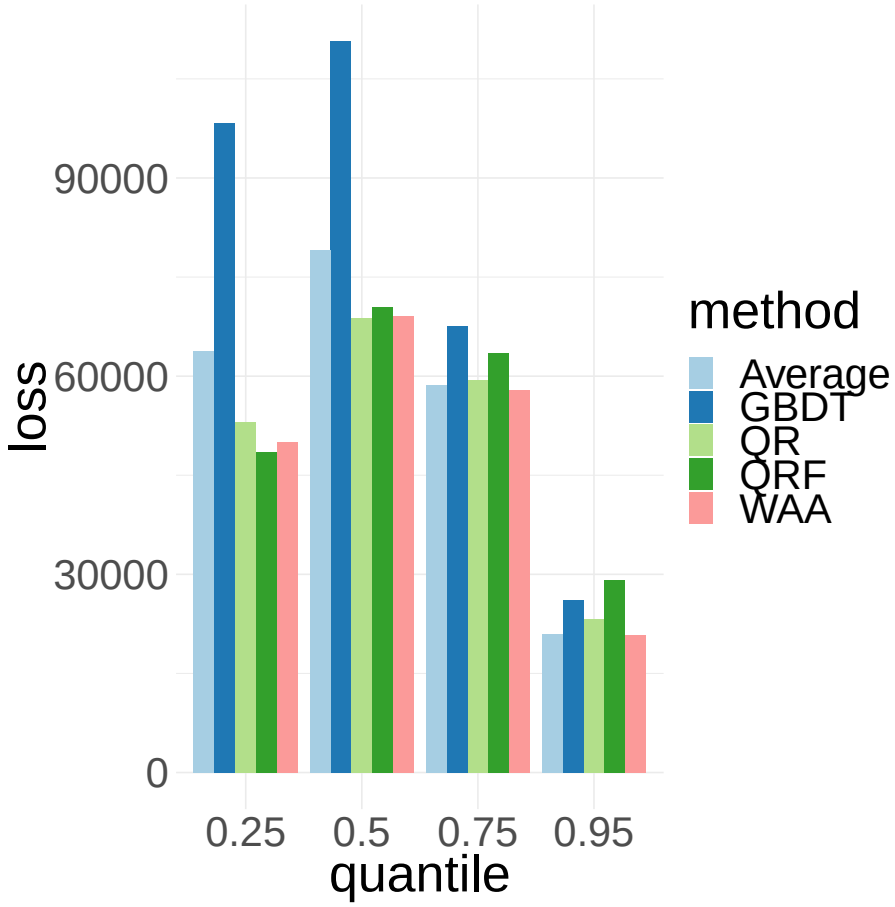
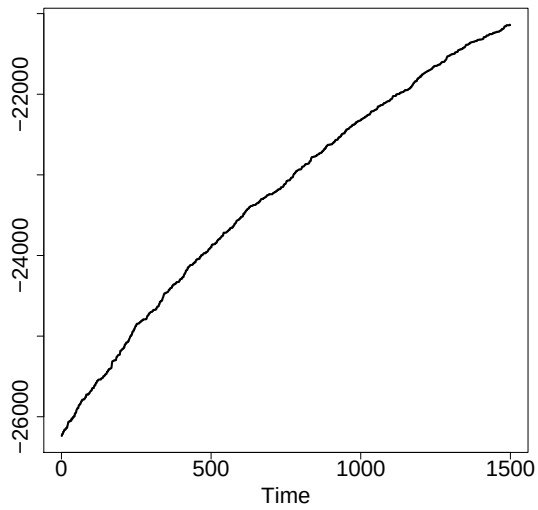
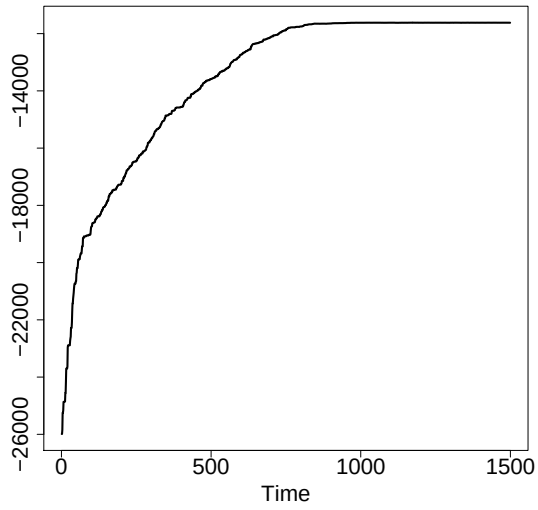


Figure 5.4: Total losses of methods for solar power

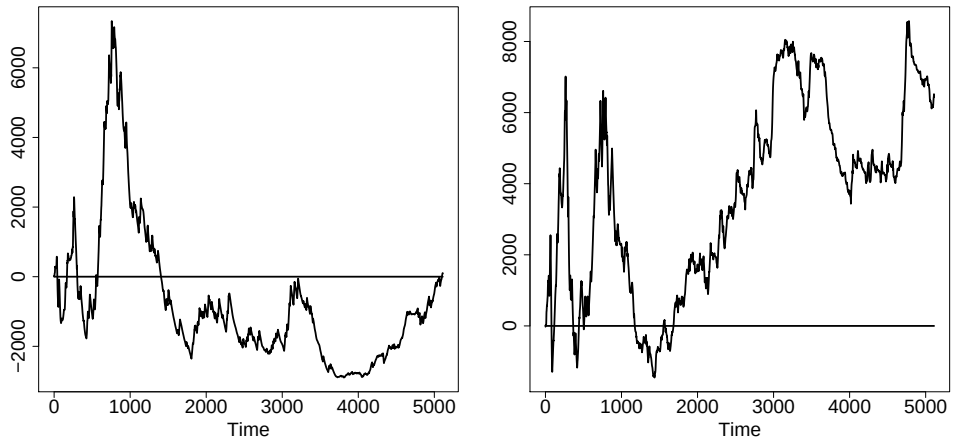


(a) $\sigma = 0.5$

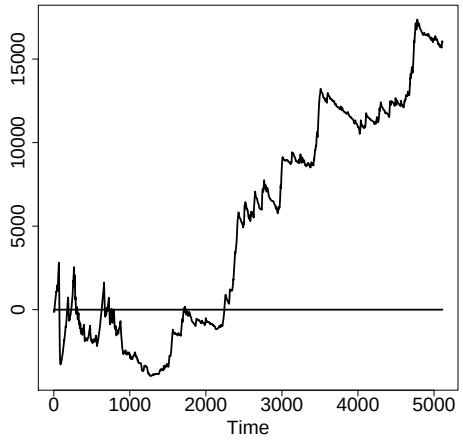


(b) $\sigma = 3.0$

Figure 5.5: Log-likelihood of parameters for $a = 0.1$

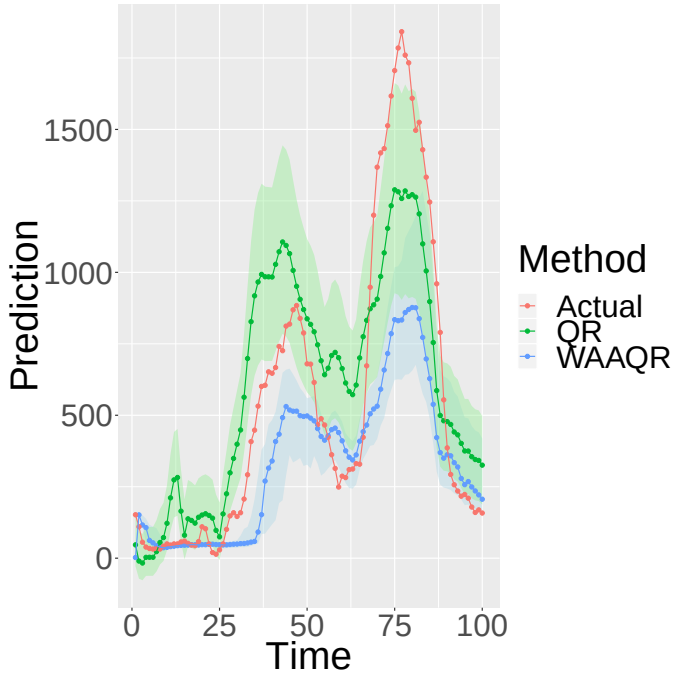


(a) $q = 0.25$ (b) $q = 0.50$

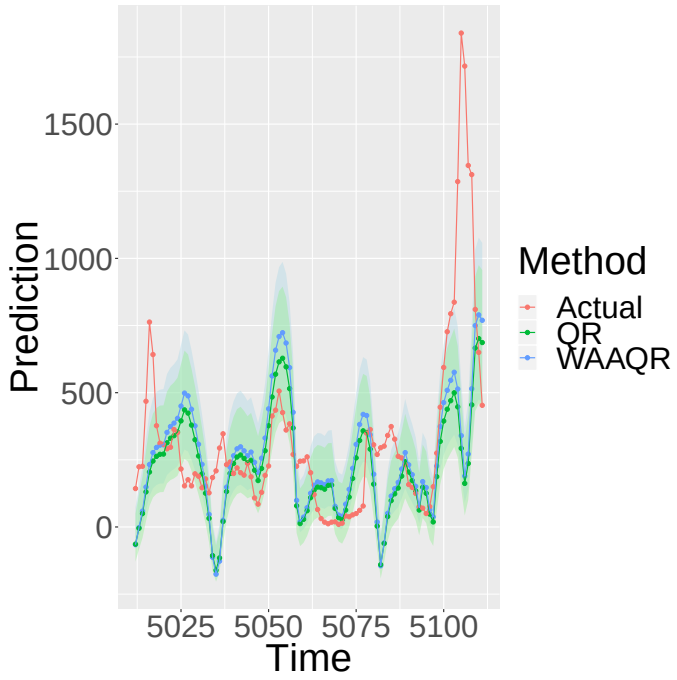


(c) $q = 0.75$

Figure 5.6: Cumulative loss difference between QR and WAAQR



(a) first 100 steps



(b) last 100 steps

Figure 5.7: Predictions with $[25\%, 75\%]$ confidence interval for WAAQR and QR

Chapter 6

Competitive online regression under Continuous Ranked Probability Score

In this chapter, we propose the algorithm that combines point predictions of an infinite pool of linear experts and outputs probability forecasts in the form of cumulative distribution functions. We evaluate the quality of probabilistic prediction by the continuous ranked probability score (CRPS), which is a widely used proper scoring rule. We provide a strategy that allows us to 'track the best expert' and derive the theoretical bound on the discounted loss of the strategy.

6.1 Introduction

One of the frequently used proper scoring rules that evaluates the quality of probabilistic predictions is the continuous ranked probability score (CRPS). The CRPS provides a direct way of comparing point forecasts and probabilistic forecasts. Weighted versions of the CRPS were introduced in Matheson and Winkler (1976).

In this chapter, we look for performance guarantees relative to other predictive models. We propose an algorithm that combines point forecasts of an infinite pool of linear regressions and provides probabilistic predictions in the form of cumulative distribution functions. The proposed strategy allows us to 'track the best expert', and the theoretical bound on the discounted loss of the strategy is derived.

Our approach uses the Aggregating Algorithm, which gives a guarantee ensuring that the learner's loss is as small as best expert's loss plus a constant in a case of mixable loss functions and a finitely many experts. In a recent paper V'yugin and Trunov (2019)

it is shown that the CRPS is a mixable loss function, and the theoretical bound for the case of a finite number of experts is derived. In this chapter, we consider the same problem setting, but we choose a pool of linear regressions to be our experts. We consider the case of discounted loss along the lines of Chernov and Zhdanov (2010). Discounting allows us to give less importance to older losses, which is an important property for practical applications. In Freund and Hsu (2008) the authors noticed that in the context of prediction with expert advice, the discounted loss provides an alternative to 'tracking the best expert' framework of Herbster and Warmuth (1998). Indeed, if the best expert changes after some steps, an algorithm that competes under discounted loss will not take into account small losses of the old best expert because they are strongly discounted, and will switch to track the new best expert.

Our prediction strategy mixes an infinite pool of linear experts in a way which is similar to Aggregating Algorithm for Regression for the case of linear experts under the square loss. The generalisation for the case of discounted square loss for linear regression was proposed in Chernov and Zhdanov (2010).

We perform experiments on a synthetic dataset and apply our algorithm for the prediction of solar power. We compare the performance of our algorithm with linear regression and quantile regression. Quantile regression has been extensively used to produce renewable energy power quantile forecasts (Koenker and Bassett (1978)) and in probabilistic energy forecasting competitions (Nagya et al. (2016)). Our prediction algorithm uses Markov chain Monte Carlo (MCMC) method in a way which is similar to the algorithm WAAQR introduced in Section 5.4. Experiments show that our algorithm requires some time for training, however by the end of the period the performance of our algorithm becomes close to the performance of the retrospectively best quantile regression.

6.2 Framework

Suppose that F is the distribution function $F = F_X$ of some random variable X. Then

(a) $F : \mathbb{R} \to [0, 1]$, F is non-decreasing (that is $x \leq y \Rightarrow F(x) \leq F(y)$),

(b) $\lim_{x\to+\infty} F(x) = 1$, $\lim_{x\to-\infty} F(x) = 0$,

(c) F is right-continuous.

(See Section II.6 in Loève (1977) or Section 3.10 in Williams (1991)).

We take this class of functions to be our decision space Γ and take the space of

outcomes $\Omega = [A, B] \subset \mathbb{R}$. We measure loss by the CRPS loss function:

$$\lambda(y, F) = \int_A^B (F(u) - 1_{u \geq y})^2 \, du. \tag{6.1}$$

CRPS loss function generalizes the absolute error; it reduces to the absolute error if F is a point forecast. Indeed, if $F_z(u) = 1_{u \geq z}$, then $\lambda(y, F_z) = |y - z|$.

Learner works according to the Protocol 3 defined in Section 2.4.

We want to find a strategy which is capable of competing with all prediction strategies $\theta \in \mathbb{R}^n$ that at step t outputs:

$$\xi_t(\theta) = F_t^\theta, \tag{6.2}$$

where

$$F_t^\theta(u) = 1_{u \geq x_t' \theta}, \ u \in \mathbb{R} \tag{6.3}$$

and the loss of the strategy θ is:

$$\lambda(y, \xi_t(\theta)) = |y - x_t' \theta|. \tag{6.4}$$

The space of prediction strategies is the whole $\mathbb{R}$, not the interval $[A, B]$. Note that in the case of the absolute loss the capabilities of prediction with expert advice are restricted. The absolute loss is not mixable and the regret term should have the order of $\sqrt{T}$ (see Kalnishkan and Vyugin (2008)).

The cumulative losses of the learner are discounted with a factor $\alpha_t \in (0, 1]$ at each step. If L_{T-1} is the discounted cumulative loss of the learner at step $T - 1$, then the discounted cumulative loss of the learner at step T is defined in (2.20):

$$L_T := \alpha_{T-1} L_{T-1} + \lambda_T(y_T, \gamma_T) = \sum_{t=1}^{T-1} \left(\prod_{j=t}^{T-1} \alpha_j \right) \lambda_t(y_t, \gamma_t) + \lambda_T(y_T, \gamma_T).$$

If L_{T-1}^θ is the discounted cumulative loss of the prediction strategy θ at the step $T - 1$, then the discounted cumulative loss of the prediction strategy θ at the step T is defined in (2.21):

$$L_T^\theta := \alpha_{T-1} L_{T-1}^\theta + \lambda_T(y_T, \xi_T(\theta)) = \sum_{t=1}^{T-1} \left(\prod_{j=t}^{T-1} \alpha_j \right) \lambda_t(y_t, \xi_t(\theta)) + \lambda_T(y_T, \xi_T(\theta)).$$

6.3 Theoretical Bounds

Theorem 6.3.1. *Let $a > 0$, $y \in \Omega = [A, B]$, $\gamma \in \Gamma$. There exists a prediction strategy for Learner such that for every positive integer T, every sequence of outcomes of length T, every sequence $\alpha_t \in (0, 1]$, $t = 1, \dots, T$, and every $\theta \in \mathbb{R}^n$ the discounted cumulative losses L_T of Learner and L_T^θ of expert θ satisfy*

$$L_T \le L_T^\theta + a\|\theta\|_1 + \frac{n(B-A)}{2} \ln\left(1 + \frac{\sum_{t=1}^T w_{t,T}}{a} \max_{t=1,\dots,T} \|x_t\|_\infty\right), \tag{6.5}$$

where $w_{t,T} = \prod_{j=t}^{T-1} \alpha_j$.

The theorem states that the algorithm predicts as well as the best 'switching' prediction strategy, defined in (6.2), up to an additive regret of the form $C \ln T$ in terms of the discounted cumulative loss.

It is easier to see that the regret term is of the form $C \ln T$ for the undiscounted case, when we have:

Corollary 3. *Let $a > 0$, $y \in \Omega = [A, B]$ and $\gamma \in \Gamma$. There exists a prediction strategy for Learner such that for every positive integer T, every sequence of outcomes of length T, and every $\theta \in \mathbb{R}^n$ the cumulative losses L_T of Learner and L_T^θ of expert θ satisfy*

$$L_T \le L_T^\theta + a\|\theta\|_1 + \frac{n(B-A)}{2} \ln\left(1 + \frac{T}{a} \max_{t=1,\dots,T} \|x_t\|_\infty\right). \tag{6.6}$$

6.4 Prediction Strategy

Let $\tilde{\mathcal{G}}$ be the square-loss game with the outcome space $\tilde{\Omega} = \{0, 1\}$, prediction space $\tilde{\Gamma} = [0, 1]$, and the square loss function $\tilde{\lambda}(y, \gamma) = (y - \gamma)^2$. We consider the game $\mathcal{G}$ as the 'limit' of a sequence of games $\tilde{\mathcal{G}}$ with the vector-valued forecasts. For $K \in \mathcal{N}$ we take points $z_k = A + k\frac{B-A}{K}$, $k = 0, 1, \dots, K$ and approximate any function $F \in \Gamma$ by a piecewise-constant function F_K defined by $F_K(u) = F(z_k)$ for any $u \in [z_k, z_{k+1})$, $k = 0, 1, \dots, K-1$. For the game $\tilde{\mathcal{G}}$ the learner's prediction is defined by (2.17):

$$\gamma_t = \frac{1}{2} - \frac{g_t(1) - g_t(0)}{2}, \ t = 1, \dots, T. \tag{6.7}$$

Let $F_t^\theta \in \Gamma$ be a set of predictions parameterised by $\theta \in \Theta$ at time t. Since the game $\tilde{\mathcal{G}}$ is 2-mixable (Lemma 2.2.4), we obtain the learner's prediction in the game $\tilde{\mathcal{G}}$

by putting the expression for generalised prediction of AAD (2.23) in (6.7):

$$F_t(z_k) = \frac{1}{2} - \frac{1}{4} \ln \frac{\int_{\theta\in\Theta} P_0(d\theta) \left(\tilde{P}_{t-1}(\theta)\right)^{\alpha_{t-1}} e^{-2(F_t^\theta(z_k))^2}}{\int_{\theta\in\Theta} P_0(d\theta) \left(\tilde{P}_{t-1}(\theta)\right)^{\alpha_{t-1}} e^{-2(1-F_t^\theta(z_k))^2}}, \ k = 0, 1, \ldots, K-1. \quad (6.8)$$

By letting $K \to +\infty$ in (6.8), we obtain the expression for computing the learner's forecast:

$$\gamma_t = F_t, \quad (6.9)$$

where

$$F_t(u) = \frac{1}{2} - \frac{1}{4} \ln \frac{\int_{\theta\in\Theta} P_0(d\theta) \left(\tilde{P}_{t-1}(\theta)\right)^{\alpha_{t-1}} e^{-2(F_t^\theta(u))^2}}{\int_{\theta\in\Theta} P_0(d\theta) \left(\tilde{P}_{t-1}(\theta)\right)^{\alpha_{t-1}} e^{-2(1-F_t^\theta(u))^2}}, \ u \in [A, B]. \quad (6.10)$$

We choose the initial distribution of the parameters for some $a > 0$:

$$P_0(d\theta) = \left(\frac{a\eta}{2}\right)^n e^{-a\eta\|\theta\|_1} d\theta, \quad (6.11)$$

where $\theta \in \mathbb{R}^n$.

Then the learner's prediction (6.10) can be re-written as follows:

$$F_t(u) = \frac{1}{2} - \frac{1}{4} \ln \frac{\int_{\theta\in\Theta} q_t^*(\theta) e^{-2(F_t^\theta(u))^2} d\theta}{\int_{\theta\in\Theta} q_t^*(\theta) e^{-2(1-F_t^\theta(u))^2} d\theta}, \quad (6.12)$$

where

$$q_T^*(\theta) = Z q_T(\theta) = Z \exp\left(-\eta \sum_{t=1}^{T-1} \left(\prod_{j=t}^{T-1} \alpha_j\right) |y_t - x_t'\theta| - a\eta\|\theta\|_1\right), \quad (6.13)$$

and Z is the normalising constant ensuring that $\int_\Theta q_T^*(\theta) d\theta = 1$.

Since

$$e^{-2} \le \int_{\theta\in\Theta} e^{-2(F^\theta(u))^2} q^*(\theta) d\theta, \int_{\theta\in\Theta} e^{-2(1-F^\theta(u))^2} q^*(\theta) d\theta \le 1 \quad (6.14)$$

we get $0 \le F(u) \le 1$. Since $F^\theta(u)$ is non-decreasing in u, our $F(u)$ is non-decreasing too. By the Monotone Convergence Theorem (Theorem 5.3 in Williams (1991)) if

$u \downarrow u_0$ then

$$\int_{\theta\in\Theta} e^{-2(F^\theta(u))^2} q^*(\theta)d\theta \downarrow \int_{\theta\in\Theta} e^{-2(F^\theta(u_0))^2} q^*(\theta)d\theta$$
$$\int_{\theta\in\Theta} e^{-2(1-F^\theta(u))^2} q^*(\theta)d\theta \uparrow \int_{\theta\in\Theta} e^{-2(1-F_t^\theta(u_0))^2} q^*(\theta)d\theta.$$

Therefore, F is right-continuous. We have shown that $F \in \Gamma$.

For completeness, we include the following lemma from V'yugin and Trunov (2019) and go through the details of the proof.

Lemma 6.4.1. *Game $\mathcal{G}$ where the space of outcomes $\Omega = [A, B]$ and decision space Γ contains probability distribution functions $F : [A, B] \to [0, 1]$, and CRPS loss function (6.1) is $\frac{2}{B-A}$-mixable.*

Proof. We need to show that prediction (6.12) satisfies (2.4) for $C = 1$, that is

$$\lambda(y, F) \le -\frac{1}{\eta} \ln \int_\Theta e^{-\eta\lambda(y,F^\theta)} P(d\theta) \tag{6.15}$$

for all $y \in [A, B]$.

The CRPS loss function can be represented as:

$$\begin{aligned}\lambda(y, F_K) = \sum_{k=0}^{j-1} \int_{z_k}^{z_{k+1}} F_K^2(u)du + \int_{z_j}^{y} F_K^2(u)du + \int_{y}^{z_{j+1}} (1 - F_K(u))^2 \, du \\ + \sum_{k=j+1}^{K-1} \int_{z_k}^{z_{k+1}} (1 - F_K(u))^2 \, du = \frac{B-A}{K} \sum_{k=0}^{j-1} F^2(z_k) + (y - z_j)F^2(z_j) \\ + (z_{j+1} - y) \left(1 - F(z_j)\right)^2 + \frac{B-A}{K} \sum_{k=j+1}^{K-1} \left(1 - F(z_k)\right)^2, \end{aligned} \tag{6.16}$$

where $y \in [z_j, z_{j+1})$.

An outcome y can be identified with a vector $\omega = (\omega_0^y, \omega_1^y, \ldots, \omega_{K-1}^y)$, where $\omega_k^y = 1_{z_{k+1}\ge y} \in \{0, 1\}$ for $k = 0, 1, \ldots, K - 1$. Let us define the loss function $\hat{\lambda}(y, F_K)$ by

$$\hat{\lambda}(y, F_K) = \frac{B-A}{K} \sum_{i=0}^{K-1} \left(\omega_k^y - F(z_k)\right)^2 = \frac{B-A}{K} \left(\sum_{k=0}^{j-1} F^2(z_k) + \sum_{k=j}^{K-1} (1 - F(z_k))^2 \right), \tag{6.17}$$

where $y \in [z_j, z_{j+1})$. We get:

$$|\lambda(y, F_K) - \hat{\lambda}(y, F_K)| = (y - z_j)|F^2(z_j) - (1 - F(z_j))^2| \\ = (y - z_j)|2F(z_j) - 1| \le \frac{B - A}{K}, \quad (6.18)$$

where $y \in [z_j, z_{j+1}),\ j = 0, 1, \dots, K-1$.

Consider the game $\tilde{\mathfrak{G}}^K$ with the outcome and prediction spaces given by the Cartesian products $\tilde{\Omega}^K$ and $\tilde{\Gamma}^K$ and the loss function $\sum_{k=1}^K \tilde{\lambda}(\omega_k, \gamma_k)$. By Theorem 3.3.3, the game $\tilde{\mathfrak{G}}^K$ is $\frac{2}{K}$-mixable. For the experts' predictions $(\gamma_0^\theta, \dots, \gamma_{K-1}^\theta)$ $= (F^\theta(z_0), \dots, F^\theta(z_{K-1}))$, $\theta \in \Theta$, the learner's predictions $(F(z_0), \dots, F(z_{K-1}))$ satisfy

$$\frac{1}{K}\sum_{k=0}^{K-1}(\omega_k - F(z_k))^2 \le -\frac{1}{2}\ln\int_\Theta e^{-\frac{2}{K}\left(\sum_{k=0}^{K-1}\left(\omega_k - F^\theta(z_k)\right)^2\right)} P(d\theta)$$

for all $\omega_k \in [0, 1]$, $k = 0, 1, \dots, K-1$ including ω_k^y, $y \in [A, B]$. In other terms, we get

$$\frac{1}{B - A}\hat{\lambda}(y, F_K) \le -\frac{1}{2}\ln\int_\Theta e^{-\frac{2}{B-A}\hat{\lambda}(y, F_K^\theta)} P(d\theta).$$

By using inequality (6.18), we have:

$$\lambda(y, F_K) \le -\frac{B - A}{2}\ln\int_\Theta e^{-\frac{2}{B-A}\lambda(y, F_K^\theta)} P(d\theta) + 2\frac{B - A}{K}. \quad (6.19)$$

Now it remains to show that losses $\lambda(y, F_K)$ and $\lambda(y, F)$ do not differ much. Since, by construction, $F(u) \ge F_K(u)$, we get

$$|\lambda(y, F) - \lambda(y, F_K)| \le \int_A^y \left(F^2(u) - F_K^2(u)\right) du + \int_y^B \left((1 - F_K(u))^2 - (1 - F(u))^2\right) du.$$

The first integral can be upper bounded as

$$\int_A^y \left(F^2(u) - F_K^2(u)\right) du = \sum_{k=0}^{j-1}\int_{z_k}^{z_{k+1}} \left(F^2(u) - F_K^2(u)\right) du + \int_{z_j}^y \left(F^2(u) - F_K^2(u)\right) du \\ \le \frac{B - A}{K}\sum_{k=0}^{j}\left(F^2(z_{k+1}) - F^2(z_k)\right) = \frac{B - A}{K}\left(F^2(z_{j+1}) - F^2(A)\right) \\ \le \frac{B - A}{K}\left(F^2(B) - F^2(A)\right) \le \frac{B - A}{K}.$$

By doing the same for the second integral, we get

$$|\lambda(y, F) - \lambda(y, F_K)| \leq 2\frac{B - A}{K}. \tag{6.20}$$

Now inequality (6.15) follows from (6.19) by letting $K \to +\infty$. □

We use Metropolis-Hastings algorithm to calculate integrals in (6.12) in the same way as it is described in Section 5.4 for WAAQR. We generate parameters θ from the unnormalized posterior distribution $q_T(\theta)$ (6.13) and avoid the weights normalization at each step which is more computationally efficient. The pseudo-code of the algorithm is given below:

Algorithm

Parameters: number $M > 0$ of MCMC iterations,
standard deviation $\sigma > 0$,
regularization coefficient $a > 0$,
prediction interval $[A, B]$.

$\eta := \frac{2}{B-A}$
initialize $\theta_0^M := 0 \in \Theta$
define $q_0(\theta) := \exp(-a\eta\|\theta\|_1)$
for $t = 1, 2, \ldots$ do
  $\gamma_t^0 := 0,\ \gamma_t^1 := 0$
  define $q_{t-1}(\theta)$ by (6.13) if $t > 1$
  read $x_t \in \mathbb{R}^n$
  initialize $\theta_t^0 = \theta_{t-1}^M$
  for $m = 1, 2, \ldots, M$ do
    $\theta^* := \theta_t^{m-1} + \mathcal{N}(0, \sigma^2 I)$
    flip a coin with success probability
      $\min\left(1, q_{t-1}(\theta^*)/q_{t-1}(\theta_t^{m-1})\right)$
    if success then
      $\theta_t^m := \theta^*$
    else
      $\theta_t^m := \theta_{t-1}^m$
    end if
    $\gamma_t^0 := \gamma_t^0 + \exp\left(-2\left(F_t^{\theta_t^m}(u)\right)^2\right)$
    $\gamma_t^1 := \gamma_t^1 + \exp\left(-2\left(1 - F_t^{\theta_t^m}(u)\right)^2\right)$
  end for

output predictions $\gamma_t = \frac{1}{2} - \frac{1}{4}\ln\frac{\gamma_t^0}{\gamma_t^1}$
end for

6.5 Proof of Theoretical Bounds

In this section, we provide the proof of Theorem 6.3.1. Applying Lemma 2.4.2 for initial distribution (6.11) we obtain

$$L_T \leq -\frac{1}{\eta}\ln\left(\left(\frac{a\eta}{2}\right)^n \int_\Theta e^{-\eta J(\theta)} d\theta\right), \tag{6.21}$$

where

$$J(\theta) := \sum_{t=1}^{T} w_{t,T}|x_t'\theta - y_t| + a\|\theta\|_1$$

and

$$w_{t,T} = \prod_{j=t}^{T-1} \alpha_j, \;\; w_{T,T} = 1.$$

For all $\theta, \theta_0 \in \mathbb{R}^n$ we have:

$$\begin{aligned}\sum_{t=1}^{T} w_{t,T}|x_t'\theta - y_t| &\leq \sum_{t=1}^{T} w_{t,T}|x_t'\theta_0 - y_t| + \sum_{t=1}^{T} w_{t,T}|x_t'\theta - x_t'\theta_0| \\ &\leq \sum_{t=1}^{T} w_{t,T}|x_t'\theta_0 - y_t| + \sum_{t=1}^{T} w_{t,T} \max_{t=1,\ldots,T}\|x_t\|_\infty \|\theta - \theta_0\|_1.\end{aligned}$$

Then, we have:

$$\begin{aligned}J(\theta) &\leq J(\theta_0) + \sum_{t=1}^{T} w_{t,T} \max_{t=1,\ldots,T}\|x_t\|_\infty |\theta - \theta_0| + a(\|\theta\|_1 - \|\theta_0\|_1) \\ &\leq J(\theta_0) + \max_{t=1,\ldots,T}\|x_t\|_\infty \sum_{t=1}^{T} w_{t,T}\|\theta - \theta_0\|_1 + a\|\theta - \theta_0\|_1 \\ &= J(\theta_0) + \left(\max_{t=1,\ldots,T}\|x_t\|_\infty \sum_{t=1}^{T} w_{t,T} + a\right)\|\theta - \theta_0\|_1. \end{aligned} \tag{6.22}$$

Let us denote $b_T = \max_{t=1,\dots,T} \|x_t\|_\infty \sum_{t=1}^T w_{t,T} + a$. We evaluate the integral

$$\int_\Theta e^{-\eta J(\theta)} d\theta \geq \int_{\mathbb{R}^n} e^{-\eta(J(\theta_0)+b_T\|\theta-\theta_0\|_1)} d\theta$$
$$= e^{-\eta J(\theta_0)} \int_{\mathbb{R}} \dots \int_{\mathbb{R}} e^{-\eta b_T \sum_{i=1}^n |\theta_i-\theta_{i,0}|} d\theta_i = e^{-\eta J(\theta_0)} \int_{\mathbb{R}} \dots \int_{\mathbb{R}} \prod_{i=1}^n e^{-\eta b_T|\theta_i-\theta_{i,0}|} d\theta_i$$
$$= e^{-\eta J(\theta_0)} \prod_{i=1}^n \int_{\mathbb{R}} e^{-\eta b_T|\theta_i-\theta_{i,0}|} d\theta_i = e^{-\eta J(\theta_0)} \left(\frac{2}{\eta b_T}\right)^n.$$

By putting this expression in (6.21) we have

$$L_T \leq J(\theta_0) - \frac{1}{\eta} \ln\left(\left(\frac{a\eta}{2}\right)^n \left(\frac{2}{\eta b_T}\right)^n\right)$$
$$= L_T^{\theta_0} + a\|\theta_0\|_1 + \frac{n}{\eta} \ln\left(1 + \frac{\sum_{t=1}^T w_{t,T}}{a} \max_t \|x_t\|_\infty\right).$$

By putting $\eta = \frac{2}{B-A}$ from Lemma 6.4.1 we obtain the theoretical bound (6.5).

6.6 Experiments

In this section, we apply our proposed algorithm on synthetic data and solar power data, and compare its performance with other predictive models. The solar power dataset is downloaded from Open Power System Data which provides free and open data platform for power system modelling. The platform contains hourly measurements of geographically aggregated weather data across Europe and time-series of solar power. Our training data are measurements in Austria from January to December 2015, test set contains data from January to April 2016. [1]

6.6.1 Synthetic Dataset

We apply our algorithm on synthetic datasets. The first dataset is generated from the linear model $y = 2x - 1 + \epsilon$, where $\epsilon \in \mathcal{N}(0, 0.001)$ and feature x is generated from normal distribution $\mathcal{N}(0.75, 0.05)$. Figure 6.1 illustrates the generated dataset which contains 1000 observations. We divide our data in a way that it has half of its observations in training and test datasets. First, we will run our algorithm and train the linear regression on training dataset and compare their performance. From Figure 6.1 we can see that the dataset is almost perfectly linear; and the linear regression model, trained on training dataset, has $R^2 = 0.9999$ on the test data. We run our algorithm for

[1] The code written in R is available at https://github.com/RaisaDZ/CRPS.

the number of MCMC iterations $M = 1500$ and 'burn-in' period $M_0 = 300$ for different parameters of regularization a and standard deviation σ. For this example, we pick our parameters of regularization $a = 0.5$, standard deviation $\sigma = 0.1$, and we do not discount our losses $\alpha_t = 1$ for $t = 1, \dots, T$. Figure 6.3 shows the difference between cumulative losses of the linear regression and our algorithm $L_T^{\theta^*} - L_T$ on test dataset, where θ^* was obtained by linear regression model on training dataset. We also plot the theoretical bound for our algorithm. The initial large gap corresponds to the value $-a\|\theta^*\|_1$, which gives the initial start to Learner on expert θ^*. As time increases, we add an additional value $-\frac{n(B-A)}{2} \ln\left(1 + \frac{T}{a} \max\|x_t\|_\infty\right)$ to the bound. We can see from the graph that initially the loss difference is decreasing fast which means that loss of our algorithm becomes larger compared to the loss of linear regression model. The initial start $-a\|\theta^*\|_1$ gives us some time for training. After the initial training time passes, the difference between cumulative losses becomes smoother and behaves in a similar way with the theoretical bound of our algorithm which is decreasing logarithmically with the number of steps. Figure 6.4 illustrates the difference between cumulative losses of the quantile regression and our algorithm which behaves in a similar way with the previous graph. Total loss of our algorithm on the test dataset is 3.05. It is much larger than the loss of the best linear regression model which is equal to 0.42, and the loss of the quantile regression which is equal to 0.30. It is not surprising as the dataset is almost perfectly linear, and our algorithm makes a large loss during the initial training. However, the theoretical bound of our algorithm is not violated.

The second synthetic dataset is similar with the previous one, but the slope of the model slowly changes with time $y_t = (2 + 0.00005t)x_t - 1 + \epsilon,\ t = 1, \dots, T$. Figure 6.2 illustrates the generated dataset. We use the same parameters of our algorithm as in the previous example, but we add exponential discounting $\alpha_t = 0.999,\ t = 1, \dots, T$. The dataset still looks linear; and the linear regression model, trained on training dataset, has $R^2 = 0.9681$ on the test data. Figure 6.5 shows the difference between different competitors and our algorithm. We can see from the graph that after around 50 iterations the loss difference starts to increase which means that our algorithm starts to perform better than other models. At the end of the period total loss of the best linear regression (LM) is 9.32, loss of the quantile regression trained on training set (QR) is 7.75, loss of the quantile regression trained online (QR online) is 4.66. Total loss of our algorithm is equal 4.55, which is slightly lower than the total loss of the quantile regression trained online.

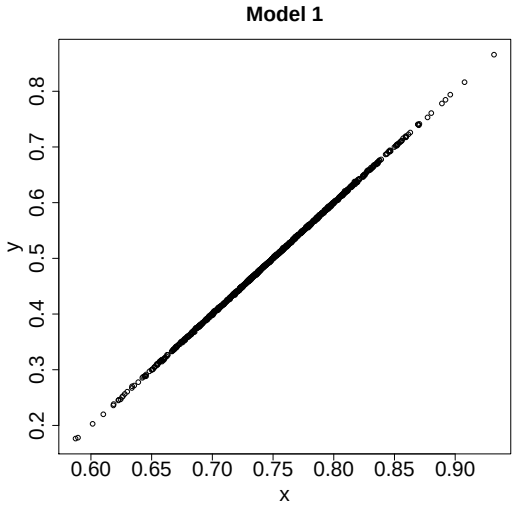


Figure 6.1: First dataset

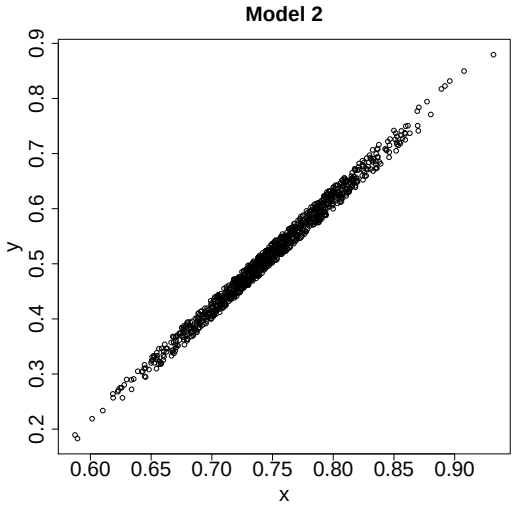


Figure 6.2: Second dataset

6.6.2 Solar Power Dataset

We perform similar experiments for prediction of solar power. We choose measurements of direct and diffuse radiations to be our explanatory variables. Figures 6.6, 6.7 show the dependence of solar power on explanatory variables on the training set. We can see that there is a linear dependence between predicted and explanatory variables. The correlation between solar power and direct radiation is equal to 0.88, whereas the correlation between solar power and diffuse radiation is equal to 0.74. First, similar to the previous experiments, we fit linear regression on the training set. The linear regression seems to perform well on this dataset; it has $R^2 = 0.8929$ on the test set.

Now we run our algorithm for the number of MCMC iterations $M = 1500$ and 'burn-in' period $M_0 = 300$ for different parameters of regularization a and standard deviation σ. Table 6.1 shows the acceptance ratio of new generated parameters θ for different parameters a and σ. We can notice from the table that standard deviation σ affects the acceptance ratio quite a lot, whereas regularization parameter a has a little affect. Figure 6.8 shows the difference between cumulative losses of the best linear regression trained on the training set and our algorithm, and Figure 6.9 shows the difference between cumulative losses of the best quantile regression trained on the training set and our algorithm. We can see from the graphs that we need a little time to outperform the linear regression model, but our algorithm performs much worse than the quantile regression as the difference of cumulative losses decreases fast. However, after around 2000 steps the difference of cumulative losses stabilizes and becomes more

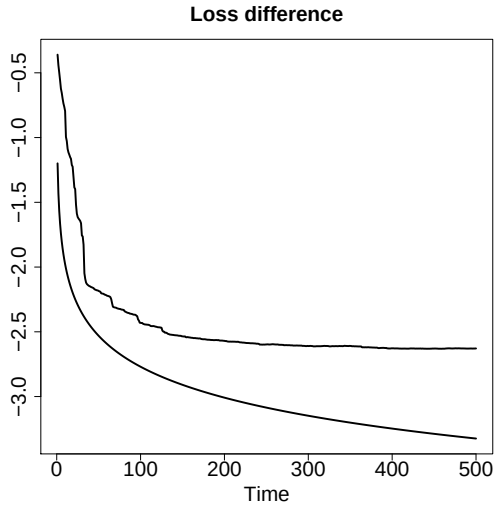


Figure 6.3: Loss difference between the best linear regression and our algorithm

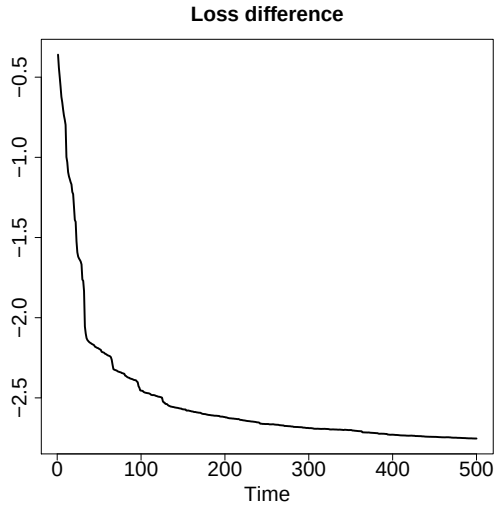


Figure 6.4: Loss difference between the best quantile regression and our algorithm

'flattened' which indicates that the performance of our algorithm becomes close to the performance of the quantile regression.

Figure 6.10 shows predictions of our algorithm and quantile regression (QR) with $[25\%, 75\%]$ confidence interval for the first 100 steps and after 1000 steps. We can see from the graph, that initially predictions of our algorithm are very different from predictions of QR. However, after the initial training of our algorithm, predictions of both methods become very close to each other.

Table 6.1: Acceptance ratio of new generated parameters

$a \setminus \sigma$	0.01	0.02	0.03	0.05	0.10
0.1	0.858	0.602	0.410	0.214	0.070
0.5	0.857	0.602	0.410	0.214	0.069
1.0	0.858	0.601	0.409	0.215	0.069

6.7 Conclusions

We propose an algorithm that combines deterministic predictions of an infinite pool of linear regressions and outputs probability forecasts in the form of cumulative distribution functions. The proposed strategy allows us to 'track the best expert'. The theoretical bound on the discounted cumulative CRPS loss function of the algorithm is derived.

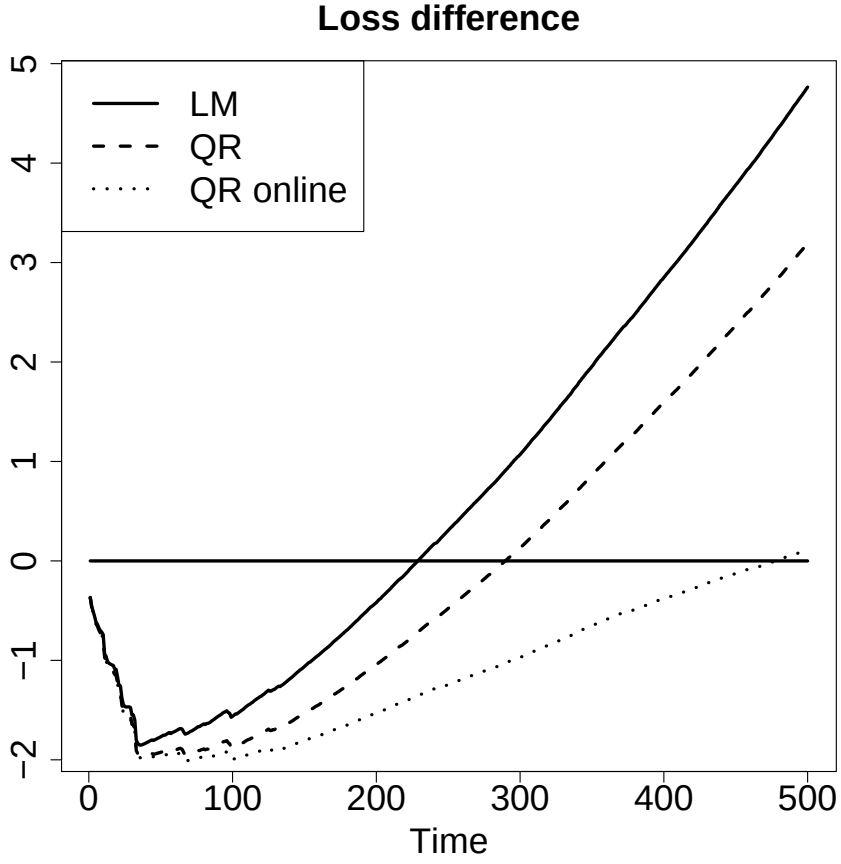


Figure 6.5: Loss difference between competitors and our algorithm

We perform experiments to evaluate the performance of our algorithm on synthetic and solar power datasets. The first experiment shows that the theoretical bound of our algorithm is not violated. The second experiment on the synthetic dataset show that the loss discounting helps in situations when the underlying nature of data changes with time; and our algorithm can outperform the best online quantile regression. The experiment with prediction of solar power shows that our algorithm needs some time for training, however after an initial time passes, the performance of our algorithm becomes close to the performance of the quantile regression.

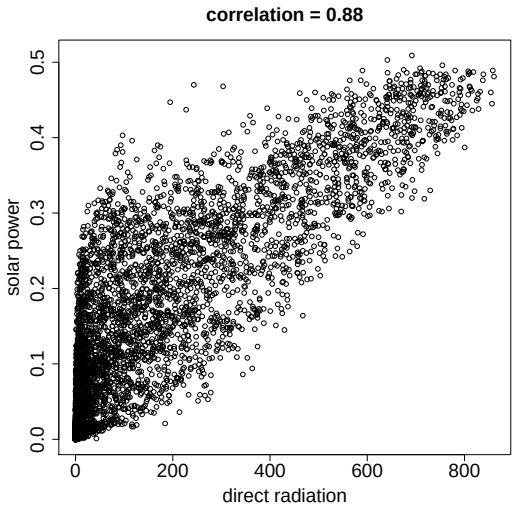


Figure 6.6: Dependence of solar power on direct radiation

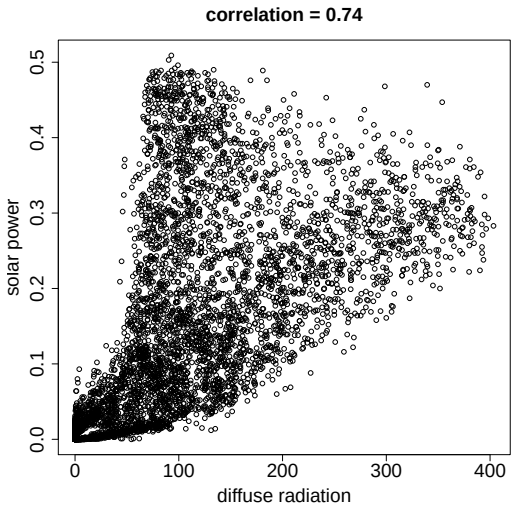


Figure 6.7: Dependence of solar power on diffuse radiation

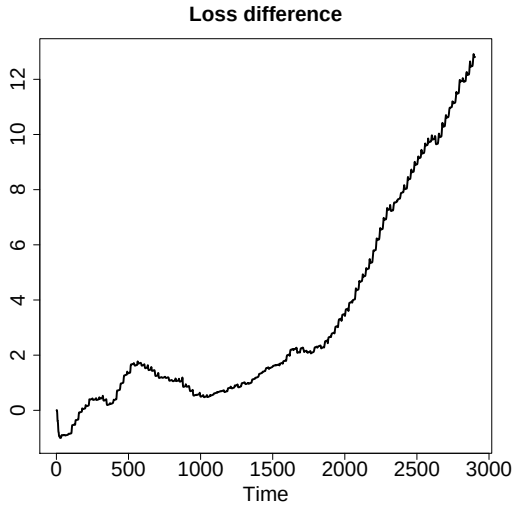


Figure 6.8: Loss difference between the best linear regression and our algorithm

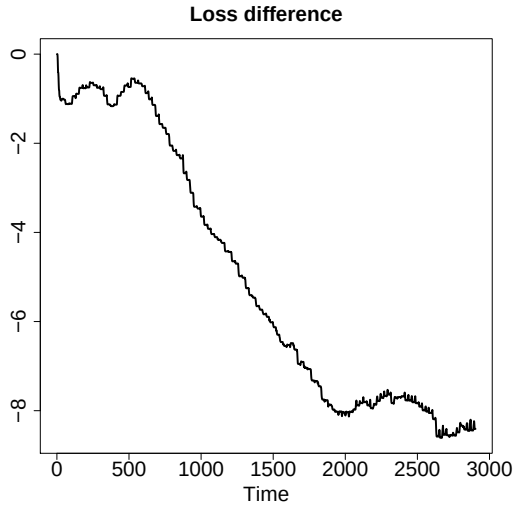
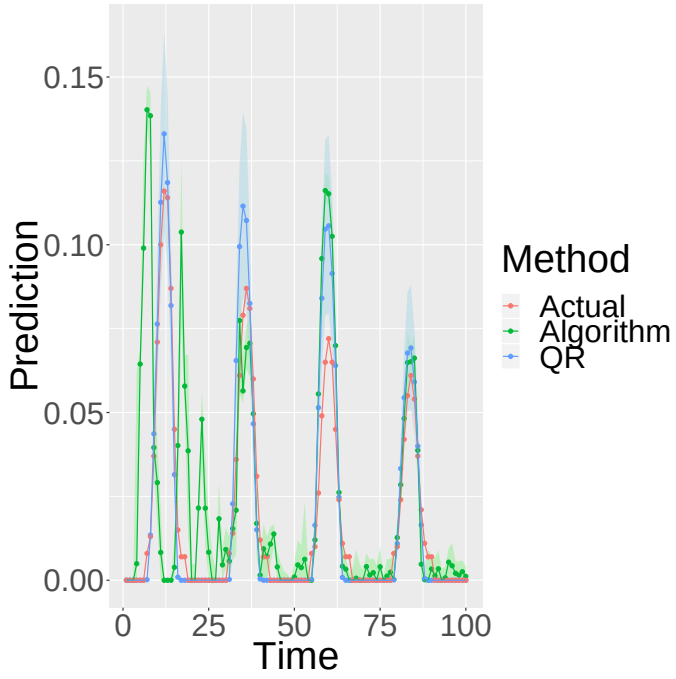
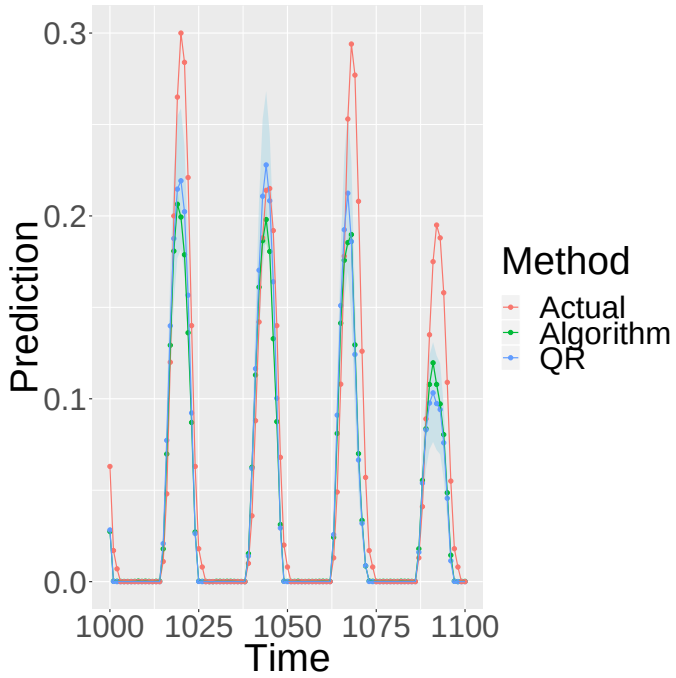


Figure 6.9: Loss difference between the best quantile regression and our algorithm



(a) first 100 steps



(b) after 1000 steps

Figure 6.10: Predictions with $[25\%, 75\%]$ confidence interval of our Algorithm and QR

Chapter 7

Universal algorithms for probabilistic multi-class classification

In this chapter, we propose a universal algorithm predicting finite-dimensional distributions, i.e. points from a simplex, under Kullback-Leibler game. A natural choice of predictors for the probability games is a class of multinomial logistic regression functions as they output a distribution that lies inside a probability simplex. We consider the class of multinomial logistic regressions to be our experts. We provide a strategy that allows us to 'track the best expert' of this type and derive the theoretical bound on the discounted loss of the strategy. We provide the kernelized version of our algorithm, which competes with a wider set of experts from Reproducing Kernel Hilbert Space (RKHS) and prove a theoretical guarantee for the kernelized strategy.

7.1 Introduction

An important class of games of prediction are probability forecasting games, where the predictions and outcomes are probability distributions on some finite set. In this chapter, we consider the Kullback-Leibler game, which is one of the most important probability games. Our experts are a wide class of multinomial logistic regression functions. Each expert follows a particular strategy, which means that it uses some particular parameters of a logistic regression function. Our goal is to develop a merging strategy that suffers loss comparable to the retrospectively best expert. If we use weights decreasing for old data, we get a strategy that performs as well as the best expert on *recent* trials; this can be thought of as a way of tracking the best expert

alternative to fixed share techniques.

In this chapter, we develop a universal algorithm for predicting finite-dimensional distributions, i.e., points from a simplex, under Kullback-Leibler loss. Related problems have been considered in the literature. Online convex optimization is a similar area where a decision-maker makes a sequence of decisions from a fixed feasible set. After each point is chosen, it encounters a convex cost function. In Hazan (2016) a logarithmic regret bound was obtained for α-convex cost functions, which have a lower bound on the second derivative; these bounds are not applicable here for the lack of such bound. A similar problem was considered in Kivinen and Warmuth (2001), where the authors proposed a general additive algorithm based on gradient descent and derived loss bounds that compare the loss of the resulting online algorithm to the best offline predictor from the relevant model class. They considered a softmax transfer function (Example 4 in Kivinen and Warmuth (2001)) and achieved a theoretical bound with a multiplicative coefficient of two in front of the loss of the best expert; whereas we achieved a multiplicative coefficient of one, which indicates that our theoretical bound is better for large losses.

The multidimensional prediction problem was considered in Zhdanov and Kalnishkan (2010), where the authors introduced an algorithm competitive with linear functions under the squared loss. One of the drawbacks of introducing linear experts for probability games is that predictions of linear experts could lie outside a probability simplex. The case of generalised linear regression experts under log-loss was introduced in Kakade and Ng (2005), and the case of the square loss was considered in Zhdanov and Vovk (2010). In all the above cases the authors achieved the theoretical bounds which were logarithmic in the number of steps. In a recent paper Foster et al. (2018) an algorithm was constructed for the case where outcomes and predictions are distributions on a finite set, the loss function is logarithmic, and competitors are linear functions with softmax applied on top of them. The paper contains an excellent survey of application domains. We propose an algorithm for the similar setting, but improve the regret term in the upper bound on the loss. Our regret bound has a lower growth rate w.r.t. the number of dimensions and does not contain the linear term on the number of steps. Asymptotically in T the regret is still of the order $C \ln T$, but our multiplicative constant C is lower.

In this chapter, we provide an explicit universal algorithm for predicting probability distributions, which can 'track the best expert' in terms of discounted cumulative Kullback-Leibler loss function. Kullback-Leibler game is one of the most important probability games (Vovk (2001)). Kullback-Leibler divergence is a measure of discrepancy between two probability distributions (Cover and Thomas (2006)), and it is widely

used in different areas such as applied statistics, econometrics, risk management and machine learning. An excellent survey of application of entropy and divergence measures in econometrics can be found in Ullah (1996). Useful applications of the entropy and Kullback-Leibler divergence for studying income inequality and welfare economics are described in Theil (1967) and Maasoumi (1986). The Kullback-Leibler loss function is also used in optimal portfolio selection and solving portfolio diversification problem (Bera and Park (2008)).

We apply AA with Discounting to multinomial logistic regression experts. Multinomial logistic regression predictors are a natural choice for probability games as they output predictions that lie inside a probability simplex. We provide a strategy that 'tracks the best expert' of this type and derive the theoretical bound on the discounted loss of the strategy. We generalise our algorithm to allow it to compete with wider set of experts from Reproducing Kernel Hilbert Space (RKHS) and prove the theoretical guarantee for the kernelized strategy.

Theoretical bounds obtained for Kullback-Leibler game are valid for logarithmic loss game. Indeed, Kullback-Leibler loss function is a generalisation of log-loss function where outcome space is the whole simplex instead of only vertices of the simplex as in the case of log-loss game. Therefore, theoretical bounds obtained for Kullback-Leibler game can be applied to the important problem of probabilistic multi-class classification under logarithmic loss function.

We conduct experiments to compare the performance of our algorithm with multinomial logistic regression. In our experiments we check that the theoretical bound for our algorithm is not violated. Our prediction algorithm is using Markov chain Monte Carlo (MCMC) method in a way which is similar to the algorithm introduced in Section 5.4. With the experiments provided we show that by tuning parameters online, our algorithm moves fast to the area of high values of the probability function and gives a good approximation of the prediction, and theoretical bounds are not violated.

7.2 Framework

We consider a probability game $\mathfrak{G}$ on some finite set $\Xi = \{1, \dots, d\}$, where space of outcomes $\Omega = \mathbb{P}(\Xi) = \{(y^{(1)}, \dots, y^{(d)}) : \sum_{i=1}^{d} y^{(i)} = 1,\ 0 \le y^{(i)} \le 1\}$, decision space $\Gamma = \mathbb{P}(\Xi) = \{(\gamma^{(1)}, \dots, \gamma^{(d)}) : \sum_{i=1}^{d} \gamma^{(i)} = 1,\ 0 \le \gamma^{(i)} \le 1\}$ are simplices in d-dimensional space, and for any $y \in \Omega$ and $\gamma \in \Gamma$ we define the Kullback-Leibler loss

$$\lambda(y, \gamma) = \sum_{i=1}^{d} y^{(i)} \ln \frac{y^{(i)}}{\gamma^{(i)}}, \tag{7.1}$$

where $y^{(i)}$ and $\gamma^{(i)}$ are the i-th coordinate of the respective vectors. As in (Cover and Thomas (2006), Section 2.3) we assume that for $p, q > 0$ we have $0 \ln \frac{0}{q} = 0$, $p \ln \frac{p}{0} = +\infty$, and $0 \ln \frac{0}{0} = 0$. The loss function λ defined in this way is continuous in γ and satisfies Assumptions 1–4 from Vovk (1998).

Learner and experts work according to the Protocol 3 defined in Section 2.4.

The cumulative losses of the learner are discounted with a factor $\alpha_t \in (0, 1]$ at each step. If L_{T-1} is the discounted cumulative loss of the learner at step $T-1$, then the discounted cumulative loss of the learner at step T is defined by (2.20):

$$L_T := \alpha_{T-1} L_{T-1} + \lambda_T(y_T, \gamma_T) = \sum_{t=1}^{T-1} \left(\prod_{j=t}^{T-1} \alpha_j \right) \lambda_t(y_t, \gamma_t) + \lambda_T(y_T, \gamma_T).$$

If L^{θ}_{T-1} is the discounted cumulative loss of the prediction strategy θ at the step $T-1$, then the discounted cumulative loss of the prediction strategy θ at the step T is defined by (2.21).

$$L^{\theta}_T := \alpha_{T-1} L^{\theta}_{T-1} + \lambda_T(y_T, \xi_T(\theta)) = \sum_{t=1}^{T-1} \left(\prod_{j=t}^{T-1} \alpha_j \right) \lambda_t(y_t, \xi_t(\theta)) + \lambda_T(y_T, \xi_T(\theta)).$$

We want to find a strategy which is capable of competing in terms of cumulative losses with all prediction strategies which at step t output $\xi_t(\theta) = (\xi^1_t(\theta), \ldots, \xi^d_t(\theta))$:

$$\xi^i_t(\theta) = \sigma_i(\theta, x_t), \ i = 1, \ldots, d, \tag{7.2}$$

where $\sigma_i(\theta, x_t)$ is multinomial logistic regression function:

$$\sigma_i(\theta, x_t) = \frac{e^{\theta_i' x_t}}{\sum_{j=1}^{d-1} e^{\theta_j' x_t} + 1}, \ i = 1, \ldots, d-1, \tag{7.3}$$

$$\sigma_d(\theta, x_t) = \frac{1}{\sum_{j=1}^{d-1} e^{\theta_j' x_t} + 1}, \tag{7.4}$$

and $\theta = (\theta_1', \ldots, \theta_{d-1}')' \in \mathbb{R}^{n(d-1)}$, $\theta_i = (\theta_{i,1}, \ldots, \theta_{i,n})' \in \mathbb{R}^n$.

Another possible choice of prediction strategies for multi-class classification problem is the class of softmax functions, which also output a distribution that lies inside a probability simplex. We consider this class of experts in Dzhamtyrova and Kalnishkan (2019). However, the class of multinomial logistic regressions achieves a slightly better regret term, which is proportional to $d-1$; whereas the regret term for the class of softmax functions is proportional to d.

7.3 Theoretical Bounds

We introduce a novel theoretical bound for our algorithm for the multi-class classification problem where $d \geq 3$. The case of $d = 2$ is considered in Kakade and Ng (2005).

Theorem 7.3.1. *Let $a > 0$. There exists a prediction strategy for Learner such that for every positive integer T, every sequence of outcomes of length T and every sequence $\alpha_t \in (0, 1]$, $t = 1, \ldots, T$, and every $\theta \in \mathbb{R}^{n(d-1)}$ the cumulative loss L_T of the Learner satisfies*

$$L_T \leq L_T^\theta + a\|\theta\|_2^2 + \frac{d-1}{2} \ln \det \left(I + \frac{d-1}{8a} X' W_T X \right), \tag{7.5}$$

where I is $n \times n$ unit matrix and X is the matrix with rows $x_1', \ldots, x_T'$, and $W_T = \operatorname{diag}(w_{1,T}, \ldots, w_{T,T})$, where $w_{t,T} = \prod_{j=t}^{T-1} \alpha_j$. If in addition $\|x_t\|_\infty \leq B$ for all t then

$$L_T \leq L_T^\theta + a\|\theta\|_2^2 + \frac{n(d-1)}{2} \ln \left(1 + \frac{d-1}{8a} B^2 \sum_{t=1}^{T} w_{t,T} \right). \tag{7.6}$$

The theorem states that the algorithm predicts as well as the best 'switching' multinomial logistic regression, defined in (7.3) and (7.4), up to an additive regret of the form $C \ln T$ in terms of the discounted cumulative loss. Large parameters of regularisation increase the bound by an additive term $a\|\theta\|_2^2$, however the regret term has a smaller growth rate as time increases. As the maximum time T is usually not known in advance, the regularisation parameter a cannot be optimised, and its choice depends on the particular task.

It is easier to see that the regret term is of the form $C \ln T$ for the undiscounted case, when we have:

Corollary 4. *Let $a > 0$. There exists a prediction strategy for Learner such that for every positive integer T, every sequence of outcomes of length T, and every $\theta \in \mathbb{R}^{n(d-1)}$ the cumulative loss L_T of Learner satisfies*

$$L_T \leq L_T^\theta + a\|\theta\|_2^2 + \frac{d-1}{2} \ln \det \left(I + \frac{d-1}{8a} \sum_{t=1}^{T} x_t x_t' \right). \tag{7.7}$$

If in addition $\|x_t\|_\infty \leq B$ for all t then

$$L_T \leq L_T^\theta + a\|\theta\|_2^2 + \frac{n(d-1)}{2} \ln \left(1 + \frac{d-1}{8a} B^2 T \right). \tag{7.8}$$

7.4 Prediction Strategy

In this section, we will provide a strategy for calculating predictions for the Kullback-Leibler game. First, we show the mixability of the Kullback-Leibler game.

Lemma 7.4.1. (Lemma 4 in Vovk (2001)). *The Kullback–Leibler game is 1-mixable. The AA for the Kullback–Leibler game with learning rate 1 coincides with the Bayesian mixture.*

The lemma states that the maximum $\eta = 1$ for the Kullback-Leibler game.

Now we will show that the Kullback-Leibler game is a generalisation of the logarithmic loss game, described in Section 2.2.2, where outcome space is the whole simplex instead of only vertices of the simplex as in the case of the log-loss game. Therefore, theoretical bounds obtained in Theorem 7.3.1 and Corollary 4 are valid for the problem of multi-class classification under logarithmic loss game.

Lemma 7.4.2. *Let $\gamma \in \Gamma$ is a permitted prediction for the logarithmic loss game, i.e. $\lambda(e_i, \gamma) \leq g_T(e_i)$, for $i = 1, \ldots, d$. Then γ is a permitted prediction for Kullback-Leibler game, i.e. $\lambda(y, \gamma) \leq g_T(y)$ for all $y \in \mathbb{P}(\Xi)$.*

Proof. Let $\gamma \in \Gamma$ be a permitted prediction for log-loss game. From (7.1) Kullback-Leibler loss function is

$$\lambda(y, \gamma) = \sum_{i=1}^{d} y^{(i)} \ln \frac{y^{(i)}}{\gamma^{(i)}} = \sum_{i=1}^{d} y^{(i)} \ln y^{(i)} - \sum_{i=1}^{d} y^{(i)} \ln \gamma^{(i)}$$
$$= \sum_{i=1}^{d} y^{(i)} \ln y^{(i)} + \sum_{i=1}^{d} y^{(i)} \lambda(e_i, \gamma).$$

Generalised prediction (2.3) for Kullback-Leibler game is

$$\begin{aligned}
g_T(y) &= -\frac{1}{\eta}\ln\int_\Theta e^{-\eta\lambda(y,\gamma)}P^*_{T-1}(d\theta) \\
&= \sum_{i=1}^d y^{(i)}\ln y^{(i)} - \frac{1}{\eta}\ln\int_\Theta e^{-\eta\sum_{i=1}^d y^{(i)}\lambda(e_i,\gamma)}P^*_{T-1}(d\theta) \\
&= \sum_{i=1}^d y^{(i)}\ln y^{(i)} - \frac{1}{\eta}\ln\int_\Theta \prod_{i=1}^d e^{-\eta y^{(i)}\lambda(e_i,\gamma)}P^*_{T-1}(d\theta) \\
&\geq \sum_{i=1}^d y^{(i)}\ln y^{(i)} - \frac{1}{\eta}\ln\prod_{i=1}^d\int_\Theta \left(e^{-\eta\lambda(e_i,\gamma)}P^*_{T-1}(d\theta)\right)^{y^{(i)}} \\
&= \sum_{i=1}^d y^{(i)}\ln y^{(i)} + \sum_{i=1}^d y^{(i)}\left(-\frac{1}{\eta}\ln\int_\Theta e^{-\eta\lambda(e_i,\gamma)}P^*_{T-1}(d\theta)\right) \\
&= \sum_{i=1}^d y^{(i)}\ln y^{(i)} + \sum_{i=1}^d y^{(i)}g_T(e_i) \\
&\geq \sum_{i=1}^d y^{(i)}\ln y^{(i)} + \sum_{i=1}^d y^{(i)}\lambda(e_i,\gamma) = \lambda(y,\gamma).
\end{aligned}$$

The first inequality follows from the generalised Hölder inequality (this follows from the version of the inequality in Section 9.3 of Loève (1977) by induction). The second inequality follows from the fact that γ is a permitted prediction for log-loss game. We showed that prediction γ satisfies the inequality (2.4) for any $y \in \mathbb{P}(\Xi)$. Therefore, γ is a permitted prediction for Kullback-Leibler game. □

We choose the normal initial distribution of parameters

$$P_0(d\theta) = (a/\pi)^{n(d-1)/2}\exp(-a\|\theta\|_2^2)d\theta \tag{7.9}$$

for some $a > 0$.

We calculate generalised prediction for AAD from unnormalised weights (2.27) and

taking initial parameter distribution (7.9)

$$G_T(e_k) = -\frac{1}{\eta}\ln\int_\Theta P_0(d\theta)\left(\tilde{P}_{T-1}(\theta)\right)^{\alpha_{T-1}} e^{-\eta\lambda(e_k,\xi_T(\theta))}$$
$$= -\frac{1}{\eta}\ln\int_\Theta P_0(d\theta)e^{-\eta\sum_{t=1}^{T-1}L_t(y_t,\xi_t(\theta))-\eta\lambda_T(e_k,\xi_T(\theta))}$$
$$= -\frac{1}{\eta}\ln(a/\pi)^{n(d-1)/2}\int_\Theta(\xi_T^k(\theta))^\eta e^{-\eta\sum_{t=1}^{T-1}(\prod_{j=t}^{T-1}\alpha_j)\sum_{i=1}^{d}y_j^i\ln\frac{y_j^i}{\xi_j^i(\theta)}-a\|\theta\|_2^2}d\theta,$$
$$k=1,\ldots,d. \quad (7.10)$$

Generalised prediction (7.10) calculated from unnormalized weights will differ from the generalised prediction (2.3) calculated from normalized weights by only an additive constant.

By putting $\eta = 1$ from Lemma 7.4.1 and applying substitution function $e^{-(.)}$ prediction at step T predicts is expressed as follows

$$\gamma_T^k = \int_\Theta \xi_T^k(\theta)q_{T-1}^*(\theta)d\theta,\ k=1,\ldots,d, \quad (7.11)$$

where

$$q_T^*(\theta) = Zq_T(\theta) = Z\exp\Big(-\sum_{t=1}^{T-1}\left(\prod_{j=t}^{T-1}\alpha_j\right)\sum_{i=1}^{d}y_t^i\ln\frac{y_t^i}{\xi_t^i(\theta)} - a\|\theta\|_2^2\Big), \quad (7.12)$$

and Z is the normalising constant ensuring that $\int_\Theta q_T^*(\theta)d\theta = 1$.

Because integral (7.11) cannot be calculated analytically, we use the same technique, described in Sections 5.4, 6.4, to approximate predictions of the proposed strategy.

We use Metropolis-Hastings sample parameters θ from the unnormalized posterior distribution $q_T(\theta)$ (7.12). The pseudo-code of the algorithm is given below:

Algorithm

Parameters: number $M > 0$ of MCMC iterations,
standard deviation $\sigma > 0$,
regularization coefficient $a > 0$

$\eta := 1$
initialize $\theta_0^M := 0 \in \Theta$
define $q_0(\theta) := \exp(-a\eta\|\theta\|_2^2)$
for $t = 1, 2, \ldots$ do
  $\gamma_t^i := 0,\ i = 1,\ldots,d$

define $q_{t-1}(\theta)$ by (7.12) if $t > 1$
read $x_t \in \mathbb{R}^n$
initialize $\theta_t^0 = \theta_{t-1}^M$
for $m = 1, 2, \ldots, M$ do
$\theta^* := \theta_t^{m-1} + \mathcal{N}(0, \sigma^2 I)$
flip a coin with success probability
$\min\left(1, q_{t-1}(\theta^*)/q_{t-1}(\theta_t^{m-1})\right)$
if success then
$\theta_t^m := \theta^*$
else
$\theta_t^m := \theta_{t-1}^m$
end if
$\gamma_t^i := \gamma_t^i + \eta \xi_t^i(\theta_t^m),\ i = 1, \ldots, d$
end for
output predictions $\gamma_t^i = \gamma_t^i / M,\ i = 1, \ldots, d$
end for

7.5 Proof of Theoretical Bounds

In this section, we provide the proof of Theorem 7.3.1.

Applying Lemma 2.4.2 for initial distribution (7.9) and putting $\eta = 1$ from Lemma 7.4.1 for Kullback-Leibler loss function we obtain

$$\sum_{t=1}^{T} w_{t,T} \sum_{i=1}^{d} y_t^i \ln \frac{y_t^i}{\gamma_t^i} \le -\ln\left((a/\pi)^{n(d-1)/2} \int_\Theta e^{-J(\theta)} d\theta\right), \tag{7.13}$$

where

$$w_{t,T} = \prod_{j=t}^{T-1} \alpha_j,\ w_{T,T} = 1$$

and

$$J(\theta) := \sum_{t=1}^{T} w_{t,T} \sum_{i=1}^{d} y_t^i \ln \frac{y_t^i}{\sigma_i(\theta, x_t)} + a\|\theta\|_2^2.$$

We use Taylor expansion (Section 1.7c in Courant and John (1989)) of $J(\theta)$ at the point θ_0 where $\min J(\theta)$ is obtained:

$$J(\theta) = J(\theta_0) + \frac{1}{2}(\theta - \theta_0)' H(\phi)(\theta - \theta_0),$$

where ϕ is a convex combination of θ_0 and θ, and H is the Hessian matrix of $J(\theta)$.

The second partial derivative of $J(\theta)$ by the l-th, j-th components of θ_k and θ_m respectively is expressed as follows:

$$\frac{\partial^2 J(\theta)}{\partial\theta_{k,l}\partial\theta_{m,j}} = 2a\delta_l^j\delta_k^m + \sum_{t=1}^{T} w_{t,T} \sum_{i=1}^{d} \Big(y_t^i \frac{1}{\sigma_i^2(\theta, x_t)} \frac{\partial\sigma_i(\theta, x_t)}{\partial\theta_{k,l}} \frac{\partial\sigma_i(\theta, x_t)}{\partial\theta_{m,j}} - y_t^i \frac{1}{\sigma_i(\theta, x_t)} \frac{\partial^2\sigma_i(\theta, x_t)}{\partial\theta_{k,l}\partial\theta_{m,j}} \Big), \quad (7.14)$$

where

$$\delta_k^m = \begin{cases} 1, & \text{if } k = m \\ 0, & \text{if } k \neq m \end{cases}$$

is Kronecker delta.

The first and second partial derivatives of the function $\sigma_i(\theta, x_t)$ are as follows:

$$\frac{\partial\sigma_i(\theta, x_t)}{\partial\theta_{k,l}} = x_{t,l}\sigma_i(\theta, x_t)(\delta_i^k - \sigma_k(\theta, x_t)),$$

$$\frac{\partial^2\sigma_i(\theta, x_t)}{\partial\theta_{k,l}\partial\theta_{m,j}} = x_{t,l}x_{t,j}\sigma_i(\theta, x_t)\Big(\delta_k^m - \delta_k^m\sigma_k(\theta, x_t) - \delta_i^k\sigma_m(\theta, x_t) - \delta_i^m\sigma_k(\theta, x_t) + 2\sigma_k(\theta, x_t)\sigma_m(\theta, x_t)\Big).$$

Expression (7.14) can be re-written as follows:

$$\frac{\partial^2 J(\theta)}{\partial\theta_{k,l}\partial\theta_{m,j}} = 2a\delta_l^j\delta_k^m + \sum_{t=1}^{T} w_{t,T}x_{t,l}x_{t,j}f_{k,m}(\theta, x_t), \quad (7.15)$$

where

$$f_{k,m}(\theta, x_t) = \sigma_k(\theta, x_t)(\delta_k^m - \sigma_m(\theta, x_t)).$$

We denote $W_T = \mathrm{diag}(w_{1,T}, w_{2,T}, \ldots, w_{T,T})$ the diagonal matrix T by T. Let X be the $T \times n$ matrix with the rows $x_1', \ldots, x_T'$ and $\Gamma_{k,m}(\phi)$ be the diagonal $T \times T$ matrix that has $f_{k,m}(\phi, x_1, y_1), \ldots, f_{k,m}(\phi, x_T, y_T)$ on the diagonal. Let Z be the block matrix as follows:

$$Z = \begin{pmatrix} X_{1,1} & \ldots & X_{1,d-1} \\ \vdots & \ddots & \vdots \\ X_{d-1,1} & \ldots & X_{d-1,d-1} \end{pmatrix},$$

where

$$X_{k,m} = \begin{cases} \sqrt{W_T}X, & \text{if } k = m \\ O, & \text{if } k \neq m \end{cases}$$

Let $\Gamma(\phi)$ to be a block matrix as follows:

$$\Gamma(\phi) = \begin{pmatrix} \Gamma_{1,1}(\phi) & \dots & \Gamma_{1,d-1}(\phi) \\ \vdots & \ddots & \vdots \\ \Gamma_{d-1,1}(\phi) & \dots & \Gamma_{d-1,d-1}(\phi) \end{pmatrix}.$$

Then Hessian matrix of $J(\theta)$ can be written in the matrix form:

$$H(\phi) = 2aI + Z'\Gamma(\phi)Z. \tag{7.16}$$

Since $\Gamma(\phi)$ is a symmetric matrix, we can see (Theorem 21.5.6 in Harville (1997)) that:

$$\psi'\Gamma(\phi)\psi \leq \psi'\lambda_{\max}(\Gamma(\phi))\psi,$$

for any $\psi \in \mathbb{R}^{T(d-1)}$ where $\lambda_{\max}(\Gamma(\phi))$ is the supremum over maximum eigenvalues of $\Gamma(\phi)$.

We will now show that matrix $\Gamma(\phi)$ is positive definite. The absolute value of the diagonal element of $\Gamma(\phi)$ is

$$|f_{k,k}(\theta, x_t)| = \sigma_k(\theta, x_t)(1 - \sigma_k(\theta, x_t)),$$

the sum of the absolute values of non-diagonal elements on the row is

$$\begin{aligned} \sum_{m \neq k} |f_{k,m}(\theta, x_t)| &= \sum_{m \neq k} |-\sigma_k(\theta, x_t)\sigma_m(\theta, x_t)| \\ &= \sigma_k(\theta, x_t) \sum_{m \neq k} \sigma_m(\theta, x_t) = \sigma_k(\theta, x_t)(1 - \sigma_k(\theta, x_t) - \sigma_d(\theta, x_t)). \end{aligned}$$

As $|f_{k,k}(\theta, x_t)| > \sum_{m \neq k} |f_{k,m}(\theta, x_t)|$, for all $k = 1, \dots, d-1$, $t = 1, \dots, T$, then by Diagonal Dominance Theorem 6.1.10 in (Horn and Johnson (1985)) $\Gamma(\phi)$ is positive definite.

By Theorem A.3 (see Appendix), we can upper bound $\lambda_{\max}(\Gamma(\phi))$, the maximum eigenvalue of $\Gamma(\phi)$, by the sum of maximum eigenvalues of diagonal blocks $\Gamma_{i,i}(\phi)$:

$$\lambda_{\max}(\Gamma(\phi)) \leq \sum_{i=1}^{d-1} \lambda_{\max}(\Gamma_{i,i}(\phi)).$$

Since $\Gamma_{i,i}(\phi)$ is diagonal then:

$$\lambda_{\max}(\Gamma_{i,i}(\phi)) = \sup_{\theta\in\mathbb{R}^{n(d-1)},\ x_t\in\mathbb{R}^n} f_{i,i}(\theta, x_t)$$
$$= \sup_{\theta\in\mathbb{R}^{n(d-1)},\ x_t\in\mathbb{R}^n} \sigma_i(\theta, x_t)(1-\sigma_i(\theta, x_t)) = \frac{1}{4}.$$

By taking $\psi = Z(\theta - \theta_0)$ and using (7.16):

$$J(\theta) \le J(\theta_0) + (\theta-\theta_0)'(aI + \frac{d-1}{8}Z'Z)(\theta-\theta_0).$$

We can obtain the lower bound on the integral in (7.13):

$$\int_\Theta e^{-J(\theta)d\theta} \ge e^{-J(\theta_0)} \int_\Theta e^{-(\theta-\theta_0)'(aI+\frac{d-1}{8}Z'Z)(\theta-\theta_0)}d\theta.$$

The integral in the right-hand side can be calculated analytically (see Theorem A.1 in Appendix):

$$\int_\Theta e^{-(\theta-\theta_0)'(aI+\frac{d-1}{8}Z'Z)(\theta-\theta_0)}d\theta = \frac{\pi^{\frac{n(d-1)}{2}}}{\sqrt{\det(aI+\frac{d-1}{8}Z'Z)}}.$$

After putting this expression in (7.13) we obtain the upper bound:

$$L_T \le -\ln\left(e^{-J(\theta_0)}\left(\frac{a}{\pi}\right)^{\frac{n(d-1)}{2}}\pi^{\frac{n(d-1)}{2}}\frac{1}{\sqrt{\det(aI+\frac{d-1}{8}Z'Z)}}\right)$$
$$= J(\theta_0) + \frac{1}{2}\ln\det\left(I+\frac{d-1}{8a}Z'Z\right) = L_T^{\theta_0} + a\|\theta_0\|_2^2$$
$$+ \frac{d-1}{2}\ln\det\left(I+\frac{d-1}{8a}X'W_TX\right).$$

If $\|x_t\|_\infty \le B$ the determinant of a symmetric positive definite matrix is upper bounded by the product of its diagonal elements (see Chapter 2, Theorem 7 in Beckenbach and Bellman (1961)):

$$\det\left(I+\frac{d-1}{8a}X'W_TX\right) \le \left(1+\frac{d-1}{8a}B^2\sum_{t=1}^T w_{t,T}\right)^n.$$

7.6 Kernelized Algorithm

In this section, we kernelize the algorithm and prove upper bounds on the Kullback-Leibler loss of the algorithm competing with wider class of experts.

We start with the definition of a kernel. A *kernel* on a domain $\mathbf{X}$, which is an arbitrary set with no structure assumed, is a symmetric positive-semidefinite function of two arguments, i. e., $k : \mathbf{X} \times \mathbf{X} \to \mathbb{R}$ such that

1. for all $x_1, x_2 \in \mathbf{X}$ we have $k(x_1, x_2) = k(x_2, x_1)$,
2. for any positive integer T, any $x_1, x_2, \ldots, x_T \in \mathbf{X}$ and any real numbers $a_1, a_2, \ldots, a_T \in \mathbb{R}$ we have $\sum_{i,j=1}^{T} a_i a_j k(x_i, x_j) \geq 0$.

An equivalent definition can be given as follows. A function $k : \mathbf{X} \times \mathbf{X} \to \mathbb{R}$ is a *kernel* if there is a Hilbert space $\mathcal{F}$ of functions on $\mathbf{X}$ such that

1. for every $x \in \mathbf{X}$ the function $k(x, \cdot)$, i. e., k considered as a function of the second argument with the first argument fixed, belongs to $\mathcal{F}$,
2. for every $x \in \mathbf{X}$ and every $f \in \mathcal{F}$ the value of f at x equals the scalar product of f by $k(x, \cdot)$, i. e., $f(x) = \langle f, k(x, \cdot)\rangle_{\mathcal{F}}$; this property is often called the *reproducing property*.

The second definition is sometimes said to specify a *reproducing kernel*. A space $\mathcal{F}$ admitting a reproducing kernel is called a *reproducing kernel Hilbert space (RKHS)*.

Finally, we give a definition of a kernel as the scalar product in a feature space. Given a feature mapping $\Phi : \mathbf{X} \to \mathcal{F}$, where $\mathcal{F}$ is a Hilbert space, a *kernel* is defined as

$$k(x_1, x_2) = \langle \Phi(x_1), \Phi(x_2)\rangle.$$

These three definitions are equivalent since a function $k(x_1, x_2) : \mathbf{X} \times \mathbf{X} \to \mathbb{R}$ can be represented in the form $\langle \Phi(x_1), \Phi(x_2)\rangle$ iff k is the reproducing kernel of an RKHS iff k is symmetric and positive semi-definite.

In order to kernelize our algorithm, we formulate it in a *dual form*, where all input vectors appear only in dot products. These dot products are then replaced by kernels. This procedure is known as the *kernel trick*. The following lemma restating Theorem 7.3.1 in the dual form.

Lemma 7.6.1. *Under the conditions of Theorem 7.3.1, the cumulative loss L_T of the Learner satisfies*

$$L_T \leq L_T^{\theta} + a\|\theta\|_2^2 + \frac{d-1}{2} \ln \det\left(I + \frac{d-1}{8a}\sqrt{W_T} X_T' X_T \sqrt{W_T}\right), \tag{7.17}$$

where $\sqrt{W_T} = \operatorname{diag}(\sqrt{w_{1,T}}, \sqrt{w_{2,T}}, \ldots, \sqrt{w_{T,T}})$.

Proof. The lemma follows from (7.5) and the Sylvester identity (Lemma A.4). □

The lemma opens the way for the kernelization of the loss bound along the usual lines, but one should be careful. We do not have an explicit formula for the universal algorithm and cannot state it in the dual form straightforwardly.

We will now define the kernel form of the algorithm. Our starting point is the representation given by (7.11).

Lemma 7.6.2. *Let J be an orthogonal $(n \times n)$-matrix. If all vectors x_t are replaced by Jx_t, $t = 1, 2, \ldots, T$, the value of γ_t^k given by (7.11) will not change.*

Proof. Vectors x_t appear in the integral of (7.11) only in scalar products $x_t'\theta_i$. Let us replace all x_t by Jx_t. We have $(Jx_t)'\theta_i = x_t'(J'\theta_i)$. The substitution $\tilde{\theta}_i = J'\theta_i$ reduces the integral to the same form as before because $\|\tilde{\theta}\|_2^2 = \sum_{i=1}^{d-1} \theta_i' J' J \theta_i = \sum_{i=1}^{d-1} \|\theta_i\|_2^2 = \|\theta\|_2^2$ and $|\det(\operatorname{diag}(J, J, \ldots, J))| = 1$. □

Lemma 7.6.3. *Let all x_t, $t = 1, 2, \ldots, T$, belong to an m-dimensional subspace S_m of $\mathbb{R}^n$, $m < n$. Then the integral in (7.11) can be taken over $\Theta_m = (S_m)^{d-1}$, so that*

$$\gamma_T^k = \int_{\Theta_m} \xi_T^k(\theta) \tilde{q}_{T-1}^*(\theta) d\theta,$$

with

$$\tilde{q}_{T-1}^*(\theta) = \tilde{Z} \exp\left(-\sum_{t=1}^{T-1} \left(\prod_{j=t}^{T-1} \alpha_j \right) \sum_{i=1}^{d} y_t^i \ln \frac{y_t^i}{\xi_t^i(\theta)} - a\|\theta\|_2^2 \right), \tag{7.18}$$

where $\tilde{Z}$ is such that $\int_{\Theta_m} \tilde{q}_T^*(\theta) = 1$.

Proof. Lemma 7.6.2 implies that without restricting the generality we can assume that vectors in S_m have their last $n - m$ coordinates equal to 0. We can then split θ_i as $\theta_i' = (\tilde{\theta}_i', \hat{\theta}_i')$, where $\tilde{\theta}_i$ has m and $\hat{\theta}_i$ has $n - m$ coordinates $(i = 1, 2, \ldots, d-1)$, and take $\tilde{\theta} = (\tilde{\theta}_1', \ldots, \tilde{\theta}_{d-1}')'$ and $\hat{\theta} = (\hat{\theta}_1', \ldots, \hat{\theta}_{d-1}')'$. Since $x_t \in S_m$, one can split them as $x_t' = (\tilde{x}_t', 0)$, where 0 is of dimension $n - m$. We have $x_t'\theta_i = \tilde{x}_t'\tilde{\theta}_i$, and therefore $\xi_T^k((\tilde{\theta}_1', \hat{\theta}_1', \ldots, \tilde{\theta}_{d-1}', \hat{\theta}_{d-1}')') = \xi_T^k((\tilde{\theta}_1', 0, \ldots, \tilde{\theta}_{d-1}', 0)')$.

We have

$$\gamma_T^k = \int_\Theta \xi_T^k(\theta) q_{T-1}^*(\theta) d\theta = \\ = \int_{\mathbb{R}^{m(d-1)}} \int_{\mathbb{R}^{(n-m)(d-1)}} \xi_T^k((\tilde{\theta}'_1, 0, \ldots, \tilde{\theta}'_{d-1}, 0)') \\ \times Z \tilde{q}_{T-1}^*(\tilde{\theta}) \exp\left(-a \sum_{i=1}^{d-1} \|\hat{\theta}_i\|_2^2\right) d\tilde{\theta} d\hat{\theta}, \quad (7.19)$$

where Z is such that

$$Z \int_\Theta \tilde{q}_{T-1}^*(\tilde{\theta}) \exp\left(-a \sum_{i=1}^{d-1} \|\hat{\theta}_i\|_2^2\right) d\theta = 1.$$

Notice, that an integral $\int_{\mathbb{R}^{(n-m)(d-1)}} \exp\left(-a \sum_{i=1}^{d-1} \|\hat{\theta}_i\|_2^2\right) d\hat{\theta}$ evaluates to a constant. Then an application of Fubini's theorem to (7.19) completes the proof. □

Let $k : \mathbf{X} \times \mathbf{X} \to \mathbb{R}$, where $\mathbf{X}$ is some domain, be a kernel and let $\mathcal{F}$ with the scalar product $\langle \cdot, \cdot \rangle_\mathcal{F}$ and norm $\| \cdot \|_\mathcal{F}$ be the corresponding RKHS. Let $\Phi : \mathbf{X} \to \mathcal{F}$ be the feature mapping given by $\Phi(x) = k(x, \cdot)$.

Consider the kernelized modification of Protocol 1 with nature outputting $x_t \in \mathbf{X}$. We want to compete with predictors of the following kind. Take an array of $d-1$ functions $\mathbf{f} = (f_1, f_2, \ldots, f_{d-1}) \in \mathcal{F}^{d-1}$. At step t array $\mathbf{f}$ outputs $\xi_t(\mathbf{f}) = (\xi_t^1(\mathbf{f}), \ldots, \xi_t^d(\mathbf{f}))$ such that

$$\xi_t^i(\mathbf{f}) = \sigma_i(\mathbf{f}, x_t), \ i = 1, \ldots, d, \quad (7.20)$$

where $\sigma_i(\mathbf{f}, x_t)$ are multinomial logistic regression functions:

$$\sigma_i(\mathbf{f}, x_t) = \frac{e^{f_i(x_t)}}{\sum_{j=1}^{d-1} e^{f_j(x_t)} + 1}, \ i = 1, \ldots, d-1, \quad (7.21)$$

$$\sigma_d(\mathbf{f}, x_t) = \frac{1}{\sum_{j=1}^{d-1} e^{f_j(x_t)} + 1}. \quad (7.22)$$

The discounted cumulative loss $L_t^\mathbf{f}$ is defined similar to (2.21).

We will now construct a universal algorithm working according to the kernelized Protocol 1. The algorithm works as follows.

On step T let $\mathcal{F}_T \subseteq \mathcal{F}$ be the span of $\Phi(x_1), \Phi(x_2), \ldots, \Phi(x_T)$. It is a space of finite dimension $T' \leq T$ and it is isomorphic to $\mathbb{R}^{T'}$. We define γ_T^k by (7.11) with $\Theta = \mathbb{R}^{T'(d-1)}$ and $x'_t \in \mathbb{R}^{T'}$ being the values corresponding to $\Phi(x_t)$, $t = 1, 2, \ldots, T$.

Lemma 7.6.2 implies that the algorithm is well-defined and independent of the choice of a linear isomorphism.

The values of γ_T^k can be computed by evaluating the integral in (7.11) as follows. Each $\theta \in \Theta$ corresponds to a predictor $(\sigma_1(\mathbf{h}, x), \dots, \sigma_d(\mathbf{h}, x))$, where $\mathbf{h} = (h_1, h_2, \dots, h_{d-1}) \in \mathcal{F}^{d-1}$ is defined by $h_i(x) = \sum_{t=1}^{T} a_t^i k(x_t, x)$, where $a_1^i, a_2^i, \dots, a_T^i \in \mathbb{R}$ are some constants. The density

$$q_T^*(\theta) \propto \exp\left(- \sum_{t=1}^{T-1} \left(\prod_{j=t}^{T-1} \alpha_j \right) \sum_{i=1}^{d} y_t^i \ln \frac{y_t^i}{\xi_t^i(\mathbf{h})} - a \sum_{i=1}^{d-1} \|\mathbf{h}_i\|_{\mathcal{F}}^2 \right), \tag{7.23}$$

where $\|\mathbf{h}_i\|_{\mathcal{F}}^2 = \sum_{t_1,t_2=1}^{T} a_{t_1}^i a_{t_2}^i k(x_{t_1}, x_{t_2})$, may be evaluated (up to a multiplicative constant) once we know $a_1^i, a_2^i, \dots, a_T^i$. Therefore we can use MCMC doing a random walk over the space of coefficients a_t^i, i.e., $\mathbb{R}^{T(d-1)}$.

Theorem 7.6.4. *Let $a > 0$. There exists a prediction strategy S for the learner such that for every positive integer T, for every sequence of outcomes of the length T, and every sequence $\alpha_t \in (0, 1]$, $t = 1, \dots, T$, and any $\mathbf{f} = (f_1, \dots, f_{d-1}) \in \mathcal{F}^{d-1}$, the loss L_T of the learner satisfies*

$$L_T \le L_T^{\mathbf{f}} + a \sum_{i=1}^{d-1} \|f_i\|_{\mathcal{F}}^2 + \frac{d-1}{2} \ln \det \left(I + \frac{d-1}{8a} \sqrt{W_T} \mathbf{K}_T \sqrt{W_T} \right), \tag{7.24}$$

where

$$\mathbf{K}_T = \begin{pmatrix} k(x_1, x_2) & \dots & k(x_1, x_T) \\ \vdots & \ddots & \vdots \\ k(x_T, x_1) & \dots & k(x_T, x_T) \end{pmatrix},$$

and $W_T = \operatorname{diag}(w_{1,T}, w_{2,T}, \dots, w_{T,T})$, where $w_{t,T} = \prod_{j=t}^{T-1} \alpha_j$.

Proof. Fix a positive integer T. The distribution γ_t output by our algorithm is constructed using $\mathcal{F}_T$ of dimension T' isomorphic to some $\mathbb{R}^{T'}$. For $t < T$ the construction relies on a different $\mathcal{F}_t$ isomorphic to $\mathbb{R}^{t'}$. However, $\mathcal{F}_t \subseteq \mathcal{F}_T$ and $\mathcal{F}_t$ is isomorphic to a subspace of $\mathbb{R}^{T'}$. Lemmas 7.6.2 and 7.6.3 imply that γ_t could be calculated by integration over the same $\mathbb{R}^{T'(d-1)}$ with the same $x_1', \dots, x_t'$.

Let $f_1, f_2, \dots, f_{d-1} \in \mathcal{F}_T$. Then each is isomorphic to a θ_i with the same norm and the theorem follows from Lemma 7.6.1.

Let $f_1, f_2, \dots, f_{d-1} \in \mathcal{F}$ be arbitrary functions from the RKHS. Using the Representer Theorem argument (Lemma A.5 in Appendix), we can project each f_i on $\mathcal{F}_T$ and write $f_i = f_i^{\|} + f_i^{\perp}$, where $f_i^{\|} \in \mathcal{F}_T$ and $f_i^{\perp}$ is orthogonal to $\mathcal{F}_T$. By the construction of $\mathcal{F}_T$, we have $f_i(x_t) = f_i^{\|}(x_t)$ $i = 1, 2, \dots, d-1$ and $t = 1, 2, \dots, T$, but $\|f_i^{\|}\|_{\mathcal{F}} \le \|f_i\|_{\mathcal{F}}$.

Thus the orthogonal component does not affect predictions but increases the norm. The theorem follows. □

Note that for the undiscounted losses we have:

Corollary 5. *Let $a > 0$. There exists a prediction strategy S for the learner such that for every positive integer T, for every sequence of outcomes of the length T, and any $\mathbf{f} = (f_1, \ldots, f_{d-1}) \in \mathcal{F}^{d-1}$, the loss L_T of the learner satisfies*

$$L_T \le L_T^{\mathbf{f}} + a \sum_{i=1}^{d-1} \|f_i\|_{\mathcal{F}}^2 + \frac{d-1}{2} \ln \det \left(I + \frac{d-1}{8a} \mathbf{K}_T \right), \tag{7.25}$$

where

$$\mathbf{K}_T = \begin{pmatrix} k(x_1, x_2) & \ldots & k(x_1, x_T) \\ \vdots & \ddots & \vdots \\ k(x_T, x_1) & \ldots & k(x_T, x_T) \end{pmatrix}.$$

The order of the regret term in (7.25) may vary. However, we show that it has the order $O(\sqrt{T})$ in many cases. We will use the notation $c_{\mathcal{F}}^2 = \sup_{z \in \mathbf{X}} k(z, z)$ and assume $c_{\mathcal{F}}^2 < \infty$.

Corollary 6. *Under the conditions of Corollary 5 and if the number of steps T is known in advance, the kernelized algorithm with $a = c_{\mathcal{F}} \sqrt{T}$ achieves loss satisfying*

$$L_T \le L_T^{\mathbf{f}} + \left(\sum_{i=1}^{d-1} \|f_i\|_{\mathcal{F}}^2 + \frac{(d-1)^2}{16} \right) c_{\mathcal{F}} \sqrt{T}. \tag{7.26}$$

Proof. The determinant of a symmetric positive definite matrix is upper bounded by the product of its diagonal elements (see Chapter 2, Theorem 7 in Beckenbach and Bellman (1961)) and thus

$$\ln \det \left(I + \frac{d-1}{8a} \mathbf{K}_T \right) \le T \ln \left(1 + \frac{(d-1) c_{\mathcal{F}}^2}{8a} \right) \le T \frac{(d-1) c_{\mathcal{F}}^2}{8a}.$$

If we know the number of steps T in advance, then we can choose a specific value $a = c_{\mathcal{F}} \sqrt{T}$.

In a case when the number of trials is not known in advance, it is still possible to use a suitable initial weights distribution over the parameter a to achieve a similar bound using the AA (see Vovk (2005)). □

7.7 Experiments

In this section, we apply our algorithm on three datasets and compare its performance with the multinomial logistic regression. For simplicity we apply our algorithm for multi-class classification problems and put $\alpha_t = 1$ for all $t = 1, \ldots, T$. We obtained the best parameters of multinomial logistic regression by using function 'multinom' from library 'nnet' in R. [1]

7.7.1 Synthetic Dataset

We generated the synthetic 'Smiley' dataset that consists of two Gaussian eyes, a trapezoid nose and a parabola mouth. The function for generating this dataset was taken from R library 'mlbench'. Figure 7.1 shows the dataset which contains 1000 observations with two features and four classes: left eye, right eye, nose, and mouth. We divide our data in a way that each class will have half of its observations in training and test datasets. Figure 7.2 illustrates the generated training dataset. The split of the training and test dataset is not random to show that sometimes the training dataset does not describe the 'underlying nature' of the data. The training dataset is obtained so that there are infinite number of linear classifiers that could classify the training dataset correctly.

First, we will run our algorithm and train the multinomial logistic regression on training dataset and compare their performance. We run our algorithm for the number of MCMC iterations $M = 3000$ and 'burn-in' period $M_0 = 1000$ for different parameters of regularization a and standard deviation σ.

Table 7.1 shows the total loss of our algorithm on training dataset. Low values of losses are achieved with small regularization parameters a and large standard deviation σ. Very small values of σ lead to big losses as the algorithm is not able to reach the area of high values of density function $f_{\mathcal{P}}$.

Table 7.2 illustrates the acceptance ratio of new sampling parameters of our algorithm. Large values of σ and large values of regularization parameter a result in low acceptance ratios. With large values of σ we move faster to the area of high values of density function while smaller values of σ can lead to more expensive computations as our algorithm would require more iterations to find the optimal parameters. Figure 7.3 illustrates logarithm of parameters likelihood $q(\theta)$ defined in (7.12) for $a = 0.001$ and $\sigma = 0.1$ and 1.5. We can see from the graphs that for $\sigma = 1.5$ the algorithm reaches maximum value of log-likelihood quite fast while for $\sigma = 0.1$ it still tries to find maximum value after 3000 iterations. It is important to keep track on the acceptance

[1] The code written in R is available at https://github.com/RaisaDZ/LogisticRegression.

ratio of the algorithm, as high acceptance ratio means that we move too slowly and need more iterations and larger 'burn-in' period to find the optimal parameters.

Now we want to demonstrate the 'power' of online learning compared to batch learning. We train the multinomial logistic regression on training dataset and will compare its performance with our algorithm applied to test dataset. We choose parameters of algorithm to be $M = 3000$, 'burn-in' period $M_0 = 1000$, regularization parameter $a = 0.001$ and standard deviation $\sigma = 1.5$. Note, that even though we use the prior knowledge about optimal parameters of our algorithm using results on training dataset, we do not actually train our algorithm, and start with initial value $\theta_0 = 0$. Figure 7.4 shows the difference between cumulative losses of the multinomial logistic regression and our algorithm $L_T^{\theta^*} - L_T$ on test dataset, where θ^* was obtained by multinomial logistic regression model on training dataset. We can see from the graph that our algorithm needs a little time to train and after a few steps it becomes better than multinomial logistic regression trained on training dataset. It is obvious from Figure 7.2 that there are infinite number of linear classifiers that could classify data correctly as training dataset contains linearly separable classes. Training dataset does not describe the 'underlying nature' of the generated data. As a result, retrospectively best model that was trained on training dataset does not perform good on test dataset.

Now we will train multinomial logistic regression on test dataset to find retrospectively the best model with parameters θ^*. Figure 7.5 shows the difference between cumulative losses of retrospectively best expert θ^* on test dataset and the cumulative loss of our algorithm. We also plot the theoretical bound for our algorithm. The initial large gap corresponds to the value $-a\|\theta^*\|_2^2$, which gives the initial start to Learner on expert θ^*. As time increases, we add an additional value $-\frac{n(d-1)}{2}\ln(1 + \frac{d-1}{8a}X^2T)$ to the bound. We can see from the graph that initially the loss difference is decreasing fast which means that loss of our algorithm becomes larger compared to the loss of multinomial logistic regression model. The initial start $-a\|\theta^*\|_2^2$ gives us some time for training. After the initial training time passes, the difference between cumulative losses becomes smoother and behaves in a similar way with the theoretical bound of our algorithm which is decreasing logarithmically with the number of steps.

7.7.2 Glass Identification Dataset

We conduct similar experiments on Glass Identification dataset which is the part of the library 'mlbench' in R or could be downloaded from UCI Machine Learning Repository. The goal is to classify the six type of glasses. The study of classification of types of glass was motivated by criminological investigation. At the scene of the crime, the glass left can be used as evidence. The dataset contains nine features and total 214

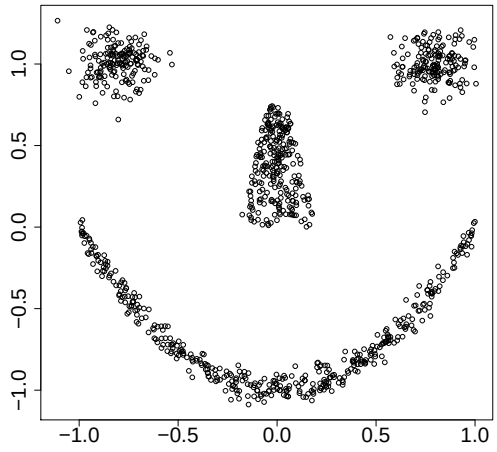


Figure 7.1: Smiley dataset

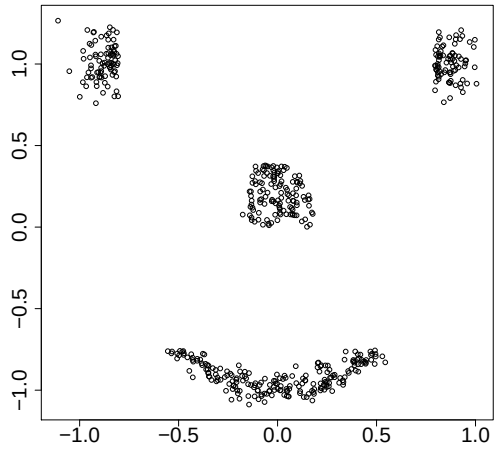


Figure 7.2: Training dataset

Table 7.1: Total Losses of our algorithm on training set

$a \setminus \sigma$	0.1	0.5	1.0	1.5	3.0
0.001	218.52	2.11	0.82	0.68	1.10
0.005	220.23	2.31	2.24	2.05	1.44
0.010	219.93	2.93	2.95	3.18	3.89
0.050	207.49	9.99	10.06	10.21	7.65
0.100	226.51	16.72	16.50	22.15	7.30
0.500	214.10	54.74	54.05	71.29	307.79
0.700	207.18	69.12	73.36	65.90	312.76
1.000	222.63	86.55	99.73	79.63	278.83

observations. As there were no timestamps in the dataset, observations were randomly shuffled, and this order was used as a time. We normalise all the features between -1 and 1 and add addition bias 1 to all observations.

Similar to the previous experiment, we find retrospectively the best multinomial logistic regression with parameters θ^* using the whole dataset. We want to compare the performance of retrospectively best expert θ^* with the performance of our algorithm.

Now we will show how the performance of our algorithm and the behaviour of the loss bound depend on different parameters of regularization a. We choose number of steps $M = 3000$, 'burn-in' period $M_0 = 1000$ and $\sigma = 0.1$. First, we run our algorithm for small regularization $a = 0.001$. Figure 7.6 shows the difference between cumulative losses of multinomial logistic regression and our algorithm. Small values of regularization gives small start on the initial parameters $-a\|\theta^*\|_2^2$ at time $t = 0$.

Table 7.2: Acceptance ratio of our algorithm on training set

$a \setminus \sigma$	0.1	0.5	1.0	1.5	3.0
0.001	0.82	0.74	0.58	0.38	0.03
0.005	0.81	0.75	0.39	0.10	0.01
0.010	0.81	0.70	0.29	0.05	0.00
0.050	0.82	0.56	0.08	0.01	0.00
0.100	0.80	0.46	0.05	0.01	0.00
0.500	0.80	0.24	0.01	0.01	0.00
0.700	0.82	0.21	0.01	0.01	0.00
1.000	0.82	0.16	0.01	0.00	0.00

However, the theoretical bound will grow faster with time $-\frac{n(d-1)}{2}\ln(1+\frac{d-1}{8a}X^2T)$ as it is inversely proportional to the logarithm of the regularization parameter a.

We will conduct the second experiment for larger regularization $a = 0.01$. Figure 7.7 shows the difference between cumulative losses of logistic regression and our algorithm $L_T^{\theta^*} - L_T$. For larger regularization we allow larger initial start on the parameters $-a\|\theta^*\|_2^2$. However, the theoretical bound decreases slower with time compared to the previous experiment.

The choice of the regularization parameter a is important as it affects the behaviour of the theoretical bound of our algorithm. Larger parameters of regularization gives larger start on the parameters of the best model, however the theoretical bound will have smaller growth rate as time increases.

7.7.3 Football Dataset

The third dataset was compiled from historical information on football matches and bookmakers odds [2]. The dataset covers three seasons, 2014/ 2015, 2015/2016 and 2016/2017 of the English Premier League and total 1140 matches. Each match can have three outcomes: 'home win', 'draw', or 'away win'. The data contains the historical information such as total number of goals, shots, corners, yellow and red cards after half-time and full-time and bookmakers' odds from different providers. For each team we generated features such as average number of games won / lost, average number of goals scored / conceded, average number of shots during the first-half, etc. In addition, we combined the odds of different bookmakers provided for the current match. There were total 46 generated features. The first two seasons were used for the training of multinomial logistic regression and the last season was left for test. We want to check if our algorithm could perform close to the model of logistic regression that will

[2] Available at http://football-Data.co.uk

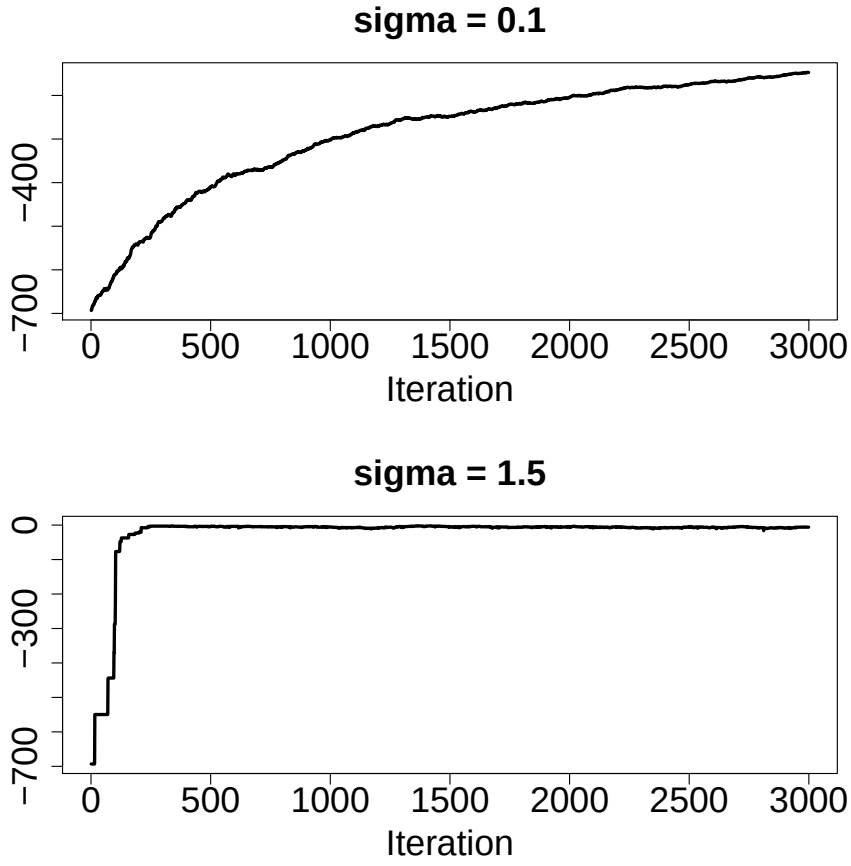


Figure 7.3: Log-likelihood of parameters depending on iteration step

be trained in online mode. We choose the parameters of our algorithm $M = 2000$, 'burn-in' period $M_0 = 500$, regularization parameter $a = 0.05$ and standard deviation $\sigma = 0.2$. At the initial step multinomial logistic regression uses the parameters of the model that was trained on the first two seasons. After that, we add data sequentially and re-train the model after each match. Figure 7.8 illustrates the difference between cumulative losses of multinomial logistic regression trained online and our algorithm $L_T^{\theta^*} - L_T$. Initially our algorithm performs much worse than logistic regression in online mode as the difference of cumulative losses decreases fast. However, after around 200 steps the difference of cumulative losses stabilizes and becoming more 'flattened' which indicates that the performance of our algorithm becomes close to the performance of logistic regression.

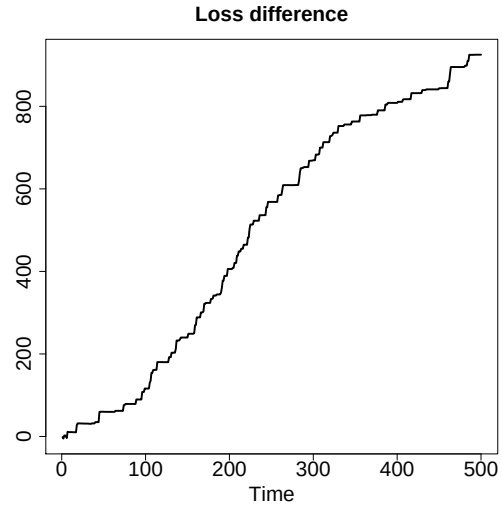


Figure 7.4: Comparison with logistic regression trained on training set

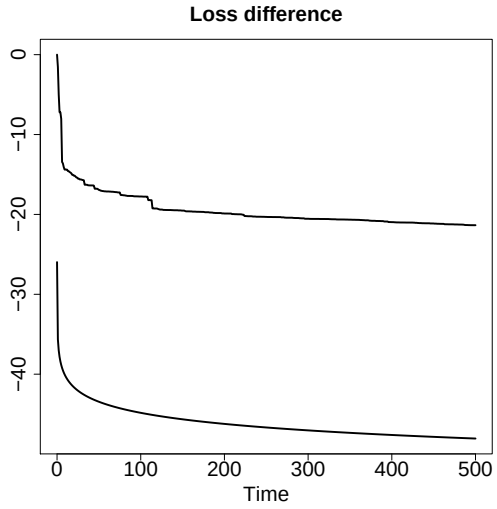


Figure 7.5: Comparison with retrospectively best logistic regression

7.7.4 Conclusions

We carry out the experiments on three datasets to evaluate the performance of our algorithm. Results show that our algorithm could perform close to the best multinomial logistic regression trained in online mode. We also compare the difference between the cumulative losses of retrospectively best multinomial logistic regression and our algorithm, and we check that the theoretical bound of our algorithm is not violated.

The choice of the regularization parameter a is important as it affects the behaviour of the theoretical bound of our algorithm. Larger parameters of regularization gives larger start on the parameters θ^* of the best model, however the theoretical bound will have smaller growth rate as time increases. The choice of the regularization parameter depends on the particular task and goals that desired to be achieved.

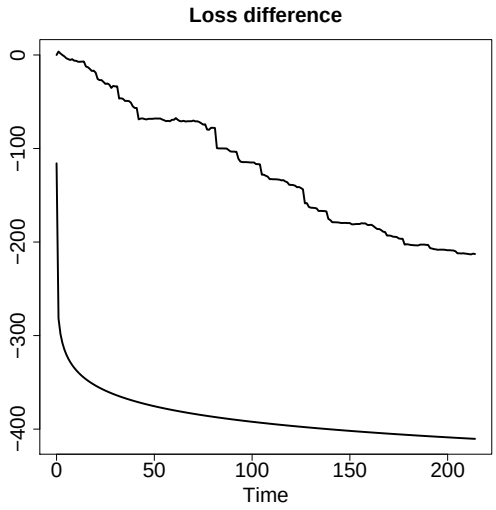


Figure 7.6: Glass dataset, $a = 0.001$

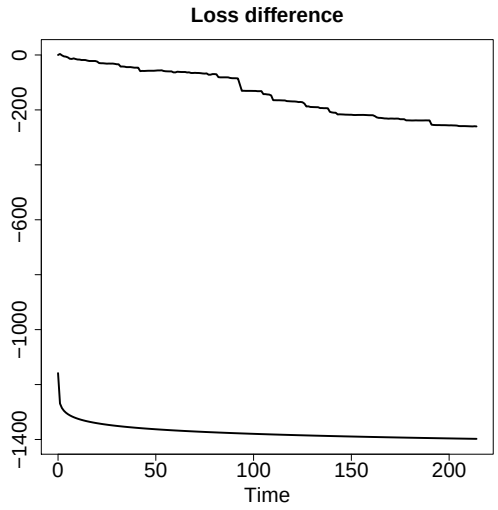


Figure 7.7: Glass dataset, $a = 0.01$

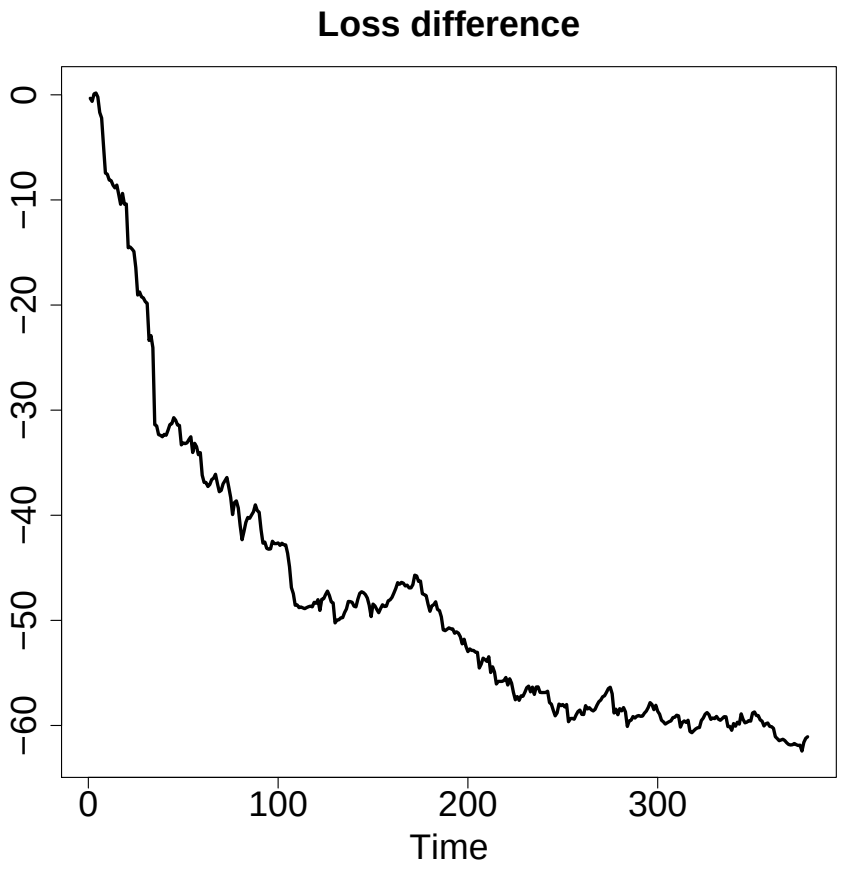


Figure 7.8: Comparison with multinomial logistic regression trained in online mode

Chapter 8

Conclusions

In this thesis, we have considered the problem of competitive online prediction. At each time step, we have an access to experts' predictions, and our task is to make a prediction before seeing the actual outcome. One of the methods which optimally merges pools of experts is the Aggregating Algorithm (AA). It resolves the problem of predicting as well as the best expert up to an additive constant for the case of mixable loss functions. For non-mixable losses, it is possible to use the Weak Aggregating Algorithm (WAA) which provides a weaker theoretical guarantee compared to the AA. However, the WAA still provides a strategy to predict asymptotically not much worse than the best expert in the pool. In this dissertation, we applied both algorithms to different sets of experts depending on the mixability of the considered loss function. In some tasks we considered the generalisation of the standard framework of prediction with expert advice by adding the discounting. The discounted loss is considered to be an alternative to the 'tracking the best expert' framework.

We start with the application of the AA to the prediction of vector-valued outcomes. We develop the theory of prediction with expert advice for packs and generalise the concept of mixability for the special case of delayed feedback. We propose three merging strategies for the prediction of packs and prove the tight worst case upper bounds on the cumulative losses. Experiments on house price and sports datasets discover the properties of the proposed algorithms and compare them with Parallel Copies of the AA. There are two main advantages of the new algorithms compared to Parallel Copies of the AA. First, these algorithms are order-independent, i.e., they do not depend on the order of predictions in the pack. Second, they require less memory to compute as only one array of experts' weights has to be maintained at each time step.

Another application of the framework of prediction with expert advice was considered in the problem of forecasting of Value at Risk (VaR). We consider the game with

pinball loss. The same loss function is used to optimize the parameters of quantile regression. As the pinball loss function is not mixable we apply the WAA to a finite number of models which are used for estimation of VaR. The experimental results on three stocks show that combining predictions of different models can provide better results compared to a single model. The Kupiec unconditional coverage test and the Christoffersen conditional coverage test show that the proposed method is the only one which fails to reject the null hypothesis for all test cases.

The second part of the dissertation is devoted to the development of probabilistic forecasting algorithms, which are competitive with a large class of functions. It is possible to provide good theoretical guarantees even if the decision pool is infinite. While the bound for the finite case can be straightforwardly applied to finite or countable sets of experts, every case of a continuous pool needs to be dealt with separately. The first competitive probabilistic forecasting algorithm provides prediction intervals so that outcomes lie in the interval with a given probability. The algorithm merges the class of quantile regressions and competes in terms of the cumulative pinball loss function. The second proposed algorithm outputs probabilistic predictions in terms of the cumulative distribution functions and allows us to 'track the best linear regression'. The theoretical bound on the discounted cumulative Continuously Ranked Probability Score loss function of the algorithm is derived. Both algorithms were applied to the tasks of renewable energy forecasting. Empirical evaluation suggests that, in general, the new methods perform close to or better than the best quantile regression model in terms of the respective loss functions.

An important class of games of prediction are probability forecasting games, where the predictions and outcomes are probability distributions on some finite set. We propose a competitive prediction strategy for the Kullback-Leibler game, which is one of the most important probability games. We choose experts to be a wide class of multinomial logistic regression functions. We provide a strategy that allows us to 'track the best expert' of this type and derive the theoretical bound on the discounted loss of the strategy. We provide the kernelized version of our algorithm, which competes with a wider set of experts from Reproducing Kernel Hilbert Space (RKHS) and prove a theoretical guarantee for the kernelized strategy. Experimental results on three datasets show that our algorithm could outperform the retrospectively best model of multinomial logistic regression trained on training dataset. We also compare the difference between the cumulative losses of retrospectively best multinomial logistic regression trained on the test dataset and our algorithm, and we check that the theoretical bound of our algorithm is not violated.

Bibliography

Adamskiy, D., Koolen, W. M., Chernov, A., and Vovk, V. (2016). A closer look at adaptive regret. *The Journal of Machine Learning Research*, 17(23):1–21. 31, 44

Alessandrini, S., Delle Monache, L., Sperati, S., and Cervone, G. (2015). An analog ensemble for short-term probabilistic solar power forecast. *Applied Energy*, pages 157: 95–110. 81

Andrieu, C., de Freitas, N., Doucet, A., and Jordan, M. I. (2003). An introduction to MCMC for machine learning. *Machine Learning Journal*, page 50:5–43. 86

Azoury, K. S. and Warmuth, M. K. (2001). Relative loss bounds for on-line density estimation with the exponential family of distributions. *Machine Learning*, 43:211–246. 10

Barton, J. P. and Infield, D. G. (2004). Energy storage and its use with intermittent renewable energy. *IEEE Transactions on energy conversion*, pages 19: 441–448. 80

Beckenbach, E. F. and Bellman, R. (1961). *Inequalities*. Springer, Berlin. 126, 131

Bellotti, T. (2017). Reliable region predictions for automated valuation models. *Annals of Mathematics and Artificial Intelligence*, pages 1–14. 48

Bera, A. K. and Park, S. Y. (2008). Optimal portfolio diversification using the maximum entropy principle. *Econometric Reviews*. 117

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, pages 31(3):307–327. 64

Breiman, L. (2001). Random forests. *Machine Learning, 45*, pages 5–32. 48

Butler, J. and Schachter, B. (1996). Estimating value-at-risk with a precision measure by combining kernel estimation with historical simulation. *Review of Derivatives Research*, pages 1(4):371–390. 63

Cesa-Bianchi, N. and Lugosi, G. (2006). *Prediction, Learning, and Games*. Cambridge University Press. 10, 32

Chernov, A. and Zhdanov, F. (2010). Prediction with expert advice under discounted loss. In *Proccedings of ALT 2010*, volume LNAI 6331, pages 255–269. Springer. See also arXiv:1005.1918 [cs.LG]. 24, 100

Christoffersen, P. (1998). Evaluating interval forecasts. *International Economic Review*, page 39(4):841. 64

Courant, R. and John, F. (1989). *Introduction to Calculus and Analysis*. Springer, New York. 123

Cover, T. and Ordentlich, E. (1996). Universal portfolios with side information. *IEEE Transactions on Information Theory*, 42(2). 10

Cover, T. M. and Thomas, J. A. (2006). Elements of information theory. *John Wiley and Sons, Inc., Second Edition*. 116, 118

De Cock, D. (2011). Ames, Iowa: Alternative to the Boston housing data as an end of semester regression project. *Journal of Statistics Education*, 19(3). 47

Dzhamtyrova, R. and Kalnishkan, Y. (2019). Competitive online generalised linear regression with multidimensional outputs. In *Proceedings of International Joint Conference on Neural Networks*, pages 1–8. IEEE. 118

Foster, D. G., Kale, S., Luo, H., Mohri, M., and Sridharan, K. (2018). Logistic regression: The importance of being improper. *Proceedings of Machine Learning Research vol, 75*, pages 75: 1–42. 116

Freund, Y. and Hsu, D. (2008). A new hedging algorithm and its application to inferring latent random variables. *Technical report, arXiv:0806.4802 [cs.GT], arXiv.org e-Print archive*. 100

Gaglianone, W. P., Lima, L. R., and Linton, O. (2008). Evaluating value-at-risk models via quantile regressions. *Banco Central Do Brasil*. 64

Gammerman, A., Kalnishkan, Y., and Vovk, V. (2004). On-line prediction with kernels and the complexity approximation principle. In *Uncertainty in Artificial Intelligence, Proceedings of the Twentieth Conference*, pages 170–176. AUAI Press. 11

Gneiting, T. and Katzfuss, M. (2014). Probabilistic forecasting. *Annual Review of Statistics and Its Application*, pages 1: 125–151. 80

Guldimann, T. (1995). Riskmetrics TM. *New York [N.Y.]: JPMorgan, p.3*. 63

Harville, D. A. (1997). *Matrix Algebra From a Statistician's Perspective*. Springer. 125, 147

Hazan, E. (2016). *Introduction to Online Convex Optimization*. nowpublishers.com. 116

Herbster, M. and Warmuth, M. K. (1998). Tracking the best expert. *Machine Learning*, 32:151–178. 100

Horn, R. A. and Johnson, C. R. (1985). *Matrix analysis*. Cambridge University Press. 125, 148

Hull, J. C. (2006). *Options, Futures, and Other Derivatives*. Prentice Hall, 6th edition. 64, 65, 68, 70

Joulani, P., Gyorgy, A., and Szepesvári, C. (2013). Online learning under delayed feedback. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 1453–1461. 32, 34

Kakade, S. M. and Ng, A. Y. (2005). Online bounds for bayesian algorithms. *Advances in Neural Information Processing Systems 17*, pages 641–648. 116, 119

Kalnishkan, Y., Adamskiy, D., Chernov, A., and Scarfe, T. (2015). Specialist experts for prediction with side information. In *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, pages 1470–1477. IEEE. 32, 49, 55

Kalnishkan, Y., Vovk, V., and Vyugin, M. V. (2004). Loss functions, complexities, and the Legendre transformation. *Theoretical Computer Science*, 313(2):195–207. 37

Kalnishkan, Y. and Vyugin, M. (2008). The weak aggregating algorithm and weak mixability. *Journal of Computer and System Sciences*, pages 74: 1228–1244. 29, 101

Kivinen, J. and Warmuth, M. (2001). Relative loss bounds for multidimensional regression problems. *Advances in Neural Information Processing Systems 17*, page 301–329. 116

Koenker, R. (2005). Quantile regression. *Cambridge, UK: Cambridge Univ. Press*. 65, 80

Koenker, R. and Bassett, G. (1978). Regression quantiles. *Econometrica*, pages 46: 33–50. 64, 65, 81, 88, 100

Kupiec, P. (1995). Techniques for verifying the accuracy of risk measurement models. *Journal of Derivatives*, pages 3(2):73–84. 64, 74

Levina, T., Levin, Y., McGill, J., Nediak, M., and Vovk, V. (2010). Weak aggregating algorithm for the distribution-free perishable inventory problem. *Operations Research Letters*, pages 38: 516–521. 29, 81

Littlestone, N. and Warmuth, M. K. (1994). The Weighted Majority Algorithm. *Information and Computation*, 108:212–261. 32

Loève, M. (1977). *Probability Theory I.* Springer, 4th edition. 37, 100, 121

Maasoumi, E. (1986). The measurement and decomposition of multidimensional inequality. *Econometrica*, pages 54:991–997. 117

Matheson, J. E. and Winkler, R. L. (1976). Scoring rules for continuous probability distributions. *Management Science*, page 22:1087–96. 99

Meinshausen, N. (2006). Quantile regression forests. *Journal of Machine Learning Research, 7*, pages 983–999. 88

Morgan, J. P. (1996). Riskmetrics technical document. *New York*. 63

Nagya, G. I., Barta, G., Kazia, S., Borbelyb, G., and Simon, G. (2016). Gefcom2014: Probabilistic solar and wind power forecasting using a generalized additive tree ensemble approach. *International Journal of Forecasting*, pages 32: 1087–1093. 81, 88, 100

Perchet, V., Rigollet, P., Chassang, S., and Snowberg, E. (2016). Batched bandit problems. *The Annals of Statistics, 44, no. 2*, pages 660–681. 31

Quanrud, K. and Khashabi, D. (2015). Online learning with adversarial delays. In *Advances in Neural Information Processing Systems*, pages 1270–1278. 32

Roberts, G. O. and Smith, A. F. M. (1993). *Simple conditions for the convergence of the Gibbs sample and Metropolis-Hastings algorithms*. Stoch. Processes Appl., 49. 81

Rokach, L. (2010). Ensemble-based classifiers. *Artificial Intelligence Review*, pages 33:1–39. 64

Taylor, J. W. (1999). A quantile regression approach to estimating the distribution of multiperiod returns. *Journal of Derivatives*, pages 7: 64–78. 64

Theil, H. (1967). Economics and information theory. *Rand McNally, Chicago*. 117

Ullah, A. (1996). Entropy, divergence and distance measures with econometric applications. *Journal of Statistical Planning and Inference, Elsevier*. 117

Vovk, V. (1990). Aggregating strategies. In *Proceedings of the 3rd Annual Workshop on Computational Learning Theory*, pages 371–383, San Mateo, CA. Morgan Kaufmann. 10, 14, 16

Vovk, V. (1998). A game of prediction with expert advice. *Journal of Computer and System Sciences*, 56:153–173. 14, 31, 43, 118

Vovk, V. (2001). Competitive on-line statistics. *International Statistical Review*, 69(2):213–248. 10, 14, 15, 16, 19, 24, 116, 120

Vovk, V. (2005). On-line regression competitive with reproducing kernel hilbert spaces. *Technical report, arXiv:cs/0511058 [cs.LG]*. 131

Vovk, V. and Zhdanov, F. (2009). Prediction with expert advice for the Brier game. *Journal of Machine Learning Research*, 10:2445–2471. 31, 46, 47, 49

V'yugin, V. and Trunov, V. (2019). Online learning with continuous ranked probability score. In *Machine Learning Research (105), Proceedings of the 8th Symposium on Conformal and Probabilistic Prediction with Applications*. 99, 104

Weinberger, M. J. and Ordentlich, E. (2002). On delayed prediction of individual sequences. *IEEE Transactions on Information Theory*, 48(7):1959–1976. 32

Williams, D. (1991). *Probability with Martingales*. Cambridge University Press. 100, 103

Zhdanov, F. (2011). Theory and applications of competitive prediction. *PhD thesis*. 19

Zhdanov, F. and Kalnishkan, Y. (2010). Linear probability forecasting. In *Artificial Intelligence Applications and Innovations, AIAI-2010, Proceedings*, volume 339 of *IFIP Advances in ICT*, pages 4–11. Springer. 116

Zhdanov, F. and Kalnishkan, Y. (2013). An identity for kernel ridge regression. *Theoretical Computer Science*, 473:157–178. 11

Zhdanov, F. and Vovk, V. (2010). Competitive online generalized linear regression under square loss. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pages 531–546. 81, 116

Zinkevich, M. (2003). Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 928–936. 32

Appendix A

Appendix

Lemma A.1. *Let $Q(\theta) = \theta' A\theta + b'\theta + c$, where $\theta, b \in \mathbb{R}^n$, c is a scalar, A is a symmetric positive definite $n \times n$ matrix. Then*

$$\int_{\mathbb{R}^n} e^{-Q(\theta)} d\theta = e^{-Q_0} \frac{\pi^{n/2}}{\sqrt{\det A}},$$

where $Q_0 = \min_{\theta \in \mathbb{R}^n} Q(\theta)$.

The lemma is proven in (Theorem 15.12.1, Harville (1997)).

Lemma A.2. *Let*

$$F(a, b, z) = \min_{\theta \in \mathbb{R}^n} (\theta' A\theta + b'\theta + z'\theta) - \min_{\theta \in \mathbb{R}^n} (\theta' A\theta + b'\theta - z'\theta),$$

where $b, z \in \mathbb{R}^n$ and A is a symmetric positive definite $n \times n$ matrix. Then $F(A, b, z) = -b' A^{-1} z$.

Proof. The first minimum is achieved at $\theta_1 = -\frac{1}{2} A^{-1}(b + x)$, and the second minimum is achieved at $\theta_2 = -\frac{1}{2} A^{-1}(b - x)$. The substitution proves the lemma. □

Theorem A.3. *Let A is positive symmetric semidefinite block matrix such as*

$$A = \begin{pmatrix} A_{1,1} & A_{1,2} & \dots & A_{1,d} \\ A_{2,1} & A_{2,2} & \dots & A_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ A_{d,1} & A_{d,2} & \dots & A_{d,d} \end{pmatrix},$$

where $A_{i,i}$, $i = 1, \dots, d$ are square matrices. Then $A_{i,i}$, $i = 1, \dots, d$ are positive semidefinite and $\lambda_{\max}(A) \leq \sum_{i=1}^{d} \lambda_{\max}(A_{i,i})$.

Proof. Let A be an $n\times n$-matrix and $A_{i,i}$ be an $n_i\times n_i$-matrix, $i=1,\dots,d$. Every vector $x\in\mathbb{R}^n$, $\|x\|=1$ can be partitioned as $x=(x_1',\dots,x_d')'$, where $x_i\in\mathbb{R}^{n_i}$, $i=1,\dots,d$. Define c_i and u_i by $x_i=\|x_i\|\cdot\frac{x_i}{\|x_i\|}=c_iu_i$, where $\sum_{i=1}^d c_i^2=\sum_{i=1}^d\|x_i\|^2=1$, and $\|u_i\|=\frac{x_i}{\|x_i\|}=1$ for $i=1,\dots,d$. If $x_i=0$ we put $c_i=0$ and u_i be any vector such that $\|u_i\|=1$ for $i=1,\dots,d$.

We have

$$\lambda_{\max}(A)=\max_{\|x\|=1}x'Ax$$

and

$$x'Ax=\sum_{i,j:x_i,x_j\neq 0}x_i'A_{i,j}x_j=\sum_{i,j}c_i'u_i'A_{i,j}u_jc_j=\sum_{i,j}c_i'\tilde{a}_{i,j}c_j, \tag{A.1}$$

where $\tilde{a}_{i,j}=u_i'A_{i,j}u_j$ and

$$\tilde{A}=\begin{pmatrix}\tilde{a}_{1,1}&\dots&\tilde{a}_{1,d}\\ \vdots&\ddots&\vdots\\ \tilde{a}_{d,1}&\dots&\tilde{a}_{d,d}\end{pmatrix}.$$

Matrices $A_{i,i}$, $i=1,\dots,d$ are positive semidefinite (by Observation 7.1.2 in Horn and Johnson (1985)) and $\tilde{A}$ is positive semidefinite by (A.1). Then

$$\sum_{i,j}c_i'\tilde{a}_{i,j}c_j\le\lambda_{\max}(\tilde{A})\le\operatorname{tr}\tilde{A}=\sum_{i=1}^d\tilde{a}_{i,i}=\sum_{i=1}^d u_i'A_{i,i}u_i\le\sum_{i=1}^d\lambda_{\max}(A_{i,i}).$$

□

Lemma A.4. (Sylvester Identity) *For any $n\times m$ matrix B, any $m\times n$ matrix C, and any number a*

$$\det(aI_n+BC)=\det(aI_m+CB),$$

where I_n, I_m are unit matrices $n\times n$ and $m\times m$, respectively.

Proof. It follows from matrix multiplication rules that

$$\begin{pmatrix}I_n&B\\ O&I_m\end{pmatrix}\begin{pmatrix}aI_n+BC&O\\ -C&aI_m\end{pmatrix}=\begin{pmatrix}aI_n&aB\\ -C&aI_m\end{pmatrix}$$
$$=\begin{pmatrix}aI_n&O\\ -C&aI_m+CB\end{pmatrix}\begin{pmatrix}I_n&B\\ O&I_m\end{pmatrix}.$$

Taking the determinant of both sides and using rules of taking the determinant of block matrices we get the statement of the lemma. □

Lemma A.5. (Representer Theorem) *Let $\Phi : \mathbf{X} \to H$ be a mapping into a Hilbert space H and $\alpha : H \to \mathbb{R}$ be given by*

$$\alpha(h) = \delta(\langle h, \Phi(x_1)\rangle, \langle h, \Phi(x_2)\rangle, \ldots, \langle h, \Phi(x_T)\rangle, \|h\|),$$

where δ is a function from $\mathbb{R}^{T+1}$ to $\mathbb{R}$ non-decreasing in the last argument and $x_1, x_2, \ldots, x_T \in \mathbf{X}$ are some fixed elements. Then for every $h \in H$ there is a linear combination $h' = \sum_{t=1}^{T} a_t \Phi(x_t)$, where $a_t \in \mathbb{R}$ are constants, such that $\alpha(h') \leq \alpha(h)$. If δ is strictly increasing in the last argument and h does not itself have the form $\sum_{t=1}^{T} a_t \Phi(x_t)$, there is a linear combination h' such that $\alpha(h') < \alpha(h)$.

Proof. The proof is by observing that the projection h' of h on the subspace $\operatorname{span}(\{\Phi(x_1), \Phi(x_2), \ldots, \Phi(x_T)\})$ has the same scalar products $\langle h', \Phi(x_t)\rangle = \langle h, \Phi(x_t)\rangle$ with elements $\Phi(x_t)$, $t = 1, 2, \ldots, T$ and a smaller norm $\|h'\| \leq \|h\|$. □