Theory and Applications of Competitive Prediction

Fedor Zhdanov

Computer Learning Research Centre and
Department of Computer Science,
Royal Holloway, University of London,
United Kingdom

2011

A dissertation submitted in fulfilment of the degree of Doctor of Philosophy.

Declaration

I declare that this dissertation was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

Fedor Zhdanov

Supervisors and Examiners

Supervisors:	*Prof Vladimir Vovk* and *Prof Alex Gammerman*
Internal Examiner:	*Prof Glenn Shafer*
External Examiner:	*Prof Vladimir V'yugin*

Abstract

Predicting the future is an important purpose of machine learning research. In online learning, predictions are given sequentially rather than all at once. People wish to make sensible decisions sequentially in many situations of everyday life, whether month-by-month, day-by-day, or minute-by-minute.

In competitive prediction, the predictions are made by a set of experts and by a learner. The quality of the predictions is measured by a loss function. The goal of the learner is to make reliable predictions under any circumstances. The learner compares his loss with the loss of the best experts from the set and ensures that his performance is not much worse.

In this thesis a general methodology is described to provide algorithms with strong performance guarantees for many prediction problems. Specific attention is paid to the square loss function, widely used to assess the quality of predictions. Four types of the sets of experts are considered in this thesis: sets with finite number of free experts (which are not required to follow any strategy), sets of experts following strategies from finite-dimensional spaces, sets of experts following strategies from infinite-dimensional Hilbert spaces, and sets of experts following strategies from infinite-dimensional Banach spaces. The power of the methodology is illustrated in the derivations of various prediction algorithms.

Two core approaches are explored in this thesis: the Aggregating Algorithm and Defensive Forecasting. These approaches are close to each other in many interesting cases. However, Defensive Forecasting is more general and covers some problems which cannot be solved using the Aggregating Algorithm. The Aggregating Algorithm is more specific and is often more computationally efficient.

The empirical performance and properties of new algorithms are validated on artificial or real world data sets. Specific areas where the algorithms can be applied are emphasized.

Acknowledgements

I am grateful to my supervisor Vladimir Vovk for providing ideas, guidance, and help during my PhD. I am also grateful to Alex Gammerman for useful discussions and encouragement. I thank the members of my project: Alexey Chernov and Yuri Kalnishkan. They showed me a new way of approaching problems in science, introduced me to the area of online prediction, and discussed many ideas to solve the problems during the research. I also thank Alexander Shen for his suggestions for introduction.

I thank my family for constant love, support, and their belief in me. I also thank Emma Britton for her great loyalty and patience, they always helped me to move forward. I have great friends and people who worked with me: Dmitry Devetyarov, Brian Burford, Mikhail Dashevskiy, Ilia Nouretdinov, Steven Busuttil, Tim Scarfe, Dmitry Adamskiy. They helped me a lot to look at life and science from different perspectives.

I enjoyed the working environment at the Computer Science Department, and I am proud to be the member of the Computer Learning Research Centre of Royal Holloway, University of London. I would like to thank all the staff for their help and for keeping the department running smoothly. I thank all the members of the Computer Centre Badminton club for keeping me involved in the great sport.

My research was financially supported by the EPSRC grant EP/F002998/1 "Practical Competitive Prediction". It is also supported in part by the ASPIDA grant from the Cyprus Research Promotion Foundation and the Department. I am grateful for the financial support which I received during my studies.

Contents

Chapter 1

Introduction

> Prediction is very difficult,
> especially if it's about the future.
>
> *Niels Bohr*

- Is it going to rain tomorrow?
- Will the price of a stock go up or down next minute?
- Which team will win the upcoming football match?
- Will the ozone concentration in the air rise high tomorrow?

We do not know the answers to these questions. However, we can try to predict using past experiences.

The goal of our studies is to develop learning algorithms for computer for the purpose of making predictions. A computer program is said to *learn* from experience, with respect to some class of tasks and a performance measure, if its performance at these tasks, as measured by the measure, improves with the experience (Mitchell, 1997).

The task may be to predict an outcome which can take only finitely many values. In this case, it can be called a classification problem. If the outcome can take infinitely many values (for example, on the real line), the task can be called a regression problem. Even though classification may be reduced to

a regression problem, the methods which work better for each of these tasks are often different. In this thesis, we approach both regression and multi-class classification, and (separately) two-class classification.

Online setting

We consider the online setting. Online prediction is a wide area of machine learning, Cesa-Bianchi and Lugosi (2006) give a good introduction to this area. Predictions in this setting are given step by step. After each step, the algorithm learns on the new data. This is the contrary to the batch setting, where the algorithm learns only on a subset of the data (called training set), and then predicts on another subset (called test set). For example, suppose the aim is to predict the weather in January. If the data about the weather in December is available, a batch method is trained on the December data, and gives predictions for all the days in January at once. An online method gives predictions for January sequentially, adjusting its parameters every day with updated weather conditions.

The process can be presented in the form of a game between a learner and reality. The learner uses an algorithm which takes some information from reality. Then the learner predicts the next outcome in the sequence. After reality announces the actual outcome, the learner updates parameters of the algorithm. This process is repeated step by step.

Competitive prediction

In statistical learning, statistical assumptions are made about the data generating process, and guarantees are proven for the methods working under these assumptions.

In this thesis, we consider another approach, called competitive prediction. In competitive prediction, one provides guarantees for the algorithms in comparison with other predicting models (we call them experts), instead of making statistical assumptions about the data generating process.

Thus we do not develop methods which predict well under certain circumstances, but try to combine predictions which are given to us by experts. The

goal of competitive prediction is to develop algorithms whose performance is not much worse than the performance of the best experts.

We can combine predictions of experts without knowing how their predictions are obtained. On the other hand, they can be obtained as predictions of a model with different (not necessarily countable) values of parameters.

Dealing with the class of experts

One of the ways to use experts is to choose the expert which performed better than all other experts so far. In case of a model with parameters, this way corresponds to the search of the best parameter for the model.

However, good past performance does not necessarily lead to good future performance. Thus, we choose a different approach in this thesis. All the experts are assigned weights, and the predictions are given considering the weights of all the experts' predictions. The weights are updated step by step to reflect experts' current performance. We use two algorithms to deal with the weights: the Aggregating Algorithm (Vovk, 1990) and the Defensive Forecasting algorithm (Chernov et al., 2010).

A similar approach is used, for example, in Bayesian methods. In the Bayesian approach, the learner chooses a distribution (called prior) over the models, and updates this distribution using the likelihood of the available data, thus obtaining the posterior over the models. Then the learner gives its prediction as the expected prediction of all the models over the posterior distribution.

The Bayesian approach takes into account the average loss of the predicting algorithm. The guarantees on the performance for all our algorithms are given in terms of the loss suffered by them in the worst case scenario, even when the data fail to satisfy any statistical assumptions.

Classes of experts

When the algorithm to combine experts is chosen, it is important what kind of performance guarantees it can provide in comparison with different classes of experts. In this thesis, we primarily concentrate on this problem, and cover

a wide range of sets of experts, from sets with finite number of experts to sets with uncountable number of experts.

It may be difficult to weigh uncountable number of arbitrary experts, so in this case each expert is required to follow a prediction strategy. As an example, we can say that under a prediction model, each expert chooses a particular parameter for the model. For example, each expert may predict according to a linear function of input vectors. Then it is possible to provide guarantees on the performance of the algorithm competing with the linear experts.

An important practical question is "How to predict intervals in time when one expert outperforms another one?" An algorithm competing with different experts does not provide the answer to this question. The question the algorithm answers is "At each moment in time, how to achieve the overall prediction performance which is not much worse than the performance of the best expert at that moment?"

1.1 Original contributions

The following list provides the summary of the main original results achieved during the work on the thesis.

- New algorithms are developed and guarantees on their performance are proven for the following online prediction settings: generalized linear models under the square loss (Algorithm 9 and Theorem 3.5), regression of a variable which is bounded by different constants at each step (Algorithm 8 and Theorem 3.4), regression under discounted square loss (Algorithm 8, Theorem 3.4, and Corollary 3.4), and probability forecasting of multi-dimensional outcomes under the square loss (Algorithm 10, Theorems 3.7 and 3.6).

  For all of the algorithms their generalizations for Hilbert spaces are derived and the performance guarantees for them are proven (Theorem 4.6, Theorem 4.5, and Theorem 4.7 respectively).

- Practical experiments with new and existing algorithms are performed. For prediction the results of sport matches (Section 2.6), it is shown that algorithms which do not have strong guarantees on their performance can fail to be competitive with the best model. For the algorithms which have guarantees, the guarantees are often tight in practice. The experiments with the prediction of ovarian cancer (Section 2.7) showed that combining different predictors in the online setting can improve the reliability of the predictions at different stages before diagnosis.

  Areas of applications of the new algorithms are investigated, ways to choose their parameters are suggested, and the comparison with other algorithms is performed on artificial and real world data sets (Section 3.5.4, Section 3.4.3, and Section 3.6.4).

- An elegant method is used to answer the question whether the Ridge Regression algorithm has guarantees on its square loss in the online setting; the guarantee is proven in the form of equality (Theorem 3.3). It leads to the guarantee in the more standard form. The method follows from

the proof of the guarantees on the logarithmic loss of the Bayesian Ridge Regression algorithm (and its generalization for Hilbert spaces version) in the setting of regression of a variable with the presence of the Gaussian noise (Theorem 3.2 and Theorem 4.3).

Guarantees on the performance of the algorithms are proven using a new methodology (the core elements of the proofs are Lemma 3.2 and Lemma 4.1). The following is a list of other interesting results achieved during the work on the thesis.

- A way to compute efficiently predictions of the Defensive Forecasting algorithm in many cases is suggested (Equation (2.14)).
- It is found that the Aggregating Algorithm can be applied to compete with second-guessing experts under the square and logarithmic loss using a fixed point property (Theorem 2.9).
- New algorithm competing with functions from functional Banach spaces is developed. Guarantees on its performance are proven in semi-online prediction setting. This is done by considering abstract linear regression (Algorithm 12, Theorems 5.1 and 5.4).

1.2 Publications

Many of the results described in this thesis are published either as conference proceedings, or journal papers, or summarized as technical reports (arXiv submissions are not listed below). The paper "Linear probability forecasting" received the best paper award.

1. Fedor Zhdanov and Yuri Kalnishkan. An identity for Kernel Ridge Regression. In *Proceedings of the 21st International Conference on Algorithmic Learning Theory*, pages 405–419, 2010

2. Fedor Zhdanov and Vladimir Vovk. Competitive online generalized linear regression under square loss. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases 2010*, pages 531–546, 2010.

3. Alexey Chernov and Fedor Zhdanov. Prediction with expert advice under discounted loss. In *Proceedings of the 21st International Conference on Algorithmic Learning Theory*, pages 255–269, 2010.

4. Alexey Chernov, Yuri Kalnishkan, Fedor Zhdanov, and Vladimir Vovk. Supermartingales in prediction with expert advice. *Theoretical Computer Science*, 411:2647–2669, 2010.

5. Fedor Zhdanov and Yuri Kalnishkan. Linear probability forecasting. In *Proceedings of the 6th IFIP Conference on Artificial Intelligence Applications and Innovations*, pages 4–11, 2010.

6. Vladimir Vovk and Fedor Zhdanov. Prediction with expert advice for the Brier game. *Journal of Machine Learning Research*, 10:2413–2440, 2009.

7. Fedor Zhdanov, Vladimir Vovk, Brian Burford, Dmitry Devetyarov, Ilia Nouretdinov, and Alex Gammerman. Online prediction of ovarian cancer. In *Proceedings of the 12th Conference on Artificial Intelligence in Medicine*, pages 375–379, 2009.

8. Vladimir Vovk and Fedor Zhdanov. Prediction with expert advice for the Brier game. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1104–1111, 2008.

9. Alexey Chernov, Yuri Kalnishkan, Fedor Zhdanov, and Vladimir Vovk. Supermartingales in prediction with expert advice. In *Proceedings of the 19th International Conference on Algorithmic Learning Theory*, pages 199–213. Springer, Berlin, 2008.

1.3 The structure of the thesis

The thesis contains six chapters. Chapter 1 gives the introduction to the thesis, outlines the original contributions, and describes the organizational structure of it. Chapter 2 describes the general online prediction framework and the problem of competing with a finite number of experts. Chapter 3 is devoted to the problem of competing with the experts who predict according to functions from spaces of finite dimension. In Chapter 4 the algorithms competing with linear experts are generalized to compete with the experts who predict according to functions from Reproducing Kernel Hilbert Spaces. Chapter 5 gives a way to approach the problem of competing with experts in Banach spaces. Finally, Chapter 6 briefly concludes the work and gives directions for the future research.

Chapter 2

Main algorithms and their application to finite number of experts

In this chapter we describe two algorithms which we use in our studies and analyze the basic setting of competitive online prediction.

Section 2.1 and Section 2.2 contain the description of the main algorithms for the general case, without the assumption that the number of experts is finite. We introduce the general framework of competitive online prediction and describe two algorithms which are mainly used in our studies. Section 2.3 shows two basic cases of the use of our algorithms which are further developed and generalzied to cover more interesting problems. We consider the problem of competing under the Brier loss in multi-dimensional setting (Section 2.4) and competing with second-guessing experts (Section 2.5). Sections 2.6 and 2.7 describe experimental evaluation of the algorithms.

2.1 Introduction to competitive online prediction

The area of online prediction that is devoted to algorithms competing with countable (finite or infinite) number of experts is usually called prediction with expert advice. The case when the number of experts is uncountable can be considered as a generalization of the finite case; we call it competitive online regression. The idea of prediction with expert advice is very intuitive and practical: imagine there are some people or organizations which make their predictions about a sequence; they are called experts. The learner makes predictions using their advice, in other words using their predictions.

The first ideas of combining different models for making forecasts probably appeared in Barnard (1963) for predicting airline passenger numbers, and later using experts' advice in Morris (1974). First ideas of competitive online prediction go back to DeSantis et al. (1988) who applied the Bayesian approach to average experts' predictions. Later Littlestone and Warmuth (1994) and Vovk (1990) introduced online algorithms for some other games. We use Vovk's Aggregating Algorithm and the Defensive Forecasting algorithm introduced in Chernov et al. (2008) because these algorithms are known to give very strong theoretical guarantees on the loss of the learner.

Instead of choosing the expert who performed better than others so far, the learner following our algorithms takes into account predictions of all the experts in a certain way. He assigns weights to all the experts. The weights are updated from one step to another to reflect the change in the learner's level of trust to each of the expert's predictions. The performance of the learner and the experts is measured by a loss function. The goal of a good algorithm for the learner is to find a way to combine experts' predictions and to update the weights such that it provides theoretical guarantees on the loss of the learner in comparison with the loss of the best expert.

2.1.1 The framework

We assume that the outcomes lie in an *outcome set* Ω, the experts and the learner predict these outcomes using a *prediction set* Γ and we measure the quality of their predictions using a *loss function* $\lambda : \Omega \times \Gamma \to [0, \infty]$. The triple $(\Omega, \Gamma, \lambda)$ is called the *game*. We denote the index set for the experts by Θ and each individual expert by $\theta \in \Theta$. Then the prediction process follows Protocol 1.

Protocol 1 Competitive online prediction

$L_0 := 0$.
$L_0^\theta := 0$, $\theta \in \Theta$.
for $t = 1, 2, \ldots$ **do**
  Experts announce $\xi_t^\theta \in \Gamma$, $\theta \in \Theta$.
  Learner announces $\gamma_t \in \Gamma$.
  Reality announces $\omega_t \in \Omega$.
  $L_t := L_{t-1} + \lambda(\omega_t, \gamma_t)$.
  $L_t^\theta := L_{t-1}^\theta + \lambda(\omega_t, \xi_t^\theta)$, $\theta \in \Theta$.
end for

Here L_t is the cumulative loss of the learner after the step t, and L_t^θ is the cumulative loss of the expert θ after this step.

We will be interested in deriving upper bounds on the loss of the learner in terms of the loss of the best expert in the form

$$L_T \le \min_{\theta \in \Theta} L_T^\theta + R_T \tag{2.1}$$

for all $T = 1, 2, \ldots$. Here R_T is the so-called *regret term*. It can be thought of as a measure of the quality of the upper bound. If the number of experts is not finite, there is usually an additional term on the right-hand side which expresses the complexity of the strategy of the chosen expert. We are mostly interested in the bounds where the regret term divided by T converges to zero: $R_T = o(T)$.

Example 2.1 (Simple prediction game) Assume that $\Omega = \Gamma = \{0, 1\}$ and

the loss function is $\lambda(\omega, \gamma) = 1 - \delta_\omega^\gamma$, where

$$\delta_\omega^\gamma = \begin{cases} 1, & \text{if } \omega = \gamma \\ 0, & \text{otherwise} \end{cases}$$

is the Kronecker symbol. This is the classification game with zero-one loss.

Suppose that one of the experts is perfect, in other words that there exists θ such that $L_T^\theta = 0$ for all T. Then there is a strategy for the learner to achieve $L_T \leq \log_2 K$, where K is the number of experts. This is easy to prove: the learner follows the majority of the experts who have not been eliminated so far. If he makes a mistake, he eliminates all the experts which he followed at this step. Thus at each step the number of experts at least halves. After the first step it becomes $\leq K/2$, after the second step it becomes $\leq K/4$, etc. The perfect expert is found after at most $\log_2 K$ steps, and then the strategy does not make any mistakes.

If there is no perfect expert, the learner has to be more tolerant of the experts and to reduce their weights instead of completely eliminating them. Theoretical guarantees in this setting were first proven for the so-called the Weighted Majority Algorithm described by Littlestone and Warmuth (1994).

2.2 Description of the algorithms

In this section we describe two algorithms used in our study: the Aggregating Algorithm and the Defensive Forecasting algorithm.

2.2.1 Aggregating Algorithm

One of the algorithms which may be used for online prediction is Vovk's Aggregating Algorithm (AA). It has strong theoretical guarantees in many settings of competitive online prediction. It is possible to prove that the theoretical guarantees given by this algorithm can not be improved in certain cases. The Aggregating Algorithm is given as Algorithm 1. [1]

Algorithm 1 Aggregating Algorithm

Require: An initial weights distribution $P_0(d\theta) = P_0^*(d\theta)$ on the experts, a learning rate $\eta > 0$.

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

Read the experts' predictions ξ_t^θ, $\theta \in \Theta$.

Calculate generalized prediction: a function $g : \Omega \to \mathbb{R}$ on outcomes defined by $g_t(\omega) = -\frac{1}{\eta} \ln \int_\Theta e^{-\eta\lambda(\omega, \xi_t^\theta)} P_{t-1}^*(d\theta)$.

Using a substitution function $\Sigma : \mathbb{R}^\Omega \to \Gamma$ predict $\gamma_t = \Sigma(g_t)$ such that $\lambda(\omega, \gamma_t) \leq g_t(\omega), \forall \omega \in \Omega$.

Read $\omega_t \in \Omega$.

Update the weights $P_t(d\theta) = e^{-\eta\lambda(\omega_t, \xi_t^\theta)} P_{t-1}(d\theta)$.

Normalize the weights $P_t^*(d\theta) = \frac{P_t(d\theta)}{\int_\Theta P_t(d\theta)}$.

end for

The Aggregating Algorithm has two parameters: an initial weights distribution on the experts $P_0(d\theta) \in \mathcal{P}(\Theta)$ from the space $\mathcal{P}(\Theta)$ of all probability distributions on the experts, and a learning rate $\eta > 0$. It is an exponential weights algorithm; in other words, at each step it updates the weights of the

[1] In this thesis, we only work with a special case of the Aggregating Algorithm for so-called mixable loss functions, to be explained below.

experts by the following formula:

$$P_t(d\theta) = \beta^{\lambda(\omega_t, \xi_t^\theta)} P_{t-1}(d\theta) = \beta^{L_t^\theta} P_0(d\theta), \tag{2.2}$$

where $\beta = e^{-\eta} \in [0, 1)$. This weights update ensures that the experts which suffer large loss at some step receive smaller weight for further predictions.

After the Aggregating Algorithm has received the experts' predictions at some step, it combines them in the following minimax way. A *generalized prediction* is any function of the type $\Omega \to \mathbb{R}$. The Aggregating Algorithm chooses its generalized prediction to be the mapping $g_t : \Omega \to \mathbb{R}$ for each possible outcome such that

$$g_t(\omega) = \log_\beta \int_\Theta \beta^{\lambda(\omega, \xi_t^\theta)} P_{t-1}^*(d\theta), \tag{2.3}$$

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

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

We define the *Aggregating Pseudo-Algorithm* (APA) as the algorithm which does not necessarily give permitted predictions, but formal mixtures of predictions. For each possible outcome the APA suffers the loss equal to the generalized prediction.

Finally, the Aggregating Algorithm gives its prediction γ_t using a substitution function. A *substitution function* $\Sigma : \mathbb{R}^\Omega \to \Gamma$ is a function which maps generalized predictions into permitted predictions such that

$$\lambda(\omega, \gamma_t) \le g_t(\omega), \forall \omega \in \Omega \tag{2.4}$$

for $\gamma_t = \Sigma(g)$.

Let us define $\mathcal{P}(\Theta)$ as the set of all probability measures over Θ. If such a substitution function exists for any distribution $P_{t-1}^*(d\theta) \in \mathcal{P}(\Theta)$, we say that the loss function is *η-mixable*. The loss function is *mixable* if it is η-mixable for some $\eta > 0$. The game is called mixable if the loss function of it is mixable in the setting of the game. Some non-mixable games are described and analyzed

in Vovk (1998). In its general form, the AA is capable of competing under non-mixable loss functions. Relative loss bounds for non-mixable games often have the coefficient more than 1 before the loss of the best expert in (2.1).

The next lemma is Lemma 1 from Vovk (2001). It gives an expression for the sum of the generalized predictions evaluated at the actual outcomes in terms of the initial distribution and the loss of each expert. It shows that the cumulative loss of the APA, $L_T(\text{APA}) := \sum_{t=1}^{T} g_t(\omega_t)$, is an average (in a general sense) of the experts' cumulative losses.

Lemma 2.1 (Vovk, 2001, Lemma 1) *For any learning rate $\eta > 0$, initial distribution P_0, and $T = 0, 1, 2, \ldots$,*

$$L_T(\text{APA}) = \log_\beta \int_\Theta \beta^{L_T^\theta} P_0(d\theta). \tag{2.5}$$

PROOF We proceed by induction in T: for $T = 0$ the equality is obvious, and for $T > 0$ we have:

$$\begin{aligned} L_T(\text{APA}) &= L_{T-1}(\text{APA}) + g_T(\omega_T) \\ &= \log_\beta \int_\Theta \beta^{L_{T-1}^\theta} P_0(d\theta) + \log_\beta \int_\Theta \beta^{\lambda(\omega_T, \xi_t^\theta)} \frac{\beta^{L_{T-1}^\theta} P_0(d\theta)}{\int_\Theta \beta^{L_{T-1}^\theta} P_0(d\theta)} \\ &= \log_\beta \int_\Theta \beta^{L_{T-1}^\theta} P_0(d\theta) + \log_\beta \frac{\int_\Theta \beta^{L_T^\theta} P_0(d\theta)}{\int_\Theta \beta^{L_{T-1}^\theta} P_0(d\theta)} \\ &= \log_\beta \int_\Theta \beta^{L_T^\theta} P_0(d\theta). \end{aligned}$$

Here the second equality follows from the inductive assumption, the definition (2.3) of g_T, and (2.2). ∎

Note that the cumulative loss $L_T(\text{AA})$ of the Aggregating Algorithm does not exceed the cumulative loss $L_T(\text{APA})$ of the Aggregating Pseudo-Algorithm, $L_T(\text{AA}) \le L_T(\text{APA})$, for a mixable game. Now notice that

$$P_T^*(d\theta) = \frac{\beta^{L_T^\theta} P_0(d\theta)}{\beta^{L_T(\text{APA})}}$$

and thus for the countable number of experts

$$L_T(\mathrm{AA}) \le L_T(\mathrm{APA}) = L_T^\theta + \log_\beta P_0^\theta - \log_\beta P_T^\theta \tag{2.6}$$

for any $\theta \in \Theta$. Here P_0^θ is the initial weight of the expert θ, and P_T^θ is its normalized weight after the step T. There are two ways which are usually used to prove theoretical guarantees on the loss of the Aggregating Algorithm. The first one is to notice that $P_T^\theta \le 1$ and so

$$L_T(\mathrm{AA}) \le L_T^\theta + \frac{\ln(1/P_0^\theta)}{\eta}$$

for any θ. This approach is usually used for finite or countable classes of experts. The second way is to evaluate the right-hand side of (2.5) and extract the part with the loss of the best expert. This approach is usually used for larger classes of experts.

The only part of the Aggregating Algorithm which we have not explicitly explained yet is the form of the substitution function Σ in the definition of Algorithm 1. Following Vovk (2001) we require that

$$\begin{gathered} \forall g \; \Sigma(g) \in \arg\min_{\gamma\in\Gamma} \sup_{\omega\in\Omega} \bigl(\lambda(\omega,\gamma) - g(\omega)\bigr) \\ \text{and} \\ \forall g^1, \forall g^2 \bigl(\exists c \, \forall \omega \in \Omega \; g^1(\omega) - g^2(\omega) = c\bigr) \Rightarrow \bigl(\Sigma(g^1) = \Sigma(g^2)\bigr). \end{gathered} \tag{2.7}$$

Here $g^1, g^2 \in \Omega \to \mathbb{R}$ are the generalized predictions calculated with different weights distributions. The last implication ensures that if we calculate the generalized prediction with unnormalized weights $P_t(d\theta)$ instead of $P_t^*(d\theta)$, the prediction of the AA will be the same. This property becomes important for infinite classes of experts, since normalization is often computationally inefficient. We will also need this property for the case of multi-dimensional outcomes.

2.2.2 Defensive Forecasting

Another algorithm which we use in our studies is the Defensive Forecasting algorithm (DF). In some games the DF and the AA depend on the same property (2.4) being satisfied. However, the defensive forecasting technique is more flexible and allows us to compete in a larger class of situations. On the other hand, the predictions given by the DF are usually more difficult to compute than the predictions of the AA, especially if the outcome set is a subset of the Euclidean space $\mathbb{R}^n$, $n > 2$.

In defensive forecasting, it is possible to use different outcome and prediction sets Ω_t and Γ_t at each step. In other words, reality announces the outcome set and the prediction set at the beginning of each step before any predictions are made. Each expert θ is allowed to choose a loss function at each step t defined on the Cartesian product of these sets. Let λ_t^θ be the mixable loss function chosen by the expert θ at the step t. The learner wishes to compete with all the experts under their loss functions. The ability to compete under different loss functions at once is currently the main known advantage of defensive forecasting over the AA.

Moreover, it is possible to discount the loss after each step. The cumulative losses of the experts and the learner are discounted with a factor $\alpha_t \in [0, 1]$, at each step. If $\mathcal{L}_{T-1}$ is the discounted cumulative loss of the learner at the step $T-1$, then the discounted cumulative loss of the learner at the step T is defined by

$$\mathcal{L}_T := \alpha_{T-1}\mathcal{L}_{T-1} + \lambda_T^\theta(\omega_T, \gamma_T) = \sum_{t=1}^{T-1} \left(\prod_{i=t}^{T-1} \alpha_i \right) \lambda_t^\theta(\omega_t, \gamma_t) + \lambda_T^\theta(\omega_T, \gamma_T).$$

If $\mathcal{L}_{T-1}^\theta$ is the discounted cumulative loss of the expert θ at the step $T-1$, then the discounted cumulative loss of the expert θ at the step T is defined by

$$\mathcal{L}_T^\theta = \alpha_{T-1}\mathcal{L}_{T-1}^\theta + \lambda_t^\theta(\omega_T, \xi_T^\theta) = \sum_{t=1}^{T-1} \left(\prod_{i=t}^{T-1} \alpha_i \right) \lambda_t^\theta(\omega_t, \xi_t^\theta) + \lambda_T^\theta(\omega_T, \xi_T^\theta).$$

In the beginning the losses $\mathcal{L}_0, \mathcal{L}_0^\theta$ are initialized to zero. If all the discounting factors are the same, $\alpha_1 = \ldots = \alpha_T$, then at each step the dependence on the

loss at the previous steps exponentially decreases: the coefficient in front of λ_1 becomes equal to α^{T-1} at the step T.

For each step t and each expert θ we define the function

$$\begin{aligned} &Q_t^\theta : \Omega_t \times \Gamma_t \to [0, \infty) \\ &Q_t^\theta(\omega, \gamma) := e^{\eta_t^\theta \left(\lambda_t^\theta(\omega,\gamma) - \lambda_t^\theta(\omega, \xi_t^\theta)\right)}. \end{aligned} \tag{2.8}$$

We also define the mixture function

$$Q_T(\omega_1, \gamma_1, \omega_2, \gamma_2, \ldots, \omega_{T-1}, \gamma_{T-1}, \omega, \gamma) := \int_\Theta \left(\prod_{t=1}^{T-1} \left(Q_t^\theta\right)^{\prod_{i=t}^{T-1} \alpha_i} \right) Q_T^\theta P_0(d\theta)$$

with some initial weights distribution $P_0(d\theta)$ on the experts. Here $\omega_1 \in \Omega_1, \ldots, \omega_{T-1} \in \Omega_{T-1}$ are the actual outcomes and $\gamma_1 \in \Gamma_1, \ldots, \gamma_{T-1} \in \Gamma_{T-1}$ are the predictions until the step $T-1$. Here also η_t^θ are learning rate coefficients; they will be described later in the section.

In online prediction one often considers predictions which are probability distributions on outcomes (see, e.g, Dawid, 1986). Defensive Forecasting is based on a correspondence between the set of all probability distributions and the prediction set. In our studies we consider the outcome set and the prediction set which are subsets of the Euclidean space. We also restrict ourselves to the case when the loss functions $\lambda_t^\theta(\omega, \gamma)$ are continuous in γ for any ω. If $\Omega_t = \{Y_{t,1}, Y_{t,2}\}$ for $Y_{t,1}, Y_{t,2} \in \mathbb{R}$ for all t, the set of all probability distributions $\mathcal{P}(\Omega_t)$ over Ω_t can be identified with the interval $[0, 1]$. Each probability measure $\pi^p \in \mathcal{P}(\Omega_t)$ assigns the probability $p \in [0, 1]$ to the outcome $Y_{t,2}$ and the probability $1 - p$ to the outcome $Y_{t,1}$. We define the correspondence

$$\gamma^p = p(Y_{t,2} - Y_{t,1}) + Y_{t,1}, \quad p \in [0, 1], \tag{2.9}$$

between $[0, 1]$ and learner's predictions $\gamma^p \in \Gamma_t$. We will consider the cases $\Gamma_t = [Y_{t,1}, Y_{t,2}]$ and $\Gamma_t = \mathbb{R}$ below.

Standard proofs of the bounds for Defensive Forecasting are based on the fact that the properties of Q_T are very similar to the properties of a game-theoretic supermartingale (see Shafer and Vovk, 2001, p. 82). If the outcome set is the same for all steps, the prediction set is the set of all probability dis-

tributions over the outcome set, and there is no discounting ($\alpha_t = 1$ for all t), Q_T is a function defined on $(\Omega \times \mathcal{P}(\Omega))^T$. In this case it is a supermartingale by the definition of Chernov et al. (2008). It is a function defined on $(\Omega \times \mathcal{P}(\Omega))^*$ such that given any past values of the actual outcomes $\omega_1, \ldots, \omega_{T-1} \in \Omega$ and of the predictions $\pi_1, \ldots, \pi_{T-1} \in \mathcal{P}(\Omega)$, the expectation of the function $Q_T(\omega_1, \pi_1, \ldots, \omega_{T-1}, \pi_{T-1}, \omega, \pi)$ w.r.t. any prediction measure $\pi \in \mathcal{P}(\Omega)$ does not exceed the value of the function $Q_{T-1}(\omega_1, \pi_1, \ldots, \omega_{T-1}, \pi_{T-1})$ at the previous step.

However, we introduce the notion of a defensive property to incorporate more general cases. As we will see later, this property is crucial in helping the learner to defend himself against suffering the large loss (in comparison with the experts). Assume that there are fixed bijections between the spaces $\mathcal{P}(\Omega_t)$ of all probability measures on Ω_t and the set $[0, 1]$. Each $p^\pi \in [0, 1]$ corresponds to some unique $\pi \in \mathcal{P}(\Omega_t)$, and vice versa. We also take the correspondence (2.9) between $[0, 1]$ and each prediction set Γ_t.

Definition 2.1 (Defensive property) A sequence R of functions $R_1, R_2, \ldots$ such that $R_t : \Omega_t \times \Gamma_t \to (-\infty, \infty]$ is said to have *the defensive property* if, for any t and any $\pi \in \mathcal{P}(\Omega_t)$, it holds that

$$\mathrm{E}_\pi R_t(\omega, \gamma^{p^\pi}) \le 1, \tag{2.10}$$

where E_π is the expectation with respect to a measure π.

A sequence R is called *forecast-continuous* if, for all T and all $\omega \in \Omega_T$, the function $R_T(\omega, \gamma)$ is continuous in γ.

The following lemma states the most important for us property of all the sequences having the defensive property. A variant of the lemma was originally proven by Levin (1976). For a full proof in the case when Ω_t, Γ_t do not depend on t see Vovk (2007, Theorem 1). We prove it only for the simple case $\Omega_t = \{Y_{t,1}, Y_{t,2}\}$, $\Gamma_t = \mathbb{R}$ with $Y_{t,1} < Y_{t,2}$ and $Y_{t,1}, Y_{t,2} \in \mathbb{R}$: in our studies we restrict ourselves to this case and its direct generalization with $\Omega_t = [Y_{t,1}, Y_{t,2}]$.

Lemma 2.2 (Levin lemma) *Let R be a forecast-continuous sequence having*

the defensive property. For any t there exists $p \in [0,1]$ such that for all $\omega \in \Omega_t$

$$R_t(\omega, \gamma^p) \le 1.$$

PROOF Define a function $f_t : \Omega_t \times [0,1] \to (-\infty, \infty]$ by

$$f_t(\omega, p) = R_t(\omega, \gamma^p) - 1.$$

Since R is forecast-continuous and the correspondence (2.9) is continuous, $f_t(\omega, p)$ is continuous in p. Since R has the defensive property, we have

$$p f_t(Y_{t,2}, p) + (1-p) f_t(Y_{t,1}, p) \le 0 \tag{2.11}$$

for all $p \in [0,1]$. In particular, $f_t(Y_{t,1}, 0) \le 0$ and $f_t(Y_{t,2}, 1) \le 0$.

Our goal is to show that for some $p_t \in [0,1]$ we have $f_t(Y_{t,1}, p_t) \le 0$ and $f_t(Y_{t,2}, p_t) \le 0$. If $f_t(Y_{t,2}, 0) \le 0$, we can take $p_t = 0$. If $f_t(Y_{t,1}, 1) \le 0$, we can take $p_t = 1$. Assume that $f_t(Y_{t,2}, 0) > 0$ and $f_t(Y_{t,1}, 1) > 0$. Then the difference

$$f_t(p) := f_t(Y_{t,2}, p) - f_t(Y_{t,1}, p)$$

is positive for $p = 0$ and negative for $p = 1$. By the intermediate value theorem, $f_t(p_t) = 0$ for some $p_t \in (0,1)$. By (2.11) we have $f_t(Y_{t,2}, p_t) = f_t(Y_{t,1}, p_t) \le 0$.■

This lemma shows that at each step there is a probability measure (corresponding to $p_t \in [0,1]$) such that the sequence having the defensive property remains less than one for any outcome. In other words, at each step there is a prediction which defends the sequence from exceeding one. The proof presented here is constructive: p_t can be found as a root of a continuous function in the interval $[0,1]$. If the dimension of the outcomes is more than 2, the proof is based on the fixed point theorem, and the procedure to find p is often very inefficient. We will show later that if the set of outcomes Ω_t is the full interval $[Y_{t,1}, Y_{t,2}]$, the same p can sometimes be used as for the case $\Omega_t = \{Y_{t,1}, Y_{t,2}\}$.

We state here the Defensive Forecasting algorithm for $\Omega = \{Y_{t,1}, Y_{t,2}\}$,

$\Gamma = \mathbb{R}$ as Algorithm 2. The sequence of functions f_t is defined by

$$f_t(\omega, p) = Q_t(\omega_1, \gamma_1, \dots, \omega_{t-1}, \gamma_{t-1}, \omega, \gamma^p) - 1, \tag{2.12}$$

where $\omega \in \Omega_t$, $p \in [0,1]$, and $\omega_1 \in \Omega_1, \dots, \omega_{t-1} \in \Omega_{t-1}$, $\gamma_1 \in \Gamma_1, \dots, \gamma_{t-1} \in \Gamma_{t-1}$.

Algorithm 2 Defensive Forecasting algorithm

Require: An initial weights distribution $P_0(d\theta) = P_0^*(d\theta)$ on the experts.

for $t = 1, 2, \dots$ **do**

Read Ω_t, Γ_t.

Read experts' loss functions λ_t^θ, $\theta \in \Theta$.

Calculate the appropriate learning rates $\eta_t^\theta > 0$, $\theta \in \Theta$.

Read experts' predictions ξ_t^θ, $\theta \in \Theta$.

Define $f_t : \Omega_t \times [0,1] \to (-\infty, \infty]$ by (2.12) using the correspondence (2.9).

if $f_t(Y_{t,2}, 0) \le 0$ **then**

Predict $\gamma_t = \gamma^0$.

else if $f_t(Y_{t,1}, 1) \le 0$ **then**

Predict $\gamma_t = \gamma^1$.

else

Predict $\gamma_t = \gamma^p$ satisfying $f_t(Y_{t,1}, p) = f_t(Y_{t,2}, p)$.

end if

Read ω_t.

end for

The calculation of the appropriate learning rates η_t^θ depends on the outcome set and prediction set at the step t and on the loss function chosen by the expert θ at this step. In the next section we will describe ways to approach this calculation for two important cases. They closely relate to the proofs of mixability (2.4) for the Aggregating Algorithm.

Assume that all the experts choose the same loss functions λ_t on all the steps. Then it is interesting to note that Algorithm 2 in some cases finds the prediction satisfying the equation (2.7), if trivial predictions $Y_{T,1}$ and $Y_{T,2}$ are

not suitable. Indeed, it solves

$$\int_\Theta \prod_{t=1}^{T-1} e^{\eta_t\left(\prod_{i=t}^{T-1}\alpha_i\right)\left(\lambda_t(\omega_t,\gamma_t)-\lambda_t(\omega_t,\xi_t^\theta)\right)} e^{\eta_T\left(\lambda_T(Y_{T,2},\gamma)-\lambda_T(Y_{T,2},\xi_T^\theta)\right)} P_0(d\theta)$$
$$- \int_\Theta \prod_{t=1}^{T-1} e^{\eta_t\left(\prod_{i=t}^{T-1}\alpha_i\right)\left(\lambda_t(\omega_t,\gamma_t)-\lambda_t(\omega_t,\xi_t^\theta)\right)} e^{\eta_T\left(\lambda_T(Y_{T,1},\gamma)-\lambda_T(Y_{T,1},\xi_T^\theta)\right)} P_0(d\theta) = 0$$

in $\gamma \in [Y_{T,1}, Y_{T,2}]$. We define

$$g_T(\omega) = -\frac{1}{\eta_T} \ln \int_\Theta e^{-\eta_T \lambda_T(\omega,\xi_T^\theta)} \prod_{t=1}^{T-1} e^{-\eta_t\left(\prod_{i=t}^{T-1}\alpha_i\right)\lambda_t(\omega_t,\xi_t^\theta)} P_0(d\theta) \tag{2.13}$$

for any $\omega \in \Omega_T$. Rewriting the equation for the root, we have

$$e^{\eta_T\left(\lambda_T(Y_{T,2},\gamma)-g_T(Y_{T,2})\right)} - e^{\eta_T\left(\lambda_T(Y_{T,1},\gamma)-g_T(Y_{T,1})\right)} = 0$$

Moving the second exponent to the right-hand side and taking logarithms of both sides, we obtain

$$\lambda_T(Y_{T,2},\gamma) - g_T(Y_{T,2}) = \lambda_T(Y_{T,1},\gamma) - g_T(Y_{T,1}) \le 0, \tag{2.14}$$

(the inequality follows from the defensive property), which satisfies (2.7) if the loss function, the outcome set, and the prediction set are the same for all steps and there is no discounting (for all t, $Y_{t,1} = Y_1$, $Y_{t,2} = Y_2$, $\lambda_t = \lambda$ and thus $\eta_t = \eta$, and $\alpha_t = 1$). Therefore the prediction of the DF in this case coincides with the prediction of the AA clipped to the prediction region $[Y_1, Y_2]$.

Standard proofs of upper bounds for Defensive Forecasting are based on the following argument. It states that Q_t is a forecast-continuous sequence having the defensive property.

Lemma 2.3 *Assume that the sequence of functions Q_t^θ is forecast-continuous and has the defensive property. Then the mixtures Q_t as functions of two variables ω, γ at the step t (given $\gamma_1, \dots, \gamma_{t-1}$) form a forecast-continuous sequence having the defensive property.*

PROOF The continuity easily follows from the continuity of Q_t^θ and the integration functional. We proceed by induction in T. For $T = 0$ we have $\mathrm{E}_\pi Q_0 = \mathrm{E}_\pi 1 \leq 1$. For $T > 0$ assume that for any $\omega_1 \in \Omega_1, \ldots, \omega_{T-2} \in \Omega_{T-2}$ and any $\gamma_1 \in \Gamma_1, \ldots, \gamma_{T-2} \in \Gamma_{T-2}$

$$\mathrm{E}_\pi Q_{T-1}(\omega_1, \gamma_1, \ldots, \omega_{T-2}, \gamma_{T-2}, \omega, \gamma^{p^\pi}) \leq 1$$

for any $\pi \in \mathcal{P}(\Omega_{T-1})$. Then by Lemma 2.2 there exists $\pi_{T-1} \in \mathcal{P}(\Omega_{T-1})$ such that

$$Q_{T-1}(\omega_1, \gamma_1, \ldots, \omega_{T-2}, \gamma_{T-2}, \omega, \gamma^{p^{\pi_{T-1}}}) = \int_\Theta \left(\prod_{t=1}^{T-2} \left(Q_t^\theta\right)^{\prod_{i=t}^{T-2} \alpha_i} \right) Q_{T-1}^\theta P_0(d\theta) \leq 1 \tag{2.15}$$

for any $\omega \in \Omega_{T-1}$. We denote $\gamma_{T-1} := \gamma^{p^{\pi_{T-1}}}$ and fix any $\omega_{T-1} \in \Omega_{T-1}$. We obtain

$$\begin{aligned}
&\mathrm{E}_\pi Q_T(\omega_1, \gamma_1, \ldots, \omega_{T-1}, \gamma_{T-1}, \omega, \gamma^{p^\pi}) \\
&= \mathrm{E}_\pi \int_\Theta \left(\prod_{t=1}^{T-1} \left(Q_t^\theta(\omega_t, \gamma_t)\right)^{\prod_{i=t}^{T-1} \alpha_i} \right) Q_T^\theta(\omega, \gamma^{p^\pi}) P_0(d\theta) \\
&= \int_\Theta \left(\prod_{t=1}^{T-1} \left(Q_t^\theta(\omega_t, \gamma_t)\right)^{\prod_{i=t}^{T-1} \alpha_i} \right) \left(\mathrm{E}_\pi Q_T^\theta(\omega, \gamma^{p^\pi})\right) P_0(d\theta) \\
&\leq \int_\Theta \prod_{t=1}^{T-1} \left(Q_t^\theta(\omega_t, \gamma_t)\right)^{\prod_{i=t}^{T-1} \alpha_i} P_0(d\theta) \\
&= \int_\Theta \left(\left(\prod_{t=1}^{T-2} \left(Q_t^\theta\right)^{\prod_{i=t}^{T-2} \alpha_i} \right) Q_{T-1}^\theta \right)^{\alpha_{T-1}} P_0(d\theta) \\
&\leq \left(\int_\Theta \left(\prod_{t=1}^{T-2} \left(Q_t^\theta\right)^{\prod_{i=t}^{T-2} \alpha_i} \right) Q_{T-1}^\theta P_0(d\theta) \right)^{\alpha_{T-1}} \leq 1.
\end{aligned}$$

The first inequality holds because $\mathrm{E}_\pi Q_T^\theta(\omega, \gamma^{p^\pi}) \leq 1$ for any $\pi \in \mathcal{P}(\Omega_T)$. The penultimate inequality holds due to the concavity of the function x^α with $x > 0$, $\alpha \in [0, 1]$. The last inequality holds due to (2.15). This completes the proof. ∎

By Lemma 2.2 at each step t there exists a prediction γ_t such that Q_t is

less than one. Thus

$$\int_\Theta \prod_{t=1}^{T-1} e^{\eta_t^\theta \left(\prod_{i=t}^{T-1}\alpha_i\right)\left(\lambda_t^\theta(\omega_t,\gamma_t)-\lambda_t^\theta(\omega_t,\xi_t^\theta)\right)} e^{\eta_T^\theta\left(\lambda_t^\theta(\omega_T,\gamma_T)-\lambda_t^\theta(\omega_T,\xi_T^\theta)\right)} P_0(d\theta) \le 1. \quad (2.16)$$

Extracting the losses of the learner and the losses of the experts from this expression where possible, one can obtain upper bounds in the more familiar form (2.1).

Defensive Forecasting for multiple loss functions

A major problem in prediction with expert advice is the choice of the loss function used to measure the quality of the predictions. In the setting of multiple loss functions the performance of the learner and experts can be measured by many loss functions λ^m, $m = 1, \dots, M$. It is required that the learner is competitive under all the loss functions.

Assume that the number of experts is finite and equal to K: $|\Theta| = K$. We shall reduce our problem to the framework described previously. It is equivalent to say that the number of experts is larger and equal to $K \times M$ (they are then indexed by (θ, m)). All the experts with $m = 1, \dots, M$ and fixed θ give the same predictions ξ_t^θ. We take for all t the outcome set $\Omega = \{Y_1, Y_2\}$, the prediction set $\Gamma = [Y_1, Y_2]$, and the correspondence between $[0, 1]$ and Γ is defined by (2.9). We do not take discounting into account in this subsection, $\alpha_t = 1$ for all t.

By $L_T^m := \sum_{t=1}^T \lambda^m(\omega_t, \gamma_t)$ we denote the cumulative loss of the learner under the loss function λ^m at the step T, and by $L_T^{\theta,m} := \sum_{t=1}^T \lambda^m(\omega_t, \xi_t^\theta)$ we denote the cumulative loss of the expert (θ, m) under this loss function. The following performance guarantee for the loss of the learner follows from (2.16) (first proven in Chernov and Vovk, 2009).

Theorem 2.1 *Assume that Q_t^θ defined by (2.8), $t = 1, \dots,$ is a forecast-continuous sequence having the defensive property. Using Algorithm 2 as the learner's strategy for the game with multiple loss functions guarantees that, for all $T = 1, 2, \dots,$*

$$L_T^m \le L_T^{\theta,m} + \frac{\ln KM}{\eta^m}. \quad (2.17)$$

for any $\theta = 1, \dots, K$ *and any* $m = 1, \dots, M$.

PROOF We have from (2.16) for the uniform initial distribution over the new experts:

$$\frac{1}{KM} \sum_{m=1}^{M} \sum_{\theta=1}^{K} \prod_{t=1}^{T} e^{\eta^m \left(\lambda^m(\omega_t, \gamma_t) - \lambda^m(\omega_t, \xi_t^\theta) \right)} \le 1,$$

and thus

$$\prod_{t=1}^{T} e^{\eta^m \left(\lambda^m(\omega_t, \gamma_t) - \lambda^m(\omega_t, \xi_t^\theta) \right)} \le KM$$

for any $\theta = 1, \dots, K$ and any $m = 1, \dots, M$. Taking natural logarithms of both parts completes the proof. ∎

It is well known that the constant η^m in (2.17) is optimal in the case $M = 1$ (Watkins's theorem; see Appendix A of Vovk and Zhdanov, 2009).

Defensive Forecasting for discounted loss

Discounted loss (especially exponential smoothing) is another setting widely used in many practical applications of online prediction, such as finance, online tracking, and others (see Gardner, 2006, for the review of exponential smoothing).

We take for all t the outcome set $\Omega = \{Y_1, Y_2\}$, the prediction set $\Gamma = [Y_1, Y_2]$, and the same loss function λ for all the experts at all the steps. The correspondence between $[0, 1]$ and Γ is defined by (2.9).

Theorem 2.2 *Assume that* Q_t^θ *defined by* (2.8), $t = 1, \dots,$ *is a forecast-continuous sequence having the defensive property. Using Algorithm 2 as the learner's strategy for the game with discounted loss guarantees that, for all* $T = 1, 2, \dots,$

$$\mathcal{L}_T \le \mathcal{L}_T^\theta + \frac{\ln K}{\eta}. \tag{2.18}$$

for any $\theta = 1, \dots, K$.

PROOF We have from (2.16) for the uniform initial distribution over the experts:

$$\frac{1}{K}\prod_{t=1}^{T} e^{\eta\left(\prod_{i=t}^{T-1}\alpha_i\right)\left(\lambda(\omega_t,\gamma_t)-\lambda(\omega_t,\xi_t^{\theta})\right)} \leq 1$$

and thus

$$e^{\eta\left(\mathcal{L}_T-\mathcal{L}_T^{\theta}\right)} \leq K$$

for any $\theta = 1, \ldots, K$. Taking natural logarithms of both parts completes the proof. ■

The discounted linear regression is one of the most popular methods where discounting is used. We will describe the online setting in Chapter 3.

2.3 Simple cases

In this section we discuss two important mixable games: the square loss game and the logarithmic loss game. We show how the Aggregating Algorithm and Defensive Forecasting can be applied to them. Unfortunately, another popular loss function, absolute loss function, is not mixable (see Vovk, 1998). Other methods should be applied to compete with experts under this loss function, such as the weak Aggregating Algorithm (Kalnishkan and Vyugin, 2005), if we wish to have the coefficient 1 in front of the cumulative loss of the experts in (2.1). Throughout this section the number of experts is finite and equal to K: $|\Theta| = K$. For the logarithmic loss function we take the outcome set $\Omega = \{0, 1\}$ and the prediction set $\Gamma = [0, 1]$. For the square loss function we take the outcome set $\Omega = \{Y_1, Y_2\}$ and the prediction set $\Gamma = \mathbb{R}$. The facts which are necessary to handle the more general case of $\Omega = [Y_1, Y_2]$ will be proven in Chapter 3. We also take no discounting in Defensive Forecasting, $\alpha_t = 1$ for all t.

2.3.1 Logarithmic loss function

The prediction set $\Gamma = [0, 1]$ can be identified with the set of all probability distributions $\mathcal{P}(\Omega)$ over the outcome set $\Omega = \{0, 1\}$. Each prediction $\gamma \in \Gamma$ is identified with the pair $(\gamma(1), \gamma(0)) = (\gamma, 1 - \gamma)$ such that $\gamma(1) = \gamma$ is the predicted probability of the outcome 1, $\gamma(0) = 1-\gamma$ is the predicted probability of the outcome 0. This is a particular case of the identification (2.9).

The *logarithmic loss* function (log-loss function) is defined as

$$\lambda(\omega, \gamma) := \begin{cases} -\ln \gamma & \text{if } \omega = 1, \\ -\ln(1 - \gamma) & \text{if } \omega = 0, \end{cases} \tag{2.19}$$

where $\omega \in \Omega$ and $\gamma \in \Gamma$. Logarithmic loss is sometimes written as

$$\lambda(\omega, \gamma) = -\ln \gamma(\omega) \tag{2.20}$$

meaning that the loss is the minus logarithm of the predicted probability of

the realized outcome. This form allows us to generalize the logarithmic loss function on the cases of more than two possible outcomes.

Aggregating Algorithm for the logarithmic loss function

As we will see soon, the Aggregating Algorithm for the logarithmic loss function and $\eta = 1$ is the same as the Bayesian scheme, which goes back to DeSantis et al. (1988) in the case of countable Θ and Ω. We call it the Bayesian Algorithm (BA) as it is virtually identical to the Bayes rule used in Bayesian learning (the main difference being that the experts are not required to follow any prediction strategies). We take $\eta = 1$.

The weights update (2.2) becomes

$$P_t(d\theta) = \xi_t^\theta(\omega_t) P_{t-1}(d\theta) = P_0(d\theta) \prod_{i=1}^{t} \xi_i^\theta(\omega_i), \tag{2.21}$$

where $\xi_i^\theta(\omega)$ is understood in the sense of (2.20). Therefore, the normalized version $P_t^*(d\theta)$ becomes identical to the posterior distribution on Θ after observing $\omega_1, \dots, \omega_t$. The generalized prediction (2.3) becomes

$$g_t(\omega) = -\ln \int_\Theta \xi_t^\theta(\omega) P_{t-1}^*(d\theta),$$

and thus represents the loss of the Bayesian mixture. Therefore, the generalized prediction corresponds to the permitted prediction: the substitution function $\Sigma : \mathbb{R}^\Omega \to \Gamma$ is simply $e^{-(\cdot)}$. It is now clear that the log-loss game is mixable for $\eta \le 1$. Indeed, the function x^η is concave for $\eta < 1$, and thus

$$\int_\Theta (\xi_t^\theta(\omega))^\eta Q(d\theta) \le \left(\int_\Theta \xi_t^\theta(\omega) Q(d\theta) \right)^\eta$$

for any ω and $Q \in \mathcal{P}(\Theta)$. Taking minus logarithms and multiplying by $\frac{1}{\eta}$, we obtain

$$-\frac{1}{\eta} \ln \left(\int_\Theta \xi_t^\theta(\omega) Q(d\theta) \right)^\eta \le -\frac{1}{\eta} \ln \int_\Theta (\xi_t^\theta(\omega))^\eta Q(d\theta).$$

In other words, the loss of the prediction corresponding to $\eta = 1$ is less than

the generalized prediction calculated with any other $\eta < 1$.

The Bayesian Algorithm works as follows.

Algorithm 3 Bayesian Algorithm

Require: An initial weights distribution $P_0(d\theta) = P_0^*(d\theta)$ on Θ.

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

Read experts' predictions $\xi_t^\theta, \theta \in \Theta$.

Predict $\gamma_t = \int_\Theta \xi_t^\theta P_{t-1}^*(d\theta)$.

Read $\omega_t \in \Omega$.

Update the weights $P_t(d\theta) = \xi_t^\theta(\omega_t) P_{t-1}(d\theta)$.

Normalize the weights $P_t^*(d\theta) = P_t(d\theta) / \int_\Theta P_t(d\theta)$.

end for

The following theorem can be easily deduced from (2.6) for the uniform initial weights distribution $P_0(d\theta) = 1/K$.

Theorem 2.3 *Using Algorithm 3 as the learner's strategy in Protocol 1 for the game with logarithmic loss function guarantees that, for all* $T = 1, 2, \ldots$,

$$L_T \le \min_{\theta=1,\ldots,K} L_T^\theta + \ln K. \tag{2.22}$$

It is also interesting to note that (2.6) leads to the following exact equality for the loss of the BA in comparison with the loss of any expert in the case of countable number of experts. It holds because $L_T(\mathrm{AA}) = L_T(\mathrm{APA})$ for this loss function:

$$L_T(\mathrm{BA}) = L_T^\theta + \ln(1/P_0^\theta) - \ln(1/P_T^\theta)$$

for any $\theta \in \Theta$.

Defensive Forecasting for the logarithmic loss function

Function Q_t^θ from (2.8) is expressed as follows:

$$Q_t^\theta = e^{\eta\left(-\ln \gamma_t(\omega_t) + \ln \xi_t^\theta(\omega_t)\right)} = \left(\frac{\xi_t^\theta(\omega_t)}{\gamma_t(\omega_t)}\right)^\eta.$$

Lemma 2.4 *For $\eta \in (0,1]$, Q_t^θ is a forecast-continuous sequence having the defensive property.*

PROOF The continuity is obvious. It suffices to check that for all $p, q \in [0,1]$

$$pe^{\eta(-\ln p + \ln q)} + (1-p)e^{\eta(-\ln(1-p)+\ln(1-q))} \le 1,$$

where $p = \gamma_t(1)$, $1-p = \gamma_t(0)$, $q = \xi_t^\theta(1)$, $1-q = \xi_t^\theta(0)$. In other words,

$$p^{1-\eta} q^\eta + (1-p)^{1-\eta}(1-q)^\eta \le 1$$

for all $p, q \in [0,1]$. The last inequality immediately follows from the generalized inequality between arithmetic and geometric means: $u^\alpha v^{1-\alpha} \le \alpha u + (1-\alpha)v$ for any $u, v \ge 0$ and $\alpha \in [0,1]$, which in turn (after taking logarithm of both parts) follows from the fact that the logarithm function is concave. ∎

Let us assign equal weights $1/K$ to all the experts in the beginning of the prediction process. Then by Lemma 2.3 the linear combination

$$Q_T = \frac{1}{K} \sum_{\theta=1}^{K} \prod_{t=1}^{T} Q_t^\theta$$

is also a forecast-continuous sequence having the defensive property. Consequently, by Lemma 2.2 there exist predictions such that $Q_T \le 1$ for any T. Let us denote the cumulative loss of the learner following the Defensive Forecasting algorithm by $L_T = \sum_{t=1}^{T} \lambda(\omega_t, \gamma_t)$, and the cumulative loss of the expert θ by $L_T^\theta = \sum_{t=1}^{T} \lambda(\omega_t, \xi_t^\theta)$. Then the following theorem holds.

Theorem 2.4 *Using Algorithm 2 as the learner's strategy in Protocol 1 for the game with logarithmic loss function guarantees that, for all $T = 1, 2, \ldots$,*

$$L_T \le \min_{\theta=1,\ldots,K} L_T^\theta + \ln K. \tag{2.23}$$

PROOF $Q_T \leq 1$ ensures that for $\eta = 1$

$$\frac{1}{K} e^{L_T - L_T^\theta} = \frac{1}{K} \prod_{t=1}^{T} e^{\lambda(\omega_t, \gamma_t) - \lambda(\omega_t, \xi_t^\theta)} \leq \frac{1}{K} \sum_{\tilde{\theta}=1}^{K} \prod_{t=1}^{T} e^{\lambda(\omega_t, \gamma_t) - \lambda(\omega_t, \xi_t^{\tilde{\theta}})} \leq 1$$

for $\theta = \arg\min_{\tilde{\theta}=1,\dots,K} L_T^{\tilde{\theta}}$. Taking natural logarithms of the left and right hand sides completes the proof. ∎

2.3.2 Square loss function

Square loss, or square error, is a very popular measure of the quality of predictions in statistics. The *square loss* function is defined as

$$\lambda(\omega, \gamma) := (\gamma - \omega)^2, \tag{2.24}$$

where $\omega \in \Omega$ and $\gamma \in \Gamma$. Square loss for the game with $\Omega = \{0, 1\}$ is sometimes defined differently by $\lambda(\omega, \gamma) = (\gamma - \omega)^2 + ((1 - \gamma) - (1 - \omega))^2$, which is twice the square loss by the first definition. To distinguish them, we will sometimes call the loss from the last definition the *Brier loss* (following Brier, 1950). This form allows us to generalize the square loss function for the cases of multidimensional outcomes.

Aggregating Algorithm for the square loss function

We first prove that the square loss game with $\Omega = \{Y_1, Y_2\}$ and $\Gamma = \mathbb{R}$ is mixable. In other words, we prove that there exist values of η such that for any generalized prediction $g_t(\omega)$ the inequality (2.4) holds for some prediction γ_t.

Lemma 2.5 *The square loss function is η-mixable if and only if $\eta \leq \frac{2}{(Y_2 - Y_1)^2}$.*

PROOF Let us represent the generalized prediction (2.3) by the point

$$(x, y) = (e^{-\eta g(Y_1)}, e^{-\eta g(Y_2)})$$

of the $\mathbb{R}^2$ plane. The set of permitted predictions is represented by the para-

metric curve

$$(\tilde{u}, \tilde{v}) = (e^{-\eta(\gamma - Y_1)^2}, e^{-\eta(\gamma - Y_2)^2})$$

for all $\gamma \in \Gamma$. Reformulating (2.4), we can say that the loss function is η-mixable if and only if for any convex mixture (x, y) of the points of the curve $(\tilde{u}, \tilde{v})$ we can find a point γ_0 on this curve which lies to the North-East of the mixture:

$$e^{-\eta(\gamma_0 - \omega)^2} \geq e^{-\eta g(\omega)}, \quad \forall \omega \in \Omega.$$

Let us denote the set of points which correspond to $\gamma \in [Y_1, Y_2]$ of the curve $(\tilde{u}, \tilde{v})$ by (u, v). For each u on the curve we have unique v corresponding to it (note that this does not hold for $(\tilde{u}, \tilde{v})$). Therefore, it is enough to prove that the curve (u, v) is concave. We check that second derivative of v in u is less than or equal to zero:

$$\begin{aligned}
\frac{d^2v}{du^2} = \frac{\frac{d^2v}{d\gamma du}}{\frac{du}{d\gamma}} = \frac{d\left(\frac{dv/d\gamma}{du/d\gamma}\right)/d\gamma}{\frac{du}{d\gamma}} &= \frac{d\left(\frac{2\eta(Y_2-\gamma)e^{-\eta(Y_2-\gamma)^2}}{2\eta(Y_1-\gamma)e^{-\eta(Y_1-\gamma)^2}}\right)/d\gamma}{2\eta(Y_1-\gamma)e^{-\eta(Y_1-\gamma)^2}} \\
&= \frac{d\left(\frac{Y_2-\gamma}{Y_1-\gamma}e^{-\eta(Y_2^2-Y_1^2)+2\eta\gamma(Y_2-Y_1)}\right)/d\gamma}{2\eta(Y_1-\gamma)e^{-\eta(Y_1-\gamma)^2}} \\
&= \frac{e^{-\eta(Y_2^2-Y_1^2)+2\eta\gamma(Y_2-Y_1)}\left(\frac{Y_2-Y_1}{(Y_1-\gamma)^2} + 2\eta(Y_2-Y_1)\frac{Y_2-\gamma}{Y_1-\gamma}\right)}{2\eta(Y_1-\gamma)e^{-\eta(Y_1-\gamma)^2}} \leq 0.
\end{aligned}$$

Thus we need $\frac{1}{Y_1-\gamma} + 2\eta(Y_2 - \gamma) \leq 0$, which is equivalent to

$$\eta \leq \frac{1}{2(Y_2-\gamma)(\gamma-Y_1)}.$$

Since $\max_{\gamma\in[Y_1,Y_2]}(Y_2-\gamma)(\gamma-Y_1) = \frac{1}{4}(Y_2-Y_1)^2$, the curve is concave for any γ if and only if $\eta \leq \frac{2}{(Y_2-Y_1)^2}$.

For any point on the curve $(\tilde{u}, \tilde{v})$ corresponding to $\tilde{\gamma} \in \mathbb{R} \setminus [Y_1, Y_2]$ at least one of the end points of the curve (u, v) corresponding to $\gamma \in \{Y_1, Y_2\}$ lies to

the North-East:

$$e^{-\eta(\gamma-\omega)^2} \geq e^{-\eta(\tilde{\gamma}-\omega)^2}, \quad \forall \omega \in \Omega.$$

Taking these points in any convex mixture $(\tilde{x}, \tilde{y})$ instead of the points on $(\tilde{u}, \tilde{v})$, we obtain that $(\tilde{x}, \tilde{y})$ lies to the North-East of the mixture (x, y). We can always find a prediction γ_0 lying to the North-East of $(\tilde{x}, \tilde{y})$ by the argument for (u, v). ■

This mixability property leads to the following upper bound on the loss of the Aggregating Algorithm with the uniform initial weights distribution (see (2.6)).

Theorem 2.5 *Using Algorithm 1 as the learner's strategy in Protocol 1 for the game with square loss function guarantees that, for all* $T = 1, 2, \ldots$,

$$L_T \leq \min_{\theta=1,\ldots,K} L_T^\theta + (Y_2 - Y_1)^2 \frac{\ln K}{2}. \tag{2.25}$$

We now describe a formula for the substitution function. Since the loss of any permitted prediction is also a generalized prediction for some weights, it is easy to see from (2.7) that for $\gamma = \Sigma(g)$ we have for the square loss game

$$(\gamma - Y_1)^2 - g(Y_1) = (\gamma - Y_2)^2 - g(Y_2).$$

Therefore, the substitution function is expressed as follows:

$$\gamma = \frac{Y_2 + Y_1}{2} - \frac{g(Y_2) - g(Y_1)}{2(Y_2 - Y_1)}. \tag{2.26}$$

Finally, we visualize in Figure 2.1 how the Aggregating Algorithm makes predictions in the game with two experts. Clearly, the curve for the losses is strictly convex because the averaging is performed after taking the exponents.

Defensive Forecasting for the square loss function

We start by proving that Q_t^θ has the defensive property.

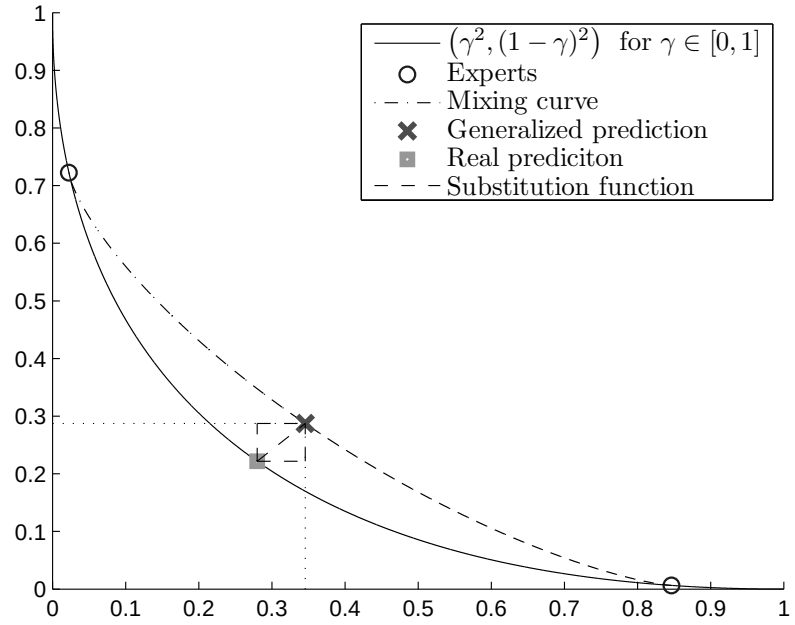


Figure 2.1: Aggregating Algorithm for two experts and $Y_1 = 0, Y_2 = 1$. Horizontal axis corresponds to the case $\omega = 0$, vertical axis corresponds to the case $\omega = 1$. The cross corresponds to a mixture of two experts. The substitution function finds the prediction point on the loss curve which is the opposite vertex of a square drawn from the generalised prediction. This square is drawn with dashed lines and the corresponding point is given by (2.26). The area to the bottom-left of the dotted lines is the area where a permitted prediction can lie in order for the upper bound (2.25) to hold.

Lemma 2.6 *For* $\eta \in \left(0, \frac{2}{(Y_2-Y_1)^2}\right]$,

$$Q_t^\theta = e^{\eta\left((\gamma_t-\omega_t)^2-(\xi_t^\theta-\omega_t)^2\right)}$$

is a forecast-continuous sequence having the defensive property.

Proof The continuity is obvious. We need to prove that

$$pe^{\eta\left((\gamma-Y_2)^2-(\xi_t^\theta-Y_2)^2\right)} + (1-p)e^{\eta\left((\gamma-Y_1)^2-(\xi_t^\theta-Y_1)^2\right)} \le 1 \tag{2.27}$$

holds for all $\gamma \in [Y_1, Y_2]$ and $\eta \in \left(0, \frac{2}{(Y_2-Y_1)^2}\right]$. Indeed, for any $\gamma \in \mathbb{R} \setminus [Y_1, Y_2]$ there exists $\tilde{\gamma} \in \{Y_1, Y_2\}$ such that $(\tilde{\gamma}-\omega)^2 \le (\gamma-\omega)^2$ for any $\omega \in \Omega$. Since

the exponent function is increasing, the inequality (2.27) for any $\gamma \in \mathbb{R}$ will follow.

We use the replacements $\gamma = p(Y_2 - Y_1) + Y_1$ for some $p \in [0,1]$, $\xi_t^\theta = q(Y_2 - Y_1) + Y_1$ for some $q \in \mathbb{R}$, and $\mu = \eta(Y_2 - Y_1)^2$. Then we have to show that for all $p \in [0,1]$, $q \in \mathbb{R}$, and $\eta \in \left(0, \frac{2}{(Y_2-Y_1)^2}\right]$,

$$pe^{\mu\left((1-p)^2-(1-q)^2\right)} + (1-p)e^{\mu\left(p^2-q^2\right)} \le 1.$$

If we substitute $q = p + x$, the last inequality will reduce to

$$pe^{2\mu(1-p)x} + (1-p)e^{-2\mu px} \le e^{\mu x^2}, \quad \forall x \in \mathbb{R}.$$

Applying Hoeffding's inequality (see Hoeffding, 1963, 4.16) to the random variable X that is equal to 1 with probability p and to 0 with probability $(1-p)$, we obtain

$$pe^{h(1-p)} + (1-p)e^{-hp} \le e^{h^2/8}$$

for any $h \in \mathbb{R}$. With the substitution $h := 2\mu x$ it reduces to

$$pe^{2\mu(1-p)x} + (1-p)e^{-2\mu px} \le e^{\mu^2 x^2/2} \le e^{\mu x^2},$$

where the last inequality holds if $\mu \le 2$. It is equivalent to $\eta \le \frac{2}{(Y_2-Y_1)^2}$. ∎

We assign equal weights $1/K$ to all the experts in the beginning of the prediction process. Derivations similar to the ones for the logarithmic loss function lead to the following upper bound on the cumulative square loss of the learner which uses the Defensive Forecasting algorithm.

Theorem 2.6 *Using Algorithm 2 as the learner's strategy in Protocol 1 for the game with square loss function guarantees that, for all $T = 1, 2, \ldots$,*

$$L_T \le \min_{\theta=1,\ldots,K} L_T^\theta + (Y_2 - Y_1)^2 \frac{\ln K}{2} \ . \tag{2.28}$$

2.4 Brier game

There are several important loss functions which have been shown to be mixable and for which the optimal regret term has been found. In Section 2.3 we described two examples for the case of two possible outcomes: the log-loss function and the square loss function.

In this section we concentrate on a generalization of the square loss function to the case of multiple possible outcomes. We consider the classification problem: outcomes take values in a finite set. Surprisingly, the problem of non-binary classification under the square loss has never been analyzed in the framework of prediction with expert advice before the conference version of these results has been published (Vovk and Zhdanov, 2008). The full version (Vovk and Zhdanov, 2009) appears in the Journal of Machine Learning Research.

Kivinen and Warmuth (1999) consider competing with finite number of experts under the square loss when the outcomes and predictions lie in a zero-centred ball of the Euclidean space and under the logarithmic loss when the outcomes and predictions lie in the simplex of $\mathbb{R}^d$. They notice an interesting property of the upper bound on the Brier loss of the algorithms which deal with multidimensional predictions. The algorithms which are intended to give multidimensional predictions directly (first type) may have a better regret term than the algorithms which predict each component separately (second type). On the other hand, the algorithms of the second type are capable to compete with the strategies which choose the best expert for each component separately: this choice has an advantage when, for example, one of the experts predicts well one of the components of the outcome and another expert predicts well another component. It is not clear whether the algorithms of the first type have this property.

Our setting is a special case of the setting of Freund and Schapire (1997), where the predictions are not made and only experts' losses are known. In that setting the upper bounds are worse since the assumptions are weaker. In online convex optimization (Zinkevich, 2003), one allows the loss functions to be unknown before the prediction is made, but requires the learner's prediction

to be the weighted average of the experts' predictions. Our way of giving predictions is different, and we obtain better bounds under the specific square loss.

In this section we apply the Aggregating Algorithm. It is possible to apply defensive forecasting, but it does not give any advantages for the Brier game which we consider. Instead, the calculation of the predictions may become very complicated if the number of classes is large.

2.4.1 Prediction algorithm and performance guarantee

In this section we are interested in the following *Brier game* (Brier, 1950): Ω is a finite and non-empty set with d elements, where each element $\omega \in \Omega$ is associated with a unit basis vector from $\mathbb{R}^d$; $\Gamma := \mathcal{P}(\Omega)$ is the set of all probability measures on Ω; and

$$\lambda(\omega, \gamma) = \sum_{i=1}^{d} \left(\gamma^i - \omega^i\right)^2, \tag{2.29}$$

where $\gamma = (\gamma^1, \dots, \gamma^d) \in \Gamma$, $\omega = (\omega^1, \dots, \omega^d) \in \Omega$. For example, if $\Omega = \{1, 2, 3\}$, $\omega = (1, 0, 0)$, $\gamma^1 = 1/2$, $\gamma^2 = 1/4$, and $\gamma^3 = 1/4$, then $\lambda(\omega, \gamma) = (1/2 - 1)^2 + (1/4 - 0)^2 + (1/4 - 0)^2 = 3/8$.

The game follows Protocol 1. The set Θ indexing experts is a finite set of K elements. We denote the cumulative loss of the learner at the step t by L_t and the cumulative loss of the expert θ by L_t^θ.

An optimal (in the sense of Theorem 2.7 below) strategy for the learner for prediction with expert advice in the Brier game is given by Algorithm 4. We use the notation $\omega\{i\}$ for the ith possible outcome in Ω (for the vector $(0, \dots, 0, 1, 0, \dots, 0)$ of the length d with 1 at the ith position) and

$$t^+ := \max(t, 0).$$

The algorithm will be derived in Section 2.4.2.

Algorithm 4 Aggregating Algorithm for the Brier game

$P_0^\theta = 1/K$, $\theta \in \Theta$.
for $t = 1, 2, \ldots$ **do**
  Read experts' predictions ξ_t^θ, $\theta \in \Theta$.
  Set $G_t(\omega) := -\ln \sum_{\theta=1}^K P_{t-1}^\theta e^{-\lambda(\omega, \xi_t^\theta)}$, $\omega \in \Omega$.
  Solve $\sum_{\omega \in \Omega} (s - G_t(\omega))^+ = 2$ in $s \in \mathbb{R}$.
  Set $\gamma_t^i = (s - G_t(\omega\{i\}))^+/2$, $i = 1, \ldots, d$.
  Output prediction γ_t.
  Read observation ω_t.
  Update weights $P_t^\theta = P_{t-1}^\theta e^{-\lambda(\omega_t, \xi_t^\theta)}$.
end for

The following theorem gives a performance guarantee for Algorithm 4, which cannot be improved by any other prediction algorithm. For the full proof, see Vovk and Zhdanov (2008). The main part in the proof of this result is the proof of the mixability (see (2.4)) of the Brier loss function.

Lemma 2.7 *The Brier loss function is η-mixable if and only if $\eta \in (0, 1]$.*

It is then easy to prove the following theorem.

Theorem 2.7 *Using Algorithm 4 as the learner's strategy in Protocol 1 for the Brier game guarantees that, for all $T = 1, 2, \ldots$,*

$$L_T \le \min_{\theta=1,\ldots,K} L_T^\theta + \ln K. \tag{2.30}$$

If $A < \ln K$, the learner does not have a strategy guaranteeing, for all $T = 1, 2, \ldots$,

$$L_T \le \min_{\theta=1,\ldots,K} L_T^\theta + A. \tag{2.31}$$

2.4.2 Derivation of the algorithm

To achieve the loss bound (2.30) in Theorem 2.7, the learner uses the Aggregating Algorithm with $\eta = 1$. In this section, a suitable substitution function for this algorithm is described. This substitution function does not require

that the weights of the experts are normalized in computation of the generalized prediction. In other words, we try to find a substitution function satisfying (2.7).

Suppose that we are given a generalized prediction $g = (l_1, \dots, l_d)$ computed from a normalized distribution on the experts. We are required to find a permitted prediction $(u_1, u_2, \dots, u_d)$, $u_i \geq 0, \forall i$, such that all the possible losses

$$\begin{array}{rcl} \lambda_1 & = & (u_1 - 1)^2 + u_2^2 + \cdots + u_d^2 \\ \lambda_2 & = & u_1^2 + (u_2 - 1)^2 + \cdots + u_d^2 \\ & \cdots & \\ \lambda_d & = & u_1^2 + u_2^2 + \cdots + (u_d - 1)^2 \end{array} \tag{2.32}$$

are less than the corresponding components of the generalized prediction; see (2.4):

$$\lambda_1 \leq l_1, \dots, \lambda_d \leq l_d. \tag{2.33}$$

Now suppose we are given a generalized prediction $(L_1, \dots, L_d)$ computed from an unnormalized distribution on the experts; in other words, we are given

$$\begin{array}{rcl} L_1 & = & l_1 + c \\ & \cdots & \\ L_d & = & l_d + c \end{array}$$

for some $c \in \mathbb{R}$. To find (2.32) satisfying (2.33) we can first find the largest $t \in \mathbb{R}$ such that (2.33) is still satisfied for $(L_1 - t, \dots, L_d - t)$ in the right-hand sides, and then find (2.32) satisfying

$$\lambda_1 \leq L_1 - t, \dots, \lambda_d \leq L_d - t. \tag{2.34}$$

Since $t \geq c$, it is clear that $(\lambda_1, \dots, \lambda_d)$ will also satisfy the required (2.33).

Proposition 2.1 *Define $s \in \mathbb{R}$ by the requirement*

$$\sum_{i=1}^{d} (s - L_i)^+ = 2. \tag{2.35}$$

The unique solution to the optimization problem $t \to \max$ under the constraints

(2.34) with $\lambda_1, \ldots, \lambda_d$ as in (2.32) will be

$$u_i = \frac{(s - L_i)^+}{2}, \quad i = 1, \ldots, d, \tag{2.36}$$

$$t = s - 1 - u_1^2 - \cdots - u_d^2. \tag{2.37}$$

PROOF By $\overline{u}_i$ and $\overline{t}$ we denote u_i and t which satisfy (2.36) and (2.37) respectively. Now notice that the ith constraint in (2.34) can be rewritten as

$$\overline{u}_1^2 + \cdots + \overline{u}_d^2 - 2\overline{u}_i + 1 \leq L_i - \overline{t},$$

which immediately follows from (2.36) and (2.37). As a by-product, we can see that the inequality becomes an equality, that is,

$$\overline{t} = L_i - 1 + 2\overline{u}_i - \overline{u}_1^2 - \cdots - \overline{u}_d^2, \tag{2.38}$$

for all i with $\overline{u}_i > 0$.

We can rewrite (2.34) as follows:

$$\begin{cases} t \leq L_1 - 1 + 2u_1 - u_1^2 - \cdots - u_d^2, \\ \qquad\qquad \vdots \\ t \leq L_d - 1 + 2u_d - u_1^2 - \cdots - u_d^2, \end{cases} \tag{2.39}$$

and our goal is to prove that these inequalities imply $t < \overline{t}$ (unless $u_1 = \overline{u}_1, \ldots, u_d = \overline{u}_d$), and so $\overline{t}$ is maximal. Choose i for which $\epsilon_i := \overline{u}_i - u_i$ is maximal (necessarily $\overline{u}_i > 0$ unless $u_1 = \overline{u}_1, \ldots, u_d = \overline{u}_d$; in the latter case, however, we can, and will, also choose $i : \overline{u}_i > 0$). Then every value of t

satisfying (2.39) will also satisfy

$$\begin{aligned} t &\le L_i - 1 + 2u_i - \sum_{j=1}^{d} u_j^2 \\ &= L_i - 1 + 2\overline{u}_i - 2\epsilon_i - \sum_{j=1}^{d} \overline{u}_j^2 + 2\sum_{j=1}^{d} \epsilon_j \overline{u}_j - \sum_{j=1}^{d} \epsilon_j^2 \\ &\le L_i - 1 + 2\overline{u}_i - \sum_{j=1}^{d} \overline{u}_j^2 - \sum_{j=1}^{d} \epsilon_j^2 \le \overline{t}. \end{aligned} \tag{2.40}$$

The penultimate inequality in (2.40) follows from

$$-\epsilon_i + \sum_{j=1}^{d} \epsilon_j \overline{u}_j = \sum_{j=1}^{d} (\epsilon_j - \epsilon_i)\overline{u}_j \le 0.$$

The last inequality in (2.40) follows from (2.38) and becomes strict when not all u_j coincide with $\overline{u}_j$. ■

There exists a unique s satisfying (2.35) since the left-hand side of (2.35) is a continuous, increasing (strictly increasing when positive) and unbounded above function of s. The substitution function is then given by (2.36). We will describe experimental results of applying Algorithm 4 for prediction the results of football and tennis matches in Section 2.6.

2.5 Second-guessing experts

In this section we consider a new setting for prediction with expert advice, where the experts are allowed to "second-guess", that is, to give conditional predictions that are functions of the future learner's prediction. If the dependence is regular enough (the expert's loss is continuous in the learner's loss), the Defensive Forecasting algorithm works in the new setting virtually without changes and guarantees the same performance bound as in the traditional setting. The AA in its original form cannot work in the new setting, and we suggest a modified version of the AA for this case. The short conference version of these results appeared in the ALT 2008 proceedings (Chernov et al., 2008), and the full version appeared in the TCS journal (Chernov et al., 2010).

In game theory, the notion of internal regret (Blum and Mansour, 2007; Foster and Vohra, 1999) is somewhat related to the idea of second-guessing experts. The internal regret appears in the framework where, for each prediction, which is called action in that context, there is an expert that consistently recommends this action, and the learner follows one of the experts at each step. The internal regret for a pair of experts i, j shows by how much the learner could have decreased his loss if he had followed expert j each time he followed expert i. This can be modeled by a second-guessing expert that "adjusts" the learner's predictions: this second-guessing expert agrees with the learner if the learner does not follow i, and recommends following j when the learner follows i.

The internal regret is usually studied in randomized prediction protocols. In the case when the learner gives deterministic predictions, one cannot hope to get any interesting loss bound without additional assumptions. Indeed, the experts can always suggest exactly the "opposite" to the learner's prediction (for example, in the log loss game, they predict 1 if the learner predicts ("the probability of 1") less than 0.5 and they predict 0 otherwise), and the reality can "agree" with them (choosing the outcome equal to the experts' prediction); then the experts' losses remain zero, but the learner's loss grows linearly in the number of steps. The results of Blum and Mansour (2007) and others are bounds of the form $L_T \le L_T^\theta + O(\sqrt{T})$ for the Freund-Schapire game (see

Vovk, 1998, Example 7), which is not mixable. In this section we consider another kind of bounds. However, here we will also make the assumption that second-guessing experts modify the prediction of the learner continuously.

2.5.1 Game with second-guessing experts

The game with second-guessing experts proceeds according to the following protocol.

Protocol 2 Prediction with second-guessing expert advice

$L_0 := 0$.
$L_0^\theta := 0$, $\theta \in \Theta$.
for $t = 1, 2, \ldots$ **do**
  Experts announce $\xi_t^\theta : \Gamma \to \Gamma$, $\theta \in \Theta$.
  Learner announces $\gamma_t \in \Gamma$.
  Reality announces $\omega_t \in \Omega$.
  $L_t := L_{t-1} + \lambda(\omega_t, \xi_t)$.
  $L_t^\theta := L_{t-1}^\theta + \lambda(\omega_t, \xi_t^\theta(\gamma_t))$, $\theta \in \Theta$.
end for

The new protocol contains only one substantial change from Protocol 1. The experts announce functions ξ_t^θ from Γ to Γ instead of an element of Γ. Therefore, the loss of each expert is determined by the learner's prediction as well as by the outcome chosen by reality. We call the experts in this protocol *second-guessing experts*. Second-guessing experts are a generalization of the experts in the standard protocol: a standard expert can be interpreted in the new protocol as predicting a constant function.

The phenomenon of second-guessing experts occurs, for example, in real-world finance. In particular, commercial banks serve as second-guessing experts for the central bank when they use variable interest rates (that is, the interest rate for the next period is announced not as a fixed value but as an explicit function of the central bank base rate).

We discuss the simple game $\Omega = \{0, 1\}, \Gamma = [0, 1]$, even though the following reasoning holds for some more general cases.

2.5.2 Defensive Forecasting

Defensive Forecasting requires virtually no modifications for this task and gives the same loss bounds as in Theorems 2.4 and 2.6.

Theorem 2.8 *Assume experts' predictions are continuous functions of the learner's prediction. Using Algorithm 2 as the learner's strategy in Protocol 1 for the game with the log-loss function guarantees that, for all $T = 1, 2, \ldots$,*

$$L_T \le \min_{\theta=1,\ldots,K} L_T^\theta + \ln K. \tag{2.41}$$

For the game with the square loss function it guarantees that, for all $T = 1, 2, \ldots$,

$$L_T \le \min_{\theta=1,\ldots,K} L_T^\theta + \frac{\ln K}{2}. \tag{2.42}$$

PROOF Clearly, function $Q_t^\theta = e^{\eta\left(\lambda(\omega_t,\gamma_t)-\lambda(\omega_t,\xi_t^\theta(\gamma_t))\right)}$ from (2.8) is forecast-continuous as a composition of continuous functions. By Lemmas 2.4 and 2.6 we have that Q_t^θ has the defensive property for $\eta = 1$ ($\eta = 2$ for the square loss): it has the property for any fixed γ_t (because the lemmas hold for any expert's predictions from Γ). Thus the mixture $Q_T = \sum_{\theta=1}^K \prod_{t=1}^T Q_t^\theta$ is a forecast-continuous sequence having the defensive property by Lemma 2.3, and by Lemma 2.2 for any T there exists $\gamma_T \in \Gamma$ such that $Q_T \le 1$. Thus

$$\prod_{t=1}^T e^{\eta\lambda(\omega_t,\gamma_t)-\eta\lambda(\omega_t,\xi_t^\theta(\gamma_t))} \le K.$$

Taking natural logarithms of both sides completes the proof. ∎

It may be surprising to see that we do not loose anything by generalizing the experts; but in terms of the upper bounds having finite number of continuous experts is equivalent to having finite number of experts predicting numbers directly.

2.5.3 Aggregating Algorithm

As opposed to the DF, the AA needs substantial improvement to be able to compete with second-guessing experts. Recall that the AA is looking for a

prediction satisfying the inequality (2.4), containing the prediction γ_t only in the left-hand side.

In the second-guessing protocol, both sides of this inequality will contain γ_t:

$$\lambda(\omega, \gamma_t) \le \log_\beta \int_\Theta \beta^{\lambda(\omega, \xi_t^\theta(\gamma_t))} P_{t-1}^*(\theta).$$

The DF implicitly solves this inequality in the proof of Lemma 2.2 using a kind of fixed point theorem. We will present a modification of the AA which uses a fixed point theorem explicitly.

A topological space X has the *fixed point property* if every continuous function $f : X \to X$ has a fixed point, that is, $\exists x \in X\ f(x) = x$. It follows from Theorem 4.10 of Agarwal et al. (2001) that the set $\Gamma = [0, 1]$ has the fixed point property. Thus in order to compete with second-guessing experts the AA needs its substitution function Σ to be continuous. In particular, for the log-loss function (we obtain this from (2.7)) we have

$$\gamma = \Sigma(g) = \frac{1}{1 + e^{g(1) - g(0)}} = e^{-g(1)},$$

and for the square loss function we have

$$\gamma = \Sigma(g) = \frac{1}{2} + \frac{g(0) - g(1)}{2}.$$

The continuous substitution function presented in Lemma 27 of Chernov et al. (2010) is suitable for more general types of games, which we do not consider here.

We modify the Aggregating Algorithm such that at each step it announces as the learner's prediction γ_t any solution of the following equation with respect to $\gamma \in \Gamma$:

$$\gamma = \Sigma\left(\log_\beta \int_\Theta \beta^{\lambda(\omega, \xi_t^\theta(\gamma))} P_{t-1}^*(d\theta)\right), \tag{2.43}$$

where ξ_t^θ are announced by the experts and Σ is a continuous mapping, like one of the standard mappings mentioned above.

Theorem 2.9 *Assume experts' predictions are continuous functions of the*

learner's prediction. Using the modified Aggregating Algorithm as the learner's strategy in Protocol 2 for the game with the log loss function guarantees that, for all $T = 1, 2, \ldots$,

$$L_T \le \min_{\theta=1,\ldots,K} L_T^\theta + \ln K. \tag{2.44}$$

For the game with the square loss function it guarantees that for all $T = 1, 2, \ldots$,

$$L_T \le \min_{\theta=1,\ldots,K} L_T^\theta + \frac{\ln K}{2}. \tag{2.45}$$

PROOF Since $\Gamma = [0, 1]$ has the fixed point property and the mapping given by (2.43) is continuous as the composition of continuous mappings, we see that equation (2.43) has a solution.

The property $\lambda(\omega, \Sigma(g)) \le g(\omega), \forall \omega$ of Σ implies that

$$\lambda(\omega, \gamma_t) \le \log_\beta \int_\Theta \beta^{\lambda(\omega, \xi_t^\theta(\gamma_t))} P_{t-1}^*(d\theta), \quad \forall \omega$$

and the usual analysis of the AA gives us the bounds. ∎

2.6 Prediction the results of sports matches

In this section we investigate the practical performance of different algorithms for prediction with expert advice. We use real world data sets containing the results of football and tennis matches over past years. We first apply the Aggregating Algorithm (Algorithm 4) and measure the quality of its predictions by the Brier loss function (2.29). Then we show that algorithms designed to compete under the Brier loss function can fail if the quality of predictions is measured by the log-loss function, and vice versa. Applying the Defensive Forecasting algorithm (Algorithm 2) allows us to predict with a performance guarantee under both loss functions simultaneously. Finally, we compare the performance of other algorithms on our data sets. Some of the results of this section are described in Vovk and Zhdanov (2008) and in more details in Vovk and Zhdanov (2009).

2.6.1 Data sets

We use two data sets. The first one contains historical data about 8999 matches in various English football league competitions, namely: the Premier League (the pinnacle of the English football system), the Football League Championship, Football League One, Football League Two, and the Football Conference. Our data, provided by Football-Data, cover four seasons: 2005/2006, 2006/2007, 2007/2008, and 2008/2009.

The matches are sorted first by date, then by league, and then by the name of the home team. The outcome of each match takes one of three possible values, "home win", "draw", or "away win"; we will encode the possible values as 1, 2, and 3.

For each match we have forecasts made by a range of bookmakers. We use eight bookmakers for which we have enough data over a long period of time, namely: Bet365, Bet&Win, Gamebookers, Interwetten, Ladbrokes, Sportingbet, Stan James, and VC Bet. The seasons mentioned above are chosen because the forecasts of these bookmakers are available for them.

The second data set involves data about a large number of tennis tournaments in 2004, 2005, 2006, and 2007, with the total number of matches $10,087$.

The tournaments include, for example, Australian Open, French Open, US Open, and Wimbledon; the data is provided by Tennis-Data.

The matches are sorted by date, then by tournament, and then by the winner's name. The data contain information about the winner of each match and the betting odds of 4 bookmakers for his/her win and for the opponent's win. Therefore, there are two possible outcomes (player 1's win and player 2's win). There are four bookmakers: Bet365, Centrebet, Expekt, and Pinnacle Sports.

The data used for all the experiments in this section can be downloaded from `http://vovk.net/ICML2008`.

2.6.2 Experimental setup

Let us consider the football case, the case with three possible outcomes. A probability forecast for each outcome is essentially a vector (π_1, π_2, π_3) consisting of positive numbers summing to 1: $\pi_1 + \pi_2 + \pi_3 = 1$.

The bookmakers do not announce these numbers directly; instead, they quote three betting odds, a_1, a_2, and a_3. Each number $a_i > 1$ is the total amount which the bookmaker undertakes to pay out to a client betting on outcome i per unit stake in the event that i happens (if the bookmaker wishes to return the stake to the bettor, it should be included in a_i; i.e., the odds are announced according to the "continental" rather than "traditional" system). The inverse value $1/a_i$, $i \in \{1, 2, 3\}$, can be interpreted as the bookmaker's quoted probability for the outcome i. The bookmaker's quoted probabilities are usually slightly (because of the competition with other bookmakers) in his favour: the sum $1/a_1 + 1/a_2 + 1/a_3$ exceeds 1 by the amount called the *overround* (at most 0.15 in the vast majority of cases). We use a different formula

$$\pi_i := a_i^{-\gamma}, \quad i = 1, 2, 3, \tag{2.46}$$

(suggested by Khutsishvili, 2009) to compute the bookmaker's probability forecasts, where $\gamma > 0$ is chosen such that $a_1^{-\gamma} + a_2^{-\gamma} + a_3^{-\gamma} = 1$. Such a value of γ exists and is unique since the function $a_1^{-\gamma} + a_2^{-\gamma} + a_3^{-\gamma}$ continuously and strictly decreases from 3 to 0 as γ changes from 0 to ∞. In practice, we usually

have $\gamma > 1$ as $a_1^{-1} + a_2^{-1} + a_3^{-1} > 1$ (i.e., the overround is positive). The method of bisection was more than sufficient for us to solve $a_1^{-\gamma} + a_2^{-\gamma} + a_3^{-\gamma} = 1$ with satisfactory accuracy. Khutsishvili's argument for (2.46) is outlined in the next subsection.

Typical values of γ in (2.46) are close to 1, and the difference $\gamma - 1$ reflects the bookmaker's target profit margin. In this respect $\gamma - 1$ is similar to the overround; indeed, the approximate value of the overround is $(\gamma - 1) \sum_{i=1}^{3} a_i^{-1} \ln a_i$ assuming that the overround is small and none of a_i is too close to 0 and using $a_i^{1-\gamma} - 1 \approx \ln a_i^{1-\gamma}$. The coefficient of proportionality $\sum_{i=1}^{3} a_i^{-1} \ln a_i$ can be interpreted as the entropy of the quoted betting odds.

2.6.3 Khutsishvili's theory

The standard formula (following, e.g., Pennock et al., 2001, Section 7 and Sauer, 2005, pp. 419–420) to convert bookmakers' betting odds into their quoted probabilities is expressed as follows:

$$p_i := \frac{1/a_i}{1/a_1 + 1/a_2 + 1/a_3}, \quad i = 1, 2, 3, \tag{2.47}$$

in place of (2.46).

We first compare the formulas (2.46) and (2.47) empirically by calculating the Brier losses of the predictions of the bookmakers found using these formulas (see Tables 2.1 and 2.2). As we can see from the tables, using Khutsishvili's formula (2.46) consistently leads to smaller losses. The improvement of each bookmaker's total loss over the football data set is in the range 0.72–5.84; over the tennis data set the difference is in the range 1.27–11.64. These differences are of the order of the differences in cumulative loss between different bookmakers, and so the improvement is significant.

We will further present the Khutsishvili's argument for (2.46). The theory is based on a very idealized model of a bookmaker, who is assumed to compute the betting odds a for an event of probability p using a continuous function f,

$$a := f(p).$$

Table 2.1: The bookmakers' cumulative Brier losses over the football data set when their probability forecasts are computed using formula (2.46) and formula (2.47).

Loss resulting from (2.46)	Loss resulting from (2.47)	Difference
5585.69	5588.20	2.52
5585.94	5586.67	0.72
5586.60	5587.37	0.77
5588.47	5590.65	2.18
5588.61	5589.92	1.31
5591.97	5593.48	1.52
5596.01	5601.85	5.84
5596.56	5598.02	1.46

Table 2.2: The bookmakers' cumulative Brier losses over the tennis data set when their probability forecasts are computed using formula (2.46) and formula (2.47).

Loss resulting from (2.46)	Loss resulting from (2.47)	Difference
3935.32	3944.02	8.69
3943.83	3945.10	1.27
3945.70	3957.33	11.64
3953.83	3957.75	3.92

Different bookmakers (and the same bookmaker at different times) can use different functions f.

The following simple corollary of Darboux's theorem describes the set of possible functions f; its interpretation will be discussed immediately after the proof.

Theorem 2.10 *Suppose a function $f : (0,1) \to (1,\infty)$ satisfies the condition*

$$f(pq) = f(p)f(q) \tag{2.48}$$

for all $p, q \in (0,1)$. There exists $c > 0$ such that $f(p) = p^{-c}$ for all $p \in (0,1)$.

PROOF Equation (2.48) is one of the four fundamental Cauchy equations, which can be easily reduced to each other. For example, introducing a new function $g : (0,\infty) \to (0,\infty)$ by $g(u) := \ln f(e^{-u})$ and new variables $x, y \in (0,\infty)$ by $x := -\ln p$ and $y := -\ln q$, we transform (2.48) to the most standard Cauchy equation $g(x+y) = g(x)+g(y)$. By Darboux's theorem (see, e.g., Aczél, 1966, Section 2.1, Theorem 1), $g(x) = cx$ for all $x > 0$, that is, $f(p) = p^{-c}$ for all $p \in (0,1)$. ∎

The function f is defined on $(0,1)$ (no bookmaker assigns a subjective probability of exactly 0 or 1 to an event on which he accepts bets). It would be irrational for the bookmaker to have $f(p) \le 1$ for some p, so $f : (0,1) \to (1,\infty)$.

To see that the requirement (2.48) is reasonable, we should take into account not only "single" but also "double" bets. If a bookmaker quotes two odds a and b on two independent (by his opinion) events, his quoted odds on the conjunction of the two events should be ab. If the probabilities of the two events are p and q, respectively, the probability of their conjunction will be pq. Therefore, we have (2.48).

We can see from Theorem 2.10 that if the bookmakers apply the same function f to all three probabilities p_1, p_2, and p_3, the formula $f(p) = p^{-c}$ for the function converting predicted probabilities into odds follows. We have $p_i = a_i^{-\gamma}$, where $\gamma = 1/c$ and $i = 1,2,3$, and γ can be found from the requirement $p_1 + p_2 + p_3 = 1$.

An important advantage of (2.46) over (2.47) is that (2.46) does not impose any upper limits on the overround that the bookmaker may charge (Khutsishvili, 2009). If the game has n possible outcomes ($n = 3$ for football and $n = 2$ for tennis) and the bookmaker uses $f(p) = p^{-c}$, the overround is

$$\sum_{i=1}^{n} a_i^{-1} - 1 = \sum_{i=1}^{n} p_i^c - 1$$

and so continuously changes between -1 and $n-1$ as c ranges over $(0,\infty)$ (in practice, the overround is usually positive, and so $c \in (0,1)$). Even for $n = 2$, the upper bound of 1 is too large to be considered a limitation. The situation with (2.47) is very different: upper bounding the numerator of (2.47) by 1 and replacing the denominator by $1 + o$, where o is the overround, we obtain

$p_i < \frac{1}{1+o}$ for all i, and so $o < \min_i p_i^{-1} - 1$; this limitation on o is restrictive when one of the p_i is close to 1.

An interesting phenomenon in racetrack betting (on horses, cars, etc.), known since Griffith (1949), is that favourites are usually underbet while longshots are overbet (see, e.g., Snowberg and Wolfers, 2007, for a recent survey and analysis). Khutsishvili's formula (2.46) can be regarded as a way of correcting this "favourite-longshot bias": when a_i is large (the outcome i is considered a longshot), (2.46) slashes $1/a_i$ when computing p_i more than (2.47) does.

2.6.4 Experimental results

The probabilities calculated from (2.46) of the bookmakers' are used as the experts' predictions in our experiments. The results of applying Algorithm 4 to the football data, with 8 experts and 3 possible outcomes, are shown in Figure 2.2. Let L_T^θ be the cumulative loss of the expert θ, $\theta = 1, \dots, 8$, over the first T matches and L_T be the corresponding number for the Aggregating Algorithm. The dashed line corresponding to the expert θ shows the excess loss $T \mapsto L_T^\theta - L_T$ of the expert θ over the Aggregating Algorithm. The excess loss can be negative, but from the first part of Theorem 2.7 we know that it cannot be less than $-\ln 8$; this lower bound is also shown in Figure 2.2. Finally, the thick line (the positive part of the x axis) is drawn for comparison: this is the excess loss of the Aggregating Algorithm over itself. We can see that at each moment in time the algorithm's cumulative loss is fairly close to the cumulative loss of the best expert (at that time; the best expert keeps changing over time).

Figure 2.3 shows the distribution of the bookmakers' overrounds. We can see that in most cases overrounds are between 0.05 and 0.15, but there are also occasional extreme values, near zero or in excess of 0.3.

The results in Figure 2.4 show the results of the experiments with the tennis data. They are presented in the same way as in Figure 2.2. Typical values of the overround are below 0.1, as shown in Figure 2.5 (analogous to Figure 2.3). As in football data, tennis bookmakers have some extreme values of the overround, like 0.48 or negatives. We can explain them as in the beginning of

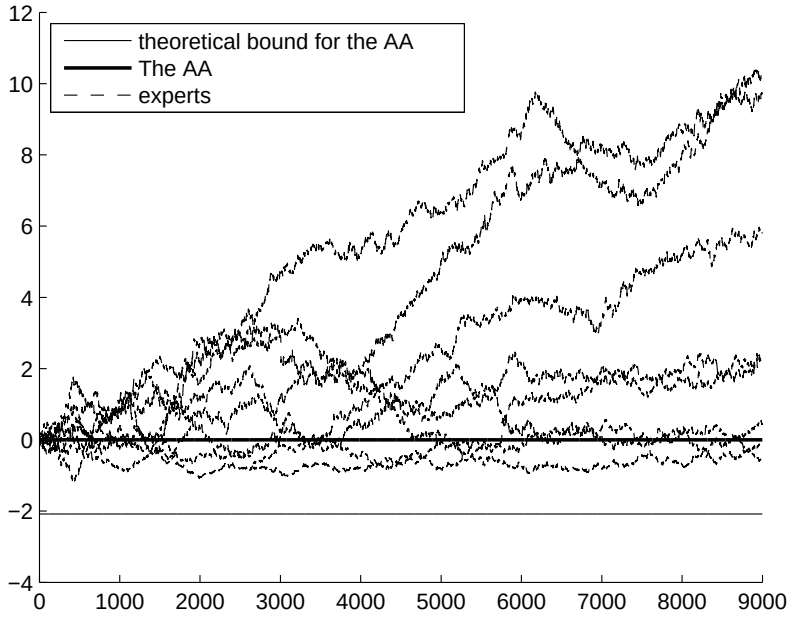


Figure 2.2: The difference between the cumulative loss of each of the 8 bookmakers (experts) and of the AA on the football data. The lower bound $-\ln 8$ obtained from Theorem 2.7 is also shown.

some games a bookmaker makes incorrect forecasts, and thus incorrect bets. The bookmaker's welfare depends not only on the accuracy of his forecasts, but also on the amount of people who stake on each event. Thus the bets can not be easily changed during the betting process, they have to be changed dramatically to attract more people. Just before the beginning of the match the bookmaker has to "save" his capital, and tries to do it by changing his bets. We have closing bets for the bookmakers and we see the result of these tricks. It is important to note that there are few of such forecasts having the overround which is less than zero or more than 0.2: 9 matches out of 10087 for all bookmakers, 8 matches for one of them, and 1 for another. The majority of the overrrounds included by the bookmakers have small values, and there is good reason to think that the extreme overrounds have no strong influence upon the real forecasts extracted from the values of bets using formula (2.46).

In both Figure 2.2 and Figure 2.4 the cumulative loss of the AA is close to the cumulative loss of the best expert. The bound (2.30) is not hopelessly

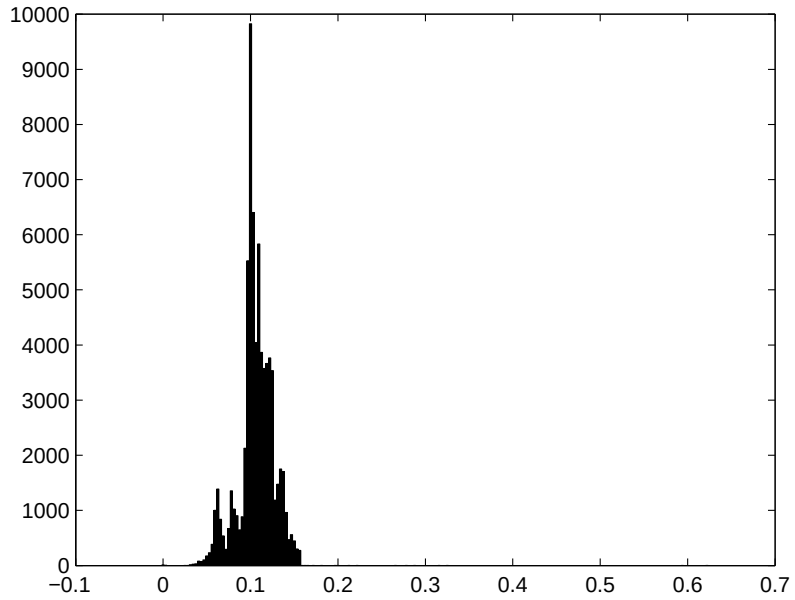


Figure 2.3: The overround distribution histogram for the football data, with 200 bins of equal size between the minimum and maximum values of the overround. The horizontal axis gives the bin, the vertical axis gives the number of forecasts in each bin.

loose for the football data and is rather tight for the tennis data. The pictures look almost the same when the AA is applied in the more realistic manner where the experts' weights $P(d\theta)$ are not updated over the matches that are played simultaneously (on the same day). Thus we do not include the pictures with the results of these experiments.

Our second empirical study (Figure 2.4) covers only binary prediction and considers prediction under multiple loss functions.

2.6.5 Prediction under multiple loss functions simultaneously

As described at the end of Section 2.2.2, the method of defensive forecasting makes it possible to compete with a set of experts with respect to several loss functions at once. We apply it to the predictions made by the bookmakers and show that the AA working with only one of the loss functions can fail if

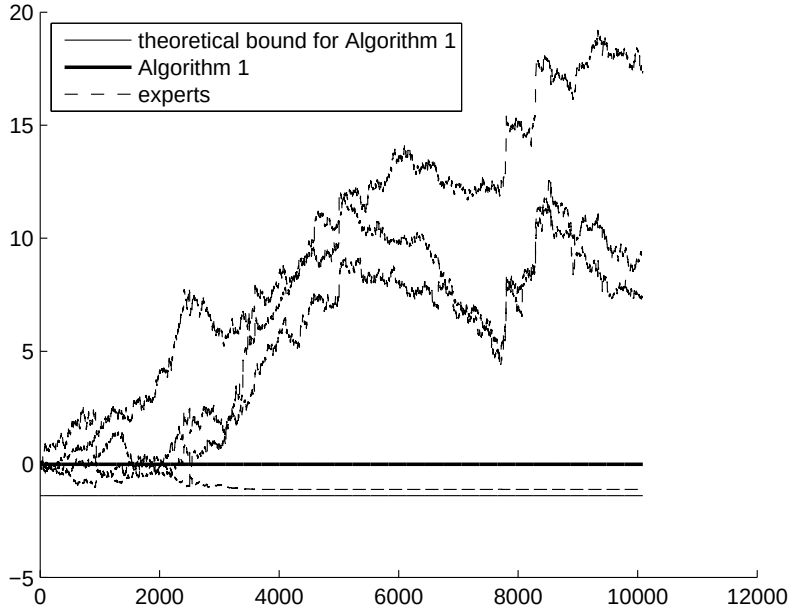


Figure 2.4: The difference between the cumulative loss of each of the 4 bookmakers and of the AA on the tennis data. Now the lower bound is $-\ln 4$.

the performance is measured by another loss function.

Experimental results

In our experiments we will use the Brier loss function (2.29) (doubled square loss (2.24) in case of two possible outcomes) and the logarithmic loss function (2.19). We show pictures only for the experiments with tennis matches, with two possible outcomes.

As we have seen earlier in Theorem 2.5, if the learner uses the Aggregating Algorithm, he can ensure that

$$L_T \le L_T^\theta + \ln K \tag{2.49}$$

in the traditional game of prediction with expert advice following Protocol 1 with the Brier loss function. Similarly, Theorem 2.3 implies that if the learner uses the Bayesian Algorithm he can ensure (2.49) in the traditional game of

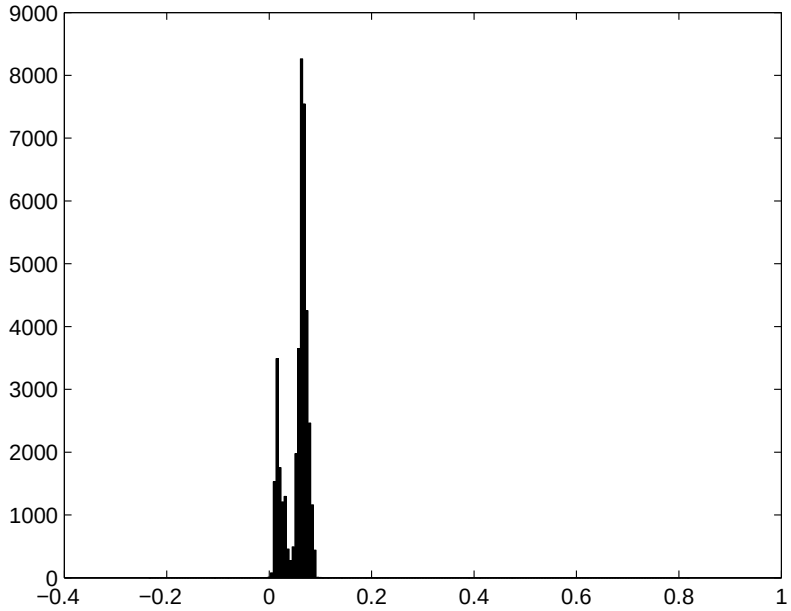


Figure 2.5: The overround distribution histogram for the tennis data. The horizontal axis gives the bin, the vertical axis gives the number of forecasts in each bin.

prediction with expert advice with the log loss function.

As in the previous section, a probability forecast for the next outcome is a vector (π_1, π_2) consisting of non-negative numbers summing to 1. For simplicity in this experiment, we convert betting odds into probabilities simply by normalizing $1/a_n$. In other words, we use the simple formula

$$\pi_i := \frac{1/a_i}{1/a_1 + 1/a_2}, \quad i = 1, 2, \tag{2.50}$$

for computing the bookmaker's probability forecasts.

In the binary case the Defensive Forecasting algorithm can be implemented efficiently using the simple method of bisection. The results of applying the DF, the AA, and the BA to the tennis data, with 4 experts and 2 possible outcomes, for both loss functions are shown in Figures 2.6–2.11. The dashed lines corresponding to the experts again show the excess loss $T \mapsto L_T^\theta - L_T$ of the experts over the given algorithm.

On all the pictures we also show the existing lower bounds on the excess

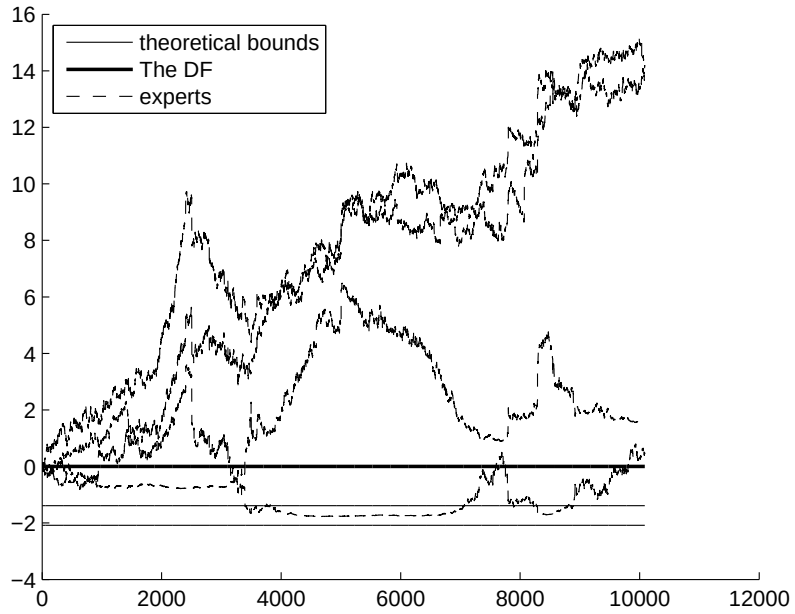


Figure 2.6: The difference between the cumulative loss of each of the 4 bookmakers and of the DF algorithm on the tennis data for the Brier loss.

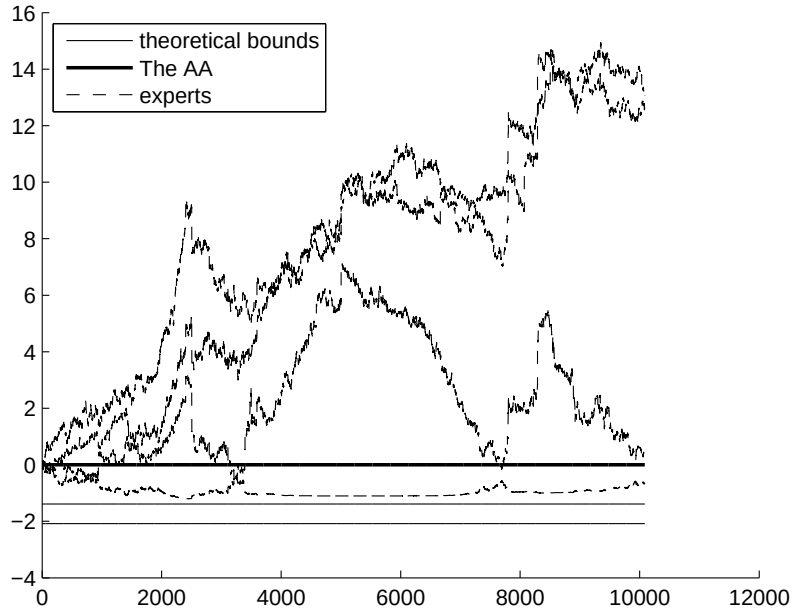


Figure 2.7: The difference between the cumulative loss of each of the 4 bookmakers and of the AA on the tennis data for the Brier loss.

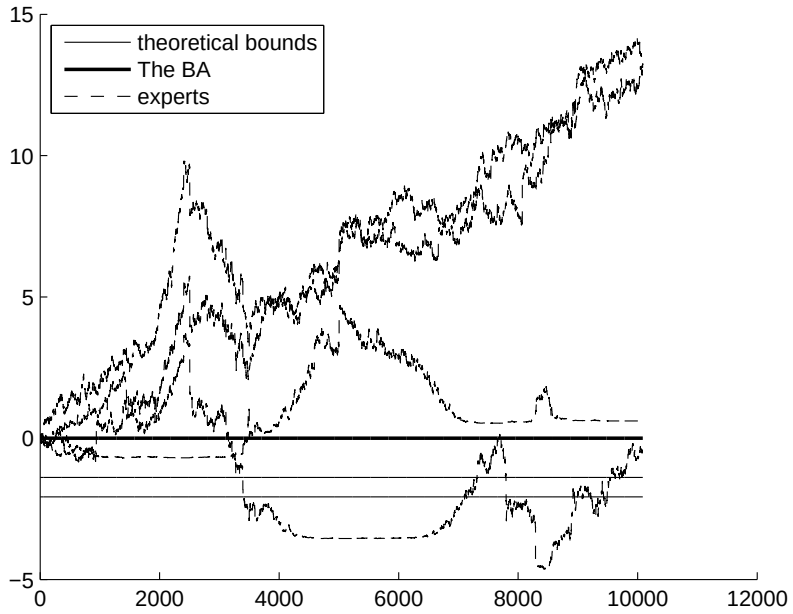


Figure 2.8: The difference between the cumulative loss of each of the 4 bookmakers and of the BA on the tennis data for the Brier loss.

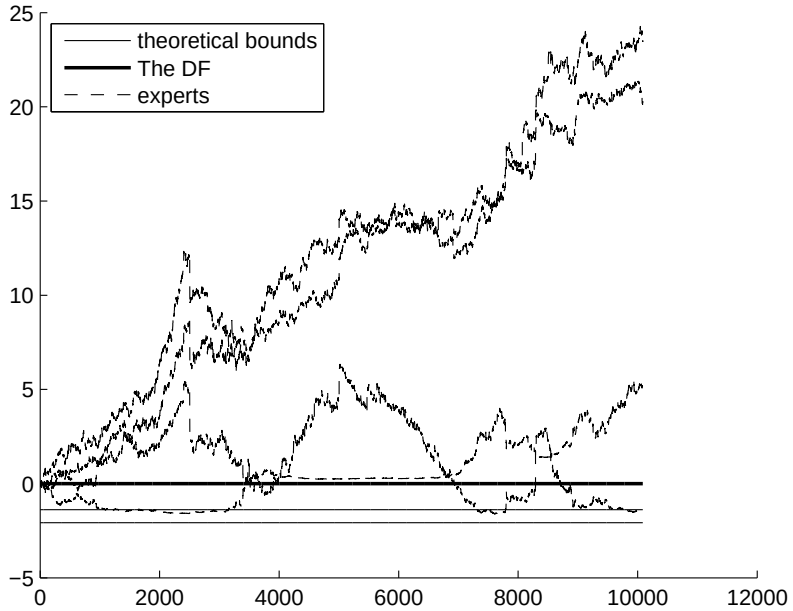


Figure 2.9: The difference between the cumulative loss of each of the 4 bookmakers and of the DF algorithm on the tennis data for the log loss.

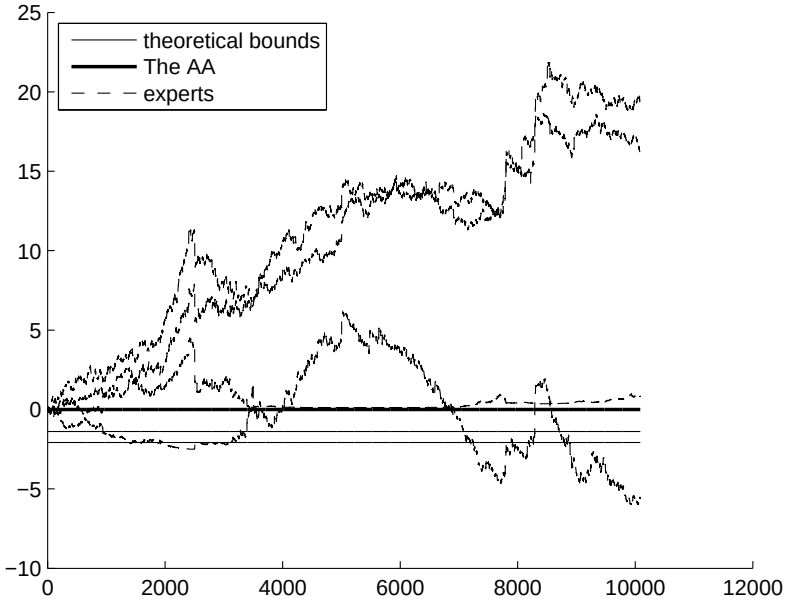


Figure 2.10: The difference between the cumulative loss of each of the 4 bookmakers and of the AA on the tennis data for the log loss.

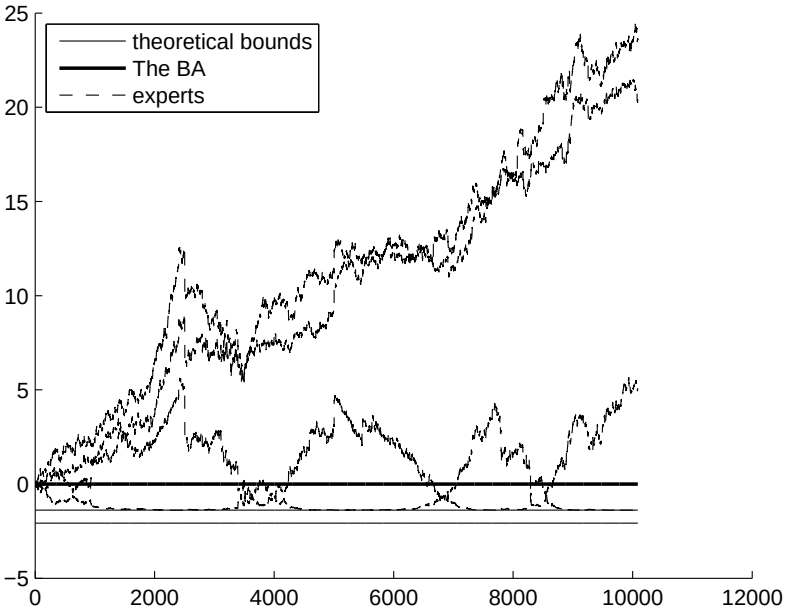


Figure 2.11: The difference between the cumulative loss of each of the 4 bookmakers and of the BA on the tennis data for the log loss.

loss of the experts over our prediction algorithms: $-\ln 8$ for the Defensive Forecasting algorithm (see Theorem 2.1), and $-\ln 4$ for the AA under the Brier loss and for the BA under the log loss (see (2.49)). The guaranteed lower bound is $-\ln 8$ for Figures 2.6 and 2.9, and the guaranteed lower bound is $-\ln 4$ for Figures 2.7 and 2.11; there are no guarantees whatsoever for Figures 2.8 and 2.10 (and it shows). We can see that at each moment in time the relevant algorithm's cumulative loss is fairly close to the cumulative loss of the best expert (at that time; the best expert keeps changing over time), provided there are theoretical guarantees for the given combination of the prediction algorithm and the loss function.

Table 2.3 gives precise values for the maximal differences $\max_{\theta,T}(L_T - L_T^\theta)$ and the upper bounds for them.

Table 2.3: The maximal difference between the loss of each prediction algorithm (first column) and the loss of the best expert for the tennis data under the two loss functions (second column); the upper bound on this difference (third column).

Algorithm and loss	Max. diff.	Bound
DF for Brier loss	1.7919	2.0794
AA for Brier loss	1.2021	1.3863
BA for Brier loss	4.6531	none
DF for log loss	1.6817	2.0794
AA for log loss	6.0410	none
BA for log loss	1.3854	1.3863

The results that we obtained for the tennis data set are sensitive to the way of converting betting odds into probabilities. Replacing the standard formula (2.50) by the less standard (2.46) may lead to different results.

The principal advantage of the DF algorithm is not that it always gives more precise predictions than more naive prediction strategies, but that there is a guarantee that it will never suffer failures such as those in Figures 2.8 and 2.10. The loss bound (2.17) guarantees decent performance under any of the selected loss functions.

2.6.6 Comparison with other prediction algorithms

It is possible to apply other popular algorithms for prediction with expert advice instead of Algorithm 4 (i.e., the AA) to predict the results of the sports matches. These are, for example, the Weighted Average Algorithm (WdAA, proposed by Kivinen and Warmuth, 1999), the Weak Aggregating Algorithm (WkAA, proposed independently by Kalnishkan and Vyugin, 2005, and Cesa-Bianchi and Lugosi, 2006, Theorem 2.3; we are using Kalnishkan and Vyugin's name), and the Hedge algorithm (HA, proposed by Freund and Schapire, 1997, we apply it in randomized form).

Since neither the WkAA nor the HA satisfy bound of the form (2.31), we pay particular attention to the WdAA. To extract probabilities from the quoted betting odds we use the formula (2.46). The reader can consult Vovk and Zhdanov, 2007, for details of experiments with the latter two algorithms and formula (2.47). We also briefly discuss three more naive algorithms.

The WdAA is also an exponential weights algorithm, as the AA. It keeps the same weights (2.2) and gives its prediction simply as the weighted average of the expert's predictions (in comparison to the AA which uses a more sophisticated scheme). The theoretical guarantee for the performance of the WdAA is weaker than the optimal (2.30).

Figures 2.12 and 2.13 show the performance of this algorithm in the same format as before (see Figures 2.2 and 2.4). We can see from the figures that for the football data the maximal difference between the cumulative loss of the WdAA and the cumulative loss of the best expert is slightly larger than that for the AA but still well within the optimal bound $\ln K$ given by (2.30). For the tennis data, the maximal difference is almost twice as large as for the AA, violating the optimal bound $\ln K$.

Kivinen and Warmuth, 1999, the beginning of Section 6, proposed that the WdAA works in the game where the outcome set and the prediction set are the unit ball in $\mathbb{R}^n$. This covers the Brier game (see Section 2.4). However, in the Brier game the predictions are known to belong to the simplex $\{(u_1, \dots, u_n) \in [0, \infty)^n | \sum_{i=1}^n u_i = 1\}$, and the outcome is known to be one of the vertices of this simplex.

If we want the WdAA to work in the Brier game and have theoretical

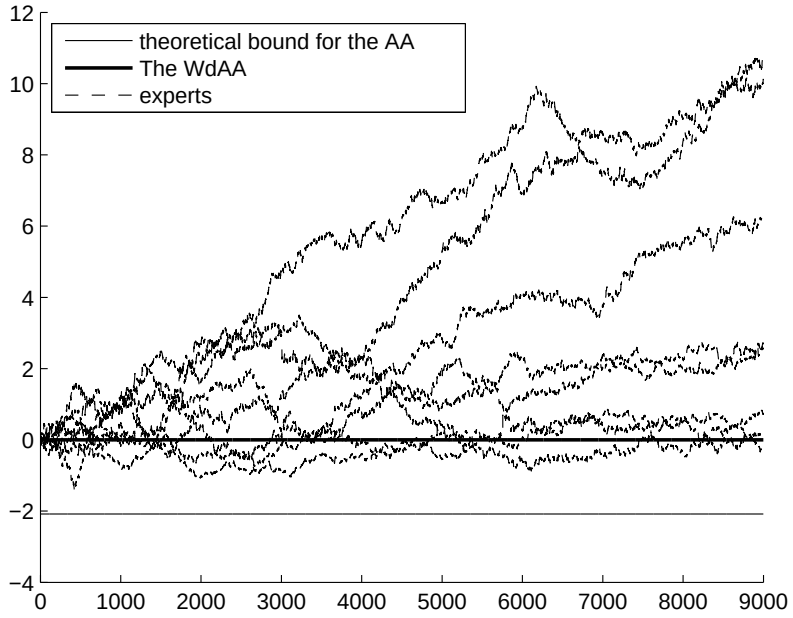


Figure 2.12: The difference between the cumulative loss of each of the 8 bookmakers and of the Weighted Average Algorithm (WdAA) on the football data. The chosen value of the parameter $c = 1/\eta$ for the WdAA, $c := 16/3$, minimizes its theoretical loss bound. The theoretical lower bound $-\ln 8 \approx -2.0794$ for the Aggregating Algorithm is also shown (the theoretical lower bound for the WdAA, -11.0904, can be extracted from Table 2.4 below).

guarantees on its loss, we can consider the smallest ball containing the simplex. This will optimize the radius of the ball which is used in the guarantee proved by Kivinen and Warmuth, 1999. The radius of the smallest ball is

$$R := \sqrt{1 - \frac{1}{n}} \approx \begin{cases} 0.7071 & \text{if } n = 2 \\ 0.8165 & \text{if } n = 3 \\ 1 & \text{if } n \text{ is large.} \end{cases}$$

The WdAA is parameterized by $c := 1/\eta$ instead of η, and the optimal value of c is $c = 8R^2$, leading to the loss bound

$$L_T \le \min_{\theta=1,\dots,K} L_T^\theta + 8R^2 \ln K$$

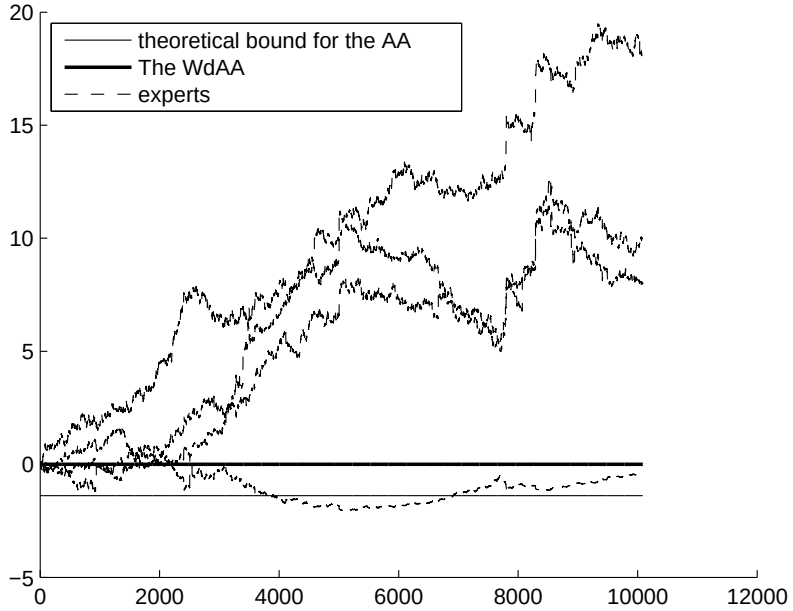


Figure 2.13: The difference between the cumulative loss of each of the 4 bookmakers and of the WdAA for $c := 4$ on the tennis data.

for all $T = 1, 2, \ldots$. This is significantly looser than the bound (2.30) for the AA.

In Figures 2.12 and 2.13 we use the values $c = 16/3$ and $c = 4$ because they minimize the theoretical guarantee for the loss of the WdAA's. However, to minimize the empirical loss, it is possible that they are not the best parameters for the data sets which we use. Figure 2.14 shows the maximal difference

$$\max_{T=1,\ldots,8999} \left(L_T(c) - \min_{\theta=1,\ldots,8} L_T^\theta \right), \tag{2.51}$$

where $L_T(c)$ is the loss of the WdAA with parameter c on the football data over the first T steps and L_T^θ is the analogous loss of the θth expert, as a function of c. Similarly, Figure 2.15 shows the maximal difference

$$\max_{T=1,\ldots,10087} \left(L_T(c) - \min_{\theta=1,\ldots,4} L_T^\theta \right) \tag{2.52}$$

for the tennis data. As we can see, in both cases the value of c minimizing the

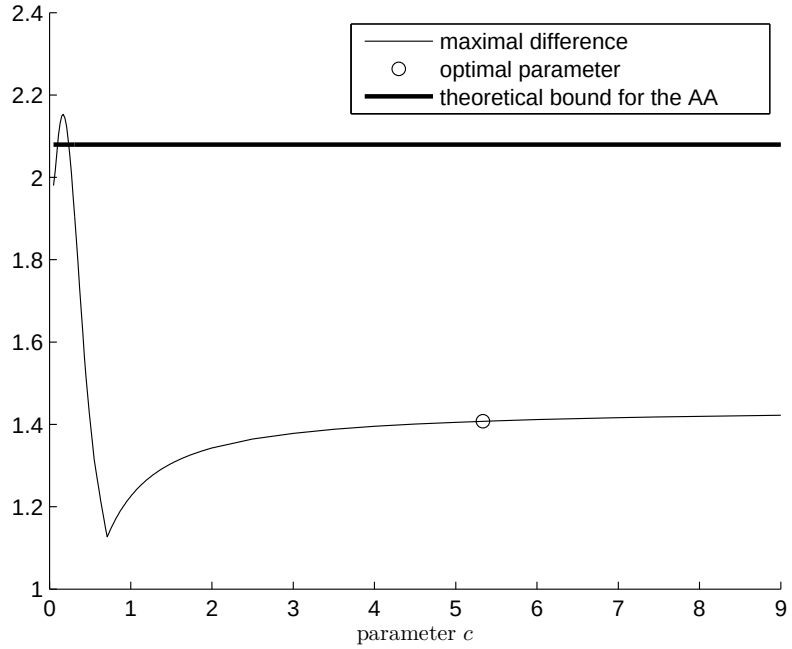


Figure 2.14: The maximal difference (2.51) for the WdAA as function of the parameter c on the football data. The theoretical guarantee $\ln 8$ for the maximal difference for the AA is also shown (the theoretical guarantee for the WdAA, 11.0904, is given in Table 2.4).

empirical loss is far from the value minimizing the bound; as could be expected, the empirical optimal value for the WdAA is not so different from the optimal value for the AA. The following two figures, 2.16 and 2.17, demonstrate that there is no such anomaly for the AA.

Figures 2.18 and 2.19 show the behaviour of the WdAA for the value of parameter $c = 1$, that is, $\eta = 1$, that is optimal for the AA. They look very similar to Figures 2.2 and 2.4, respectively.

Precise numbers associated with the figures referred to above are given in Tables 2.4 and 2.5: the second column gives the maximal differences (2.51) and (2.52), respectively. The third column gives the theoretical upper bound on the maximal difference (i.e., the optimal value of A in (2.31), if available).

Other two algorithms which we use in our studies are the weak aggregating algorithm (WkAA) and the Hedge algorithm (HA). They make weaker assumptions about the prediction game. The Aggregating Algorithm computes

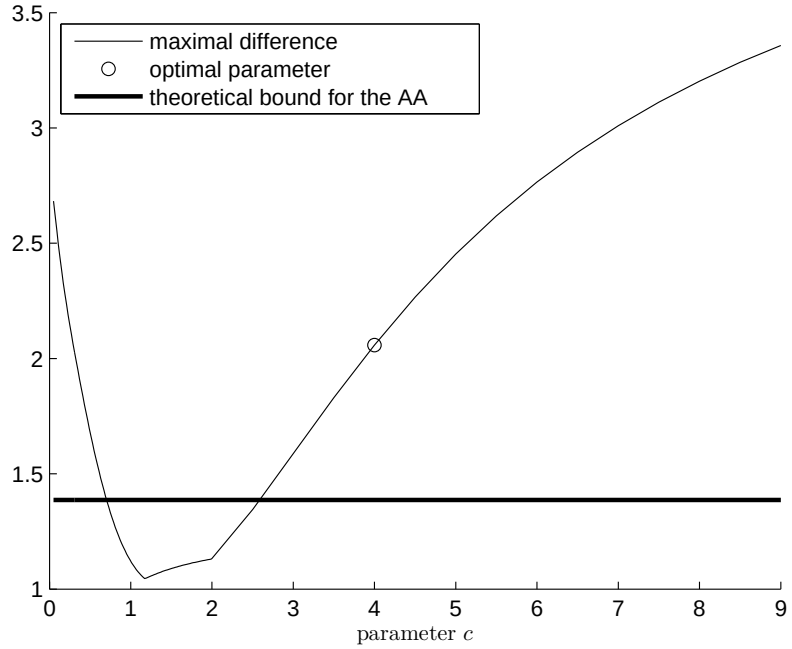


Figure 2.15: The maximal difference (2.52) for the WdAA as function of the parameter c on the tennis data. The theoretical bound for the WdAA is 5.5452 (see Table 2.5).

Table 2.4: The maximal difference between the loss of each algorithm in the selected set and the loss of the best expert for the football data (second column); the theoretical upper bound on this difference (third column).

Algorithm	Maximal difference	Theoretical bound
The AA	1.2318	2.0794
The WdAA ($c = 16/3$)	1.4076	11.0904
The WdAA ($c = 1$)	1.2255	none

the experts' weights using the information that the loss function is convex after exponentiation (see Lemma 2.5, for example). The WkAA assumes that the loss function is only convex (not necessarily exponentially) and thus computes the weights of the experts differently than the AA and the WdAA. The WkAA can use both the prediction approach of the WdAA (which is more computationally efficient), and the prediction approach similar to that of the

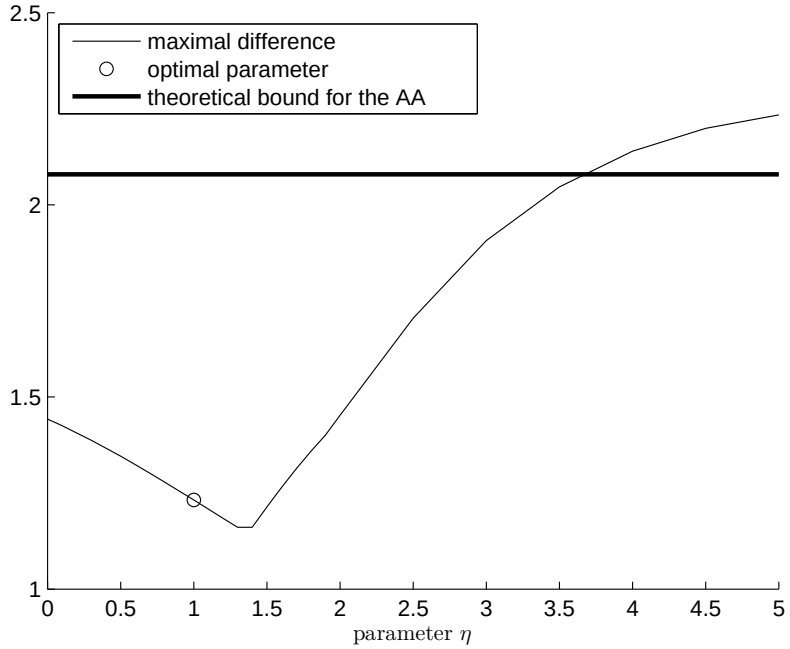


Figure 2.16: The maximal difference ((2.51) with η in place of c) for the AA as function of the parameter η on the football data.

Table 2.5: The maximal difference between the loss of each algorithm in the selected set and the loss of the best expert for the tennis data (second column); the theoretical upper bound on this difference (third column).

Algorithm	Maximal difference	Theoretical bound
The AA	1.1119	1.3863
The WdAA ($c = 4$)	2.0583	5.5452
The WdAA ($c = 1$)	1.1207	none

AA; but this appears less important than the way it computes the weights. The HA assumes even less: it does not even assume that its and the experts' performance is measured using a loss function. At each step the HA decides which expert it is going to follow, and at the end of the step it is only told the losses suffered by all experts.

We denote the parameters of the WkAA and the HA by c and β, respectively; the ranges of the parameters are $c \in (0, \infty)$ and $\beta \in [0, 1)$. The loss

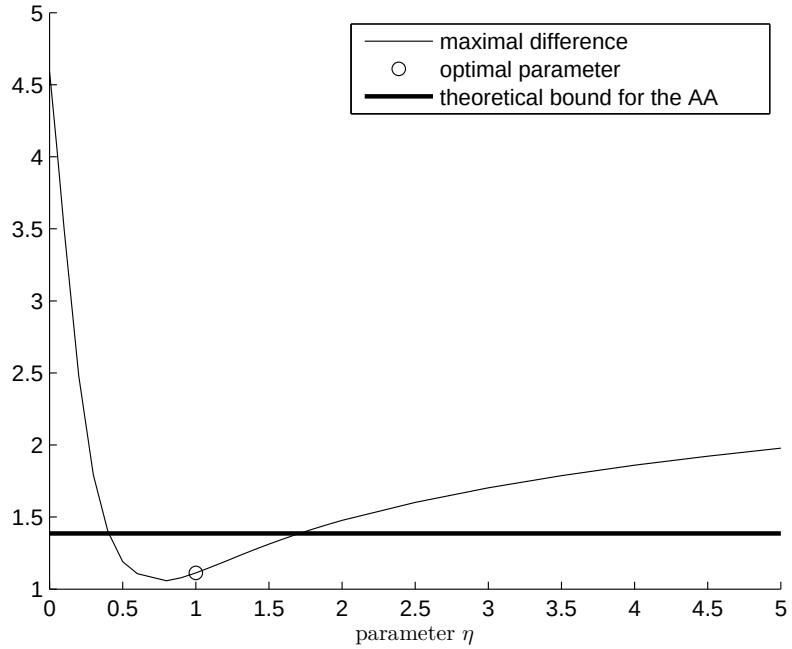


Figure 2.17: The maximal difference ((2.52) with η in place of c) for the AA as function of the parameter η on the tennis data.

bounds that we give below assume that the loss function takes values in the interval $[0, L]$, in the case of the WkAA, and that the losses are chosen from $[0, L]$, in the case of the HA, where L is a known constant. In the case of the Brier loss function, $L = 2$.

The loss of the WkAA (Kalnishkan and Vyugin, 2008, Corollary 14) is upper bounded as follows:

$$L_T \le \min_{\theta=1,\dots,K} L_T^\theta + 2L\sqrt{T \ln K}. \tag{2.53}$$

This bound is very different from (2.30) as the regret term $2L\sqrt{T \ln K}$ in (2.53) depends on T. This bound is guaranteed for $c = \sqrt{\ln K}/L$. For $c = \sqrt{8 \ln K}/L$, Cesa-Bianchi and Lugosi (2006, Theorem 2.3) prove the stronger bound

$$L_T \le \min_{\theta=1,\dots,K} L_T^\theta + L\sqrt{2T \ln K} + L\sqrt{\frac{\ln K}{8}}.$$

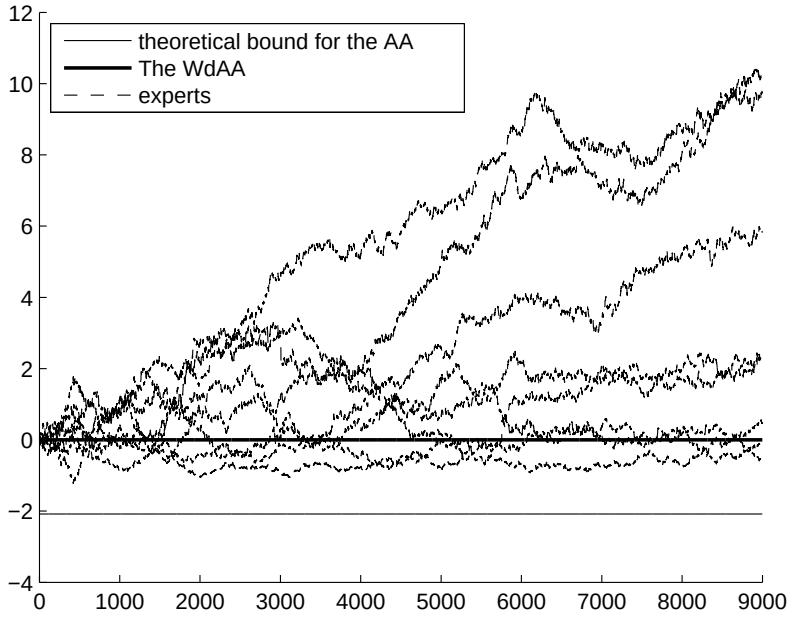


Figure 2.18: The difference between the cumulative loss of each of the 8 bookmakers and of the WdAA on the football data for $c = 1$ (the value of parameter minimizing the theoretical performance guarantee for the AA).

It is proven in Chernov and Zhdanov (2010) that WkAA achieves the regret $L\sqrt{T \ln K}$ with the parameter $c = \sqrt{4 \ln K}/L$.

The performance of the WkAA on our data sets is significantly worse than that of the WdAA with $c = 1$: the maximal difference (2.51)–(2.52) does not exceed $\ln K$ for all reasonable values of c in the case of football but only for a very narrow range of c (which is far from both Kalnishkan and Vyugin's $\sqrt{\ln K}/2$ and Cesa-Bianchi and Lugosi's $\sqrt{8 \ln K}/2$) in the case of tennis. Moreover, the WkAA violates the bound for the AA for all reasonable values of c on some natural subsets of the football data set: for example, when prediction starts from the second (2006/2007) season. Nothing similar happens for the WdAA with $c = 1$ on our data sets.

The loss bound for the HA (Freund and Schapire, 1997, Theorem 2) is as follows:

$$\mathrm{E}L_T \le \frac{\ln \frac{1}{\beta} \min_{\theta=1,\dots,K} L_T^\theta + L \ln K}{1 - \beta}, \tag{2.54}$$

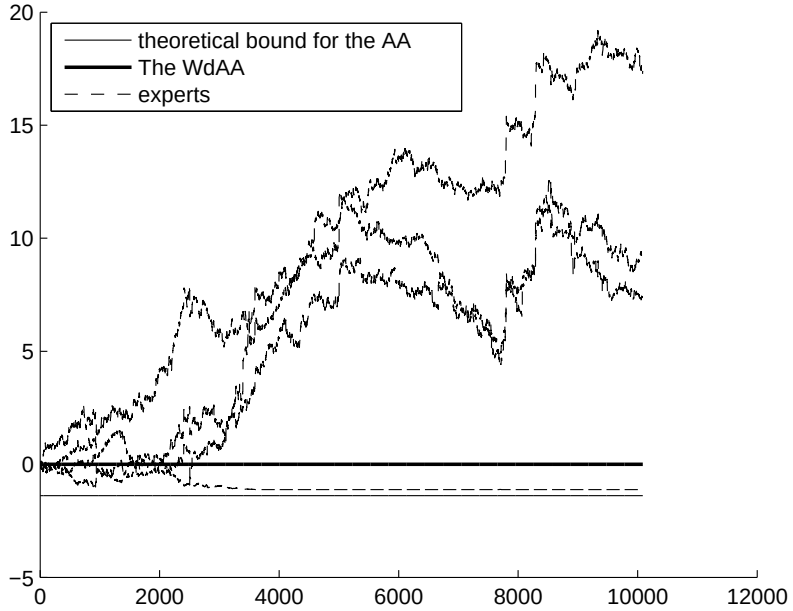


Figure 2.19: The difference between the cumulative loss of each of the 4 bookmakers and of the WdAA for $c = 1$ on the tennis data.

where $\mathrm{E}L_N$ stands for the learner's expected loss (the HA is a randomized algorithm). In the same framework, the Aggregating Algorithm attains the stronger bound

$$\mathrm{E}L_T \le \frac{\ln \frac{1}{\beta} \min_{\theta=1,\dots,K} L_T^\theta + L \ln K}{K \ln \frac{K}{K+\beta-1}} \tag{2.55}$$

(Vovk, 1998, Example 7). Of course, the AA applied to the HA framework (as described above, with no loss function) is very different from Algorithm 4, which is the AA applied to the Brier game; we refer to the former algorithm as AA-HA.

The losses suffered by the AA and the AA-HA on our data sets are close to each other and violate the optimal bound for the AA. It is interesting to note that the parameter β for the HA which minimizes its empirical loss is very close to zero (and equals zero for football). In this case the HA coincides with the Follow the Leader Algorithm (FLA). The FLA calculates the losses of the experts so far, and follows the best expert at the next step. If there are more than one best expert, its prediction is a simple average with equal

weights of the predictions of the best experts. It is known (Cesa-Bianchi and Lugosi, 2006, Section 4.3) that the FLA can fail on many natural sequences, unlike its variant Follow the Perturbed Learner.

The empirical performance of the FLA on the football data set is not so bad: it violates the loss bound for the AA only slightly; however, on the tennis data set the bound is violated badly.

Since the FLA performs well on the football data set, it is reasonable to check the performance of another similarly naive algorithm. The *Simple Average Algorithm*'s (SAA) prediction is defined as the arithmetic mean of the experts' predictions (with equal weights).

We have found that none of the naive algorithms perform consistently poorly, but they always fail badly on some natural parts of our data sets. The advantage of the more sophisticated algorithms having strong performance guarantees is that there is no danger of catastrophic performance on any data set.

2.7 Online prediction of ovarian cancer

In this section, we use the Aggregating Algorithm to diagnosing ovarian cancer using the level of the standard biomarker CA125 in conjunction with information provided by mass-spectrometry.

Early detection of ovarian cancer (OC) is important since clinical symptoms sometimes do not appear until the late stage of the disease. This leads to difficulties in treatment of the patient. Using the antigen CA125 significantly improves the quality of diagnosis. However, CA125 becomes less reliable at early stages and sometimes elevates too late to make use of it. Our goal is to investigate whether existing methods of online prediction can improve the quality of the detection of the disease and to demonstrate that the information contained in mass spectra is useful for ovarian cancer diagnosis in the early stages of the disease. The results of this section are described in Zhdanov et al. (2009a) and in more details in Zhdanov et al. (2009b).

2.7.1 Data set

We are working with a data set processed by Devetyarov et al. (2009) that was collected over the period of 7 years and has information (referred to as *samples*) about patients with the disease (referred to as *cases*) and patients who were healthy all this period, called *controls*. Description of the collection process is not our goal, so we do not state this question in detail. More detailed description of the data set and mass spectrometry peak extracting procedures can be found in Menon et al. (2005) and Devetyarov et al. (2009).

We consider prediction in *triplets*: each case sample is accompanied by two samples from healthy individuals, *matched controls*, which are chosen to be as close as possible to the case sample with respect to attributes such as age, storage conditions, and serum processing. There are 881 samples in total: 295 cases, 586 matched controls. There are up to 5 samples for each of the cases. Data for all samples contain the value of CA125, time to diagnosis (for cases), intensities of 67 mass-spectrometry peaks, and other. Time to diagnosis is the time interval measured in months between the date when the measurement was taken and the date when OC was diagnosed, or the date of operation. Peaks

are ordered by their frequency, or the percentage of samples having a non-aligned peak. We have 67 peaks of frequency more than 33%. For classification purposes we exclude cases with only one matched control, and cases with a lack of suitable information. As a result, we have 179 triplets containing 358 control samples and 179 case samples taken from 104 individuals. Each triplet is assigned a *time-to-diagnosis* defined as the time to diagnosis of the case sample in this triplet.

2.7.2 Experiments

In the given triplet of samples from different individuals we detect one sample which we predict as cancer. This framework was first described in Gammerman et al. (2008). The authors analyze an ovarian cancer data set and show that the information contained in mass-spectrometry peaks in conjunction with CA125 can help to provide more precise and reliable predictions of the samples from diseased patients than the CA125 criteria by itself several months before the moment of the diagnosis. We use the same framework and set of decision rules (CA125 combined with peak intensity) to derive an algorithm which performs better in a sense than any of these rules.

We combine the decision rules by using the Aggregating Algorithm and thus get our own prediction strategy. This section describes two experiments. The first one is a study of probability prediction of ovarian cancer. In the second one we analyse prediction at different stages before diagnosis and ensure that our results are not accidental by showing their statistical significance through calculating p-values.

Probability prediction of ovarian cancer

The aim of this experiment is to demonstrate how we give probability predictions for the samples in triplets and compare them to the predictions obtained using CA125 alone. The outcome of each event can be represented as a vector $(1, 0, 0)$, $(0, 1, 0)$, or $(0, 0, 1)$.

In order to estimate classification accuracy, we convert the predictions of all the algorithms into strict predictions by the *maximum rule*: we assign weight

1 to the labels with maximum predicted probability, weight 0 to the labels of other samples, and then normalize the assigned weights. The prediction of CA125 is represented as a vector with three components. This vector is obtained by applying the maximum rule to CA125 levels.

We refer to the *combination* of CA125 and peak intensity meaning the decision rule in the form

$$u(v, w, p) = v \ln C + w \ln I_p, \tag{2.56}$$

where C is the level of CA125, I_p is the intensity of the p-th peak, $p = 1, \ldots, 67$, $v \in \{0, 1\}$, and $w \in \{-2, -1, -1/2, 0, 1/2, 1, 2\}$. In our setting, each expert predicts according to one of the fixed combinations applying the maximum rule to their values for the samples in a triplet. The total number of different combinations, or experts, is 537: $402 = 6 \times 67$ for $v = 1, w \neq 0$, $134 = 2 \times 67$ for $v = 0$, and 1 for $v = 1$, $w = 0$. Devetyarov et al. (2009) show how such combinations can predict cancer well up to 15 months before diagnosis.

We apply Algorithm 4 to make predictions. For the purpose of simulating online prediction, we sort all the triplets by their time-to-diagnosis such that the earlier triplets go first. At each step we give the probability of being diseased for each sample in the triplet, numbers $p_1, p_2, p_3 \geq 0 : p_1 + p_2 + p_3 = 1$. We choose the uniform initial distribution over the experts and the theoretically optimal value for the parameter η, $\eta = 1$, for the Aggregating Algorithm (see Lemma 2.7). The evolution of the cumulative Brier loss of all the experts minus the cumulative loss of our algorithm over all the 179 triplets is presented in Figure 2.20. Clearly, the line for the AA is zero since we subtract its loss from itself. Experts having the line lower than zero are better than the AA, experts having the line higher than zero are worse. The x-axis presents triplets in the chronological order.

We can see from Figure 2.20 that the Aggregating Algorithm predicts better than the majority of the experts in our class after about 54 triplets, in particular better than CA125. At the end, the AA is better than all the experts. The group of lines clustered on the top of the graph separated from the main group are experts which do not include CA125. They make relatively

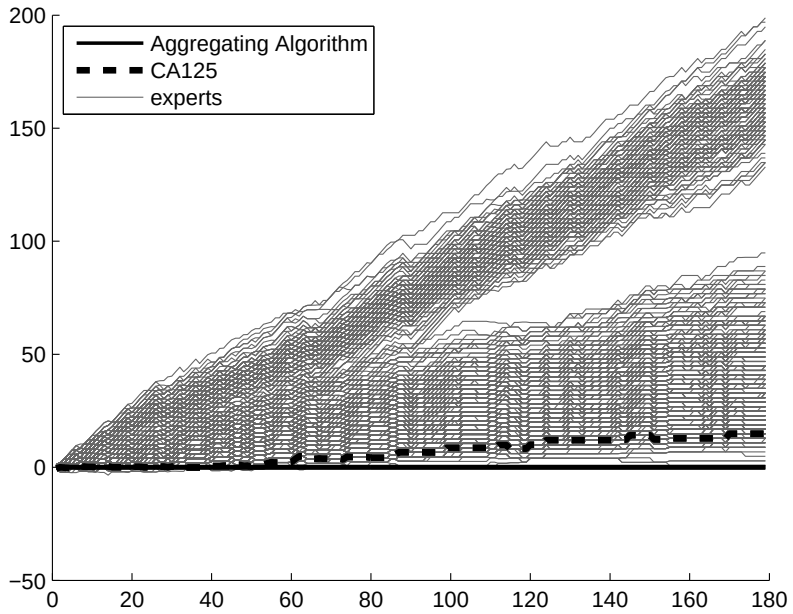


Figure 2.20: The difference between the cumulative loss of each of the experts of the AA over all the triplets.

many mistakes especially on late stages of the disease and accumulate large loss. These results show that the probability predictions of the AA are more precise than predictions of the experts interpreted as probability predictions. Moreover, we can be sure that the loss of the Aggregating Algorithm will never be much worse than the loss of the best expert since there is the bound (2.30) for it.

One can say this comparison is not fair because we allow the experts to give only strict predictions, and our algorithm is more flexible so its Brier loss is not so large. On the other hand, it is not trivial to find experts which make probability predictions, or convert the predictions of CA125 into probabilities of the disease for each sample in triplet, so this approach presents one of the ways to provide them.

In order to make a more strict comparison we allow the AA to make only strict predictions and use the maximum rule to convert its probability predictions into strict predictions. We will refer to this algorithm as to the *categorical AA*. If we calculate the Brier loss, we obtain Figure 2.21. We can see that the

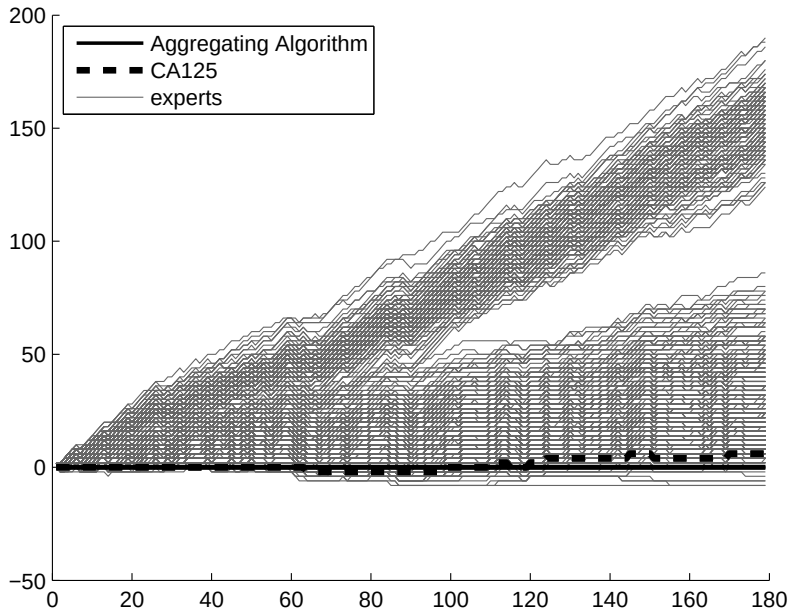


Figure 2.21: The difference between the cumulative loss of each of the experts and of the categorical AA over all the triplets.

categorical AA still beats CA125 at the end. The final performance is the performance on the whole data set. The loss of the categorical AA over the whole data set is more than the loss of only few predictors.

It may be useful to know specific combinations which perform well in this experiment. At the last step the best performance is achieved by the combinations

$$\ln C - \ln I_3, \quad \ln C - \frac{1}{2}\ln I_3, \quad \ln C - \ln I_2, \quad \ln C - \frac{1}{2}\ln I_7. \tag{2.57}$$

After them combinations with the peaks 50, 2, 7, 1, 34, 47 follow.

Prediction on different stages of the disease

Our second experiment is aimed to investigate whether it is possible to predict well at early stages of the disease. In this experiment we follow the approach proposed in Devetyarov et al. (2009). We consider 6-month time intervals with the starting points $t = 0, 1, \ldots, 16$ months before diagnosis. So if, for example,

$t = 1$, then samples with time-to-diagnosis from 1 to 7 are chosen. We will show below that our predictions are not reliable for the stages earlier than 15 months. For each time interval we select only the triplets within it; the latest for each case patient is taken if there are more than one. We denote the triplets in the interval t of length θ by $S_{t,\theta}$. We use $\theta = 6$.

In this experiment, we do not use a uniform initial weights distribution on the experts because the number of triplets within each time interval is too small for the algorithm to learn a lot. Instead, we assume that the importance of each peak decreases as its number increases in accordance with a power law, and that different combinations including the same peak have the same importance. This makes sense because the peaks are sorted by their frequency in the data set, so the peaks further down the list are less frequent and important for fewer people. Our specific weighing scheme is that the combinations with peak 1 have the initial weight $1 = d^0$, the combinations with peak 2 have the initial weight d^{-1}, etc. We empirically choose the coefficient for this distribution, $d = 1.2$, and the parameter η for the AA, $\eta = 0.65$, to achieve the reliability of prediction for all time intervals. The number of errors was calculated as half of Brier loss, which corresponds to counting errors in the case where all predictions are strict. Figure 2.22 shows the fractions of erroneous predictions made by different algorithms over different time periods. It presents the values for CA125, for the categorical Aggregating Algorithm, and for the best combination of the form (2.56) at each time period separately (assuming we know in advance which combination is the best for each period). It also includes the fractions of erroneous predictions for the three best combinations in (2.57) as peaks 2 and 3 were noticed to perform well.

This figure shows that the performance of the categorical AA is at least as good as the performance of CA125 on all stages before diagnosis. For the period 9–13 months the combination $\ln C - \ln I_3$ performs better than the AA, but on late stages 0–8 months it performs worse. Other combinations are even worse. Thus we can say that instead of choosing one particular combination, we should use the Aggregating Algorithm to mix all the combinations. This allows us to predict reliably on many stages of the disease.

The choice of the coefficients for the categorical AA requires us to check

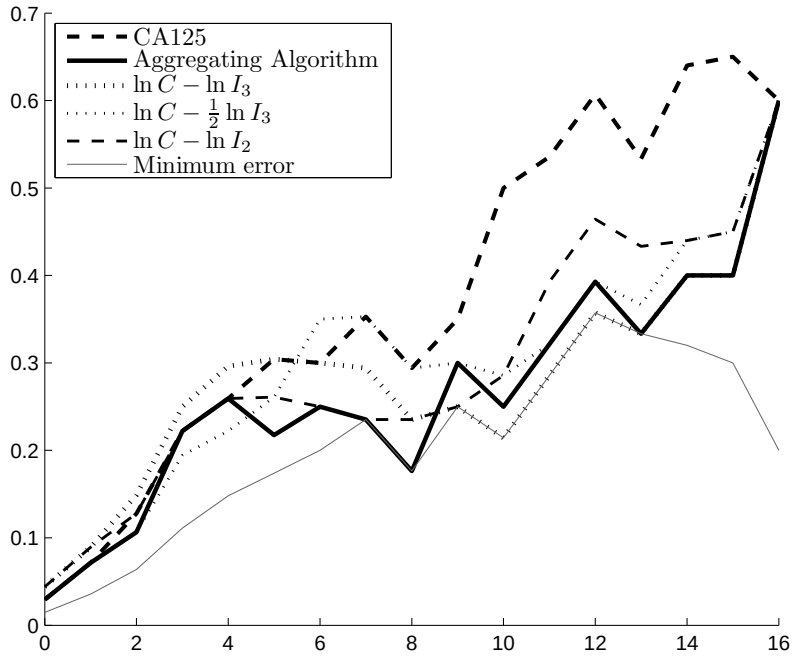


Figure 2.22: Fraction of erroneous predictions of different predictors over different time periods.

that our results are not accidental. Since the amount of data we have does not allow us to carry out reliable cross-validation procedure, we follow the approach for calculating p-values proposed in Gammerman et al. (2008). For each stage of the disease (time interval), we test the null hypothesis that the peak intensities and CA125 do not carry any information relevant for predicting labels. The test statistic is the performance of the categorical AA with the best parameters. Except for the earliest stages, we show that either this hypothesis is violated or some very unlikely event happened.

We calculate p-values for testing the null hypothesis. The *p-value* can be defined as a value taken by the random variable ξ satisfying

$$\forall \delta \text{ Probability}(\xi \leq \delta) \leq \delta$$

for all $\delta \in (0, 1)$ under the null hypothesis. To calculate p-values we choose the test statistic T described below, apply it to our data, and get the value

T_0. Then we calculate the probability of the event that $T \leq T_0$ under the null hypothesis.

Let τ be a triplet in $S_{t,6}$ and take the categorical AA with the parameter η and the initial weights distribution with parameter d. Let $\mathrm{err}(\tau, d, \eta)$ be its half Brier loss on the triplet τ. Then the half loss over each time interval $[t, t+6]$ is expressed as

$$\mathrm{Err}(S_{t,6}; d; \eta) = \sum_{\tau \in S_{t,6}} \mathrm{err}(\tau; d; \eta),$$

where $S_{t,6}$ is the set of triplets for the time interval $[t, t+6]$. We randomly reassign the labels in the triplets N times. For each t and each $j = 1, \ldots, N$ we calculate the minimum number of errors E made by the categorical AA by the rule

$$E = \min_{d \in D, \eta \in R} \mathrm{Err}(S_{t,6}, d, \eta).$$

Here $D = \{1.1, 1.2, \ldots, 2.0\}$ and $R = \{0.1, 0.15, 0.2, \ldots, 1.0\}$, so we consider different values for all parameters of the algorithm. By E_0 we denote the number of errors made by the categorical AA on the data with the correct label assignment. E is the loss of the best categorical AA on the data with permutations. This number is our test statistic. The p-value is calculated by the Monte-Carlo procedure stated as Algorithm 5.

Algorithm 5 Calculation of p-value

Require: Time to diagnosis t, number of trials $N = 10^4$.
  Calculate $E_0 := \min_{d \in D, \eta \in R} \mathrm{Err}(S_{t,6}, d, \eta)$.
  Initialize $Q := 0$.
  for $j = 1, \ldots, N$ **do**
    Assign a case label to a randomly chosen sample in each triplet in $S_{t,6}$.
    Calculate $E = \min_{d \in D, \eta \in R} \mathrm{Err}(S_{t,6}, d, \eta)$ for this data set.
    if $E \leq E_0$ **then**
      Update $Q = Q + 1$.
    end if
  end for
  Output $\frac{Q+1}{N+1}$ as a p-value.

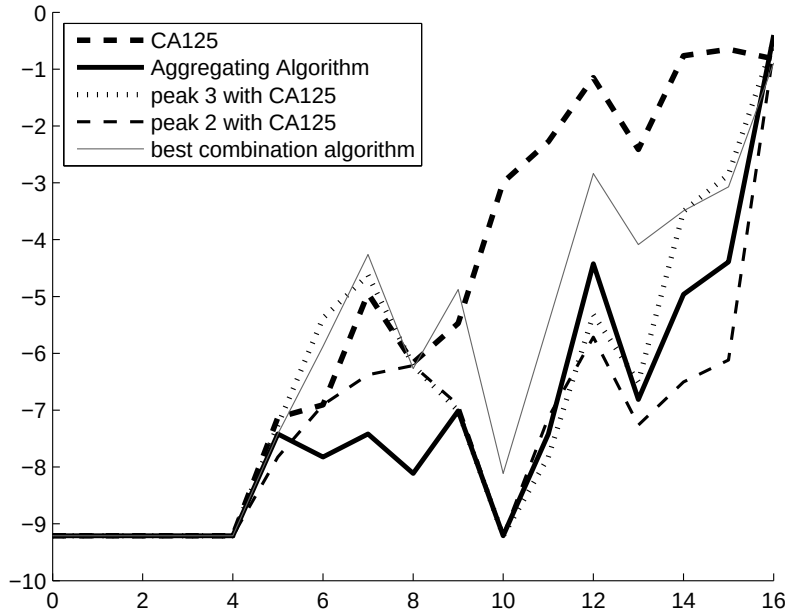


Figure 2.23: The logarithm of p-values for different algorithms.

Algorithm 5 calculates the ratio of times when the best categorical AA on a labels permutation performs better than the best categorical AA on the correct labels assignment. If the particular choice of the parameters d, η allowed the categorical AA to adjust well to any given data (overfit), this ratio would not be very small.

The logarithms of p-values for different algorithms are presented in Figure 2.23. It includes the values for the categorical AA. It also includes the values taken from Devetyarov et al. (2009) for the CA125 only. It includes p-values for an algorithm described in Devetyarov et al. (2009). This algorithm chooses the combination with the best performance and the least number of peak for each permutation of the labels. The figure also includes the p-values for the algorithms which choose the best combination with one particular peak, 2 or 3.

As we can see, our algorithm has small p-values, comparable with or even smaller than p-values for other algorithms. At the same time, our algorithm has fewer adjustments, because it does not choose even the peak but mixes all the peaks in the same manner. It also does not choose the best parameters

for every time interval but chooses them for all the time periods. The precise values for errors and p-values are presented in Table 2.6. Lower index $_e$ stands for the half loss for a given algorithm (AA stands for the categorical AA, CA stands for CA125), lower index $_p$ stands for the p-values for a given algorithm. The column Min_e shows the minimum number of errors made by one of the combinations, the p column shows the p-values for the method which chooses the best combination for each time period (see Devetyarov et al. (2009)), $C^3_{1,e}$ shows the number of errors for the combination $\ln C - \ln I_3$, $C^3_{2,e}$ shows the number of errors for the combination $\ln C - \frac{1}{2}\ln I_3$, C^2_e shows number of errors for the combination $\ln C - \ln I_2$. Columns 3_p and 2_p contain the p-values for the combinations with the peaks 3 and 2, respectively.

Table 2.6: Number of errors and p-values for different algorithms.

t	$\lvert S_{t,6}\rvert$	CA_e	CA_p	AA_e	AA_p	Min_e	p	$C^3_{1,e}$	$C^3_{2,e}$	3_p	C^2_e	2_p
0	68	2	0.0001	2	0.0001	1	0.0001	3	2	0.0001	3	0.0001
1	56	4	0.0001	4	0.0001	2	0.0001	5	4	0.0001	5	0.0001
2	47	6	0.0001	5	0.0001	3	0.0001	7	5	0.0001	6	0.0001
3	36	8	0.0001	8	0.0001	4	0.0001	9	7	0.0001	8	0.0001
4	27	7	0.0001	7	0.0001	4	0.0001	8	6	0.0001	7	0.0001
5	23	7	0.0008	5	0.0006	4	0.0006	7	6	0.0007	6	0.0004
6	20	6	0.0010	5	0.0004	4	0.0028	6	7	0.0046	5	0.0010
7	17	6	0.0071	4	0.0006	4	0.0141	5	6	0.0098	4	0.0017
8	17	5	0.0021	3	0.0003	3	0.0019	4	5	0.0020	4	0.0020
9	20	7	0.0042	6	0.0009	5	0.0076	5	6	0.0009	5	0.0010
10	28	14	0.0503	7	0.0001	6	0.0003	6	8	0.0001	8	0.0001
11	28	15	0.1028	9	0.0006	8	0.0042	8	9	0.0004	11	0.0008
12	28	17	0.3164	11	0.0120	10	0.0585	10	11	0.0049	13	0.0033
13	30	16	0.0895	10	0.0011	10	0.0168	10	11	0.0015	13	0.0007
14	25	16	0.4661	10	0.0070	8	0.0304	10	11	0.0301	11	0.0015
15	20	13	0.5211	8	0.0124	6	0.0464	8	9	0.0577	9	0.0022
16	10	6	0.4406	6	0.6708	2	0.4101	6	6	0.5979	6	0.5165

In practice, one often chooses a suitable significance level for their particular task. If we choose it at 5%, then we can see from the table that CA125 classification is significant up to 9 months in advance of diagnosis (the p-values are less than 5%). At the same time, the results for peaks combinations and for the AA are significant for up to 15 months.

2.7.3 Discussion

Our results show that the CA125 criterion, which is a current standard for the detection of ovarian cancer, can be outperformed, especially at early stages. Our approach gives more reliable predictions than the vast majority of particular combinations. It performs well on different stages of disease.

When testing the hypothesis that CA125 and peaks do not contain useful information for the prediction of the disease at its early stages, our algorithm gives better p-values in comparison to the algorithm which chooses the best combination; in addition, our algorithm requires fewer adjustments. This hypothesis can be rejected at the standard significance level 5% later than 16 months before diagnosis. Other approaches to probability prediction of ovarian cancer using CA125 criteria are based on the Risk of Ovarian Cancer algorithm (see Skates et al., 2003) and require multiple statistical assumptions about the data and a much larger size of data sets. Thus they can not be comparable in our setting.

The triplet setting looks a bit unrealistic. Our results can still be useful as a good demonstration that the choice of a particular prediction model may lead to worse predictions, and the mixture of the models may result in better and more reliable prediction. On the other hand, the online triplet setting can be used by itself: it is enough to choose a threshold level for the probability predictions of the AA which will determine whether in a given triplet of patients the prediction about one of them is certain. If any of the probabilities in a triplet are less than the threshold, additional analysis and application of other methods is required.

Chapter 3

Online regression in finite-dimensional spaces

The regression task is to predict a label of the given input vector. The label is usually allowed to take infinitely many values (for example, on real line).

In this chapter, we describe online linear (or generalized linear) regression framework. Experts predict according to linear functions of the input vectors given by reality at each step.

If the advice from a finite number of experts is given, the task of competing with all linear combinations of them can be reduced to linear regression. It is enough to interpret the vector of experts' predictions as the input vector for a regression algorithm.

In almost all the sections of this chapter, we consider the square (or Brier) loss. The Least Squares method was the earliest form of regression, and was first described by Gauss around 1794, and published by Legendre (1805) and Gauss (1809). Later it was noticed that in its simplest form Least Squares Regression may tend to data *overfitting* (good performance on the set of vectors where the coefficients of the method are found, but bad performance on many other sets) and large computational errors with certain types of data. One of the ways which was suggested to overcome these problems uses a Tikhonov regularization term in addition to the square error while searching for the best fit; the corresponding Ridge Regression method was published by Hoerl (1962).

The first work with linear regression in competitive online setting was per-

formed by Foster (1991), who considered a variant of Ridge Regression to compete with all linear functions with coefficients from the probability simplex.

3.1 Introduction to online regression

By online regression we usually understand the case when the benchmark class of experts is uncountable. Each expert is taken to follow certain prediction strategy, in other words to predict according to a function. In this case the notion of complexity of the functions is needed. The learner wishes to compete with the best expert which follows a strategy which is not too complex. The tradeoff between the complexity and the loss of the best expert is regulated by a parameter of the online regression algorithm chosen for the learner.

A popular field related to the competitive prediction is the online convex optimization introduced by Zinkevich (2003). Both fields cover a common special case: a compact set of experts under loss functions of a specific form (the square or logarithmic loss for our applications). In the general case, online convex optimization significantly relaxes the condition on loss functions, whereas competitive prediction removes the compactness requirement. Since many algorithms for competitive prediction are designed for a narrower class of loss functions, they have a better coefficient in the regret term.

3.1.1 Online regression framework

We follow the notation of the previous chapter: an outcome set is denoted by Ω, a prediction set is denoted by Γ, a loss function is denoted by $\lambda : \Omega \times \Gamma \mapsto [0, \infty]$, an index set for the experts supplied with a norm is denoted by Θ, and each individual expert is denoted by $\theta \in \Theta$. In online regression framework, reality gives a "hint", an *input vector* from a set $\mathbf{X}$, to the learner and the experts to help them in predicting the outcome. The prediction process follows Protocol 3.

Protocol 3 Online regression

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

Reality announces $x_t \in \mathbf{X}$.

Experts announce $\xi_t^\theta \in \Gamma$.

Learner announces $\gamma_t \in \Gamma$.

Reality announces $y_t \in \Omega$.

end for

For the compliance with the online regression literature we denote the outcomes by y, instead of ω as in the previous chapter. The experts and the learner record their cumulative losses: by L_t we denote the cumulative loss of the learner after the step t, and by L_t^θ we denote the cumulative loss of the expert θ after this step.

We will mostly be interested in deriving upper bounds on the loss of the learner in the form

$$L_T \le L_T^\theta + a\|\theta\|^2 + R_T \tag{3.1}$$

for any $\theta \in \Theta$, all $T = 1, 2, \ldots$, and a regret term $R_T = o(T)$. Here $a > 0$ is the regularization coefficient responsible for the tradeoff between the loss of the best expert and the complexity of his strategy.

3.2 Aggregating Algorithm for Regression

In this section we describe the Aggregating Algorithm for Regression (AAR) introduced by Vovk (2001). If the outcome set Ω is a bounded interval $[Y_1, Y_2]$, $Y_1 < Y_2$, the AAR is competitive with all the linear functions $\Theta = \mathbb{R}^n$ of the input vectors.

3.2.1 Derivation of the algorithm

Let the outcome set Ω be the bounded interval $[Y_1, Y_2]$, the prediction set Γ be the real line $\mathbb{R}$, and the loss function be the square loss (2.24): $\lambda(y, \gamma) = (\gamma - y)^2$ for $y \in \Omega$, $\gamma \in \Gamma$.

Let also the set $\mathbf{X}$ of input vectors be a subset of $\mathbb{R}^n$, $\mathbf{X} \subseteq \mathbb{R}^n$. A linear expert $\theta \in \mathbb{R}^n$ predicts ξ_t^θ at the step t:

$$\xi_t^\theta = c + \theta' x_t \tag{3.2}$$

for some $c \in \mathbb{R}$, we call c *the concentration point* of the experts. We apply the Aggregating Algorithm (Algorithm 1) to compete with these experts. Vovk (2001) first applied it for online linear regression with $Y_1 = -Y$, $Y_2 = Y$, $Y > 0$, and symmetric experts with $c = 0$.

The unnormalized weight of the expert θ after the step T is represented by

$$P_T(d\theta) = e^{-\eta(\xi_T^\theta - y_T)^2} P_{T-1}(d\theta) = e^{-\eta \sum_{t=1}^{T} (\xi_t^\theta - y_t)^2} P_0(d\theta).$$

We set the initial weights distribution P_0 over the set $\Theta = \mathbb{R}^n$ of the experts to have the Gaussian density with a parameter $a > 0$:

$$P_0(d\theta) = \left(\frac{a\eta}{\pi}\right)^{n/2} e^{-a\eta\|\theta\|^2} d\theta. \tag{3.3}$$

Then the generalized prediction (2.3) at the step T is represented as follows:

$$g_T(y) = -\frac{1}{\eta}\ln\left(\frac{1}{K}\int_\Theta e^{-\eta\left((\xi_T^\theta - y)^2 + \sum_{t=1}^{T-1}(\xi_t^\theta - y_t)^2 + a\|\theta\|^2\right)}d\theta\right)$$
$$= -\frac{1}{\eta}\ln\left(\frac{1}{K}\int_\Theta e^{-\eta\left(\theta' A_T\theta - 2\theta' b_{T-1} - 2\theta' x_T(y-c) + \sum_{t=1}^{T-1}(y_t-c)^2 + (y-c)^2\right)}d\theta\right),$$

where $A_T = aI + \sum_{t=1}^{T} x_t x_t'$ is a symmetrical positive definite matrix $n\times n$, I is the unit $n\times n$ matrix, and $b_{T-1} = \sum_{t=1}^{T-1} x_t(y_t - c)$ is a column vector $n\times 1$. Here $K = \int_\Theta e^{-\eta\sum_{t=1}^{T-1}(\xi_t^\theta - y_t)^2 - a\eta\|\theta\|^2}d\theta$ is the normalizing constant for the weights distribution.

Substitution function for the game with infinite number of possible outcomes

We show that the substitution function (2.26) can be used for the case of infinite number of possible outcomes, $\Omega = [Y_1, Y_2]$. The following lemma is an elaboration of the result in Haussler et al. (1998) that shows that any substitution function in the game with two possible outcomes $\{Y_1, Y_2\}$ is also a substitution function in the game with $\Omega = [Y_1, Y_2]$.

Lemma 3.1 (Vovk, 2001, Lemma 3) *Fix Y_1 and Y_2 such that $Y_1 < Y_2$. Let y and γ range over $[Y_1, Y_2]$ and $\mathbb{R}$ respectively, and $\lambda(y,\gamma) = (\gamma - y)^2$ be the square loss function. Let P be a probability distribution in $\mathbb{R}$, and let g be the following mixture:*

$$g(y) = \log_\beta \int \beta^{(\gamma-y)^2} P(d\gamma), \quad \beta \in [0,1).$$

For every $\gamma \in \mathbb{R}$, if

$$\lambda(Y_1,\gamma) \le g(Y_1) \text{ and } \lambda(Y_2,\gamma) \le g(Y_2),$$

then

$$\lambda(y,\gamma) \le g(y), \quad \forall y \in [Y_1, Y_2].$$

The prediction algorithm

Recall that we take $\beta = e^{-\eta}$. Due to Lemma 3.1 we can use the substitution function (2.26):

$$\begin{aligned}
\gamma_T &= \frac{Y_2 + Y_1}{2} - \frac{g_T(Y_2) - g_T(Y_1)}{2(Y_2 - Y_1)} \\
&= \frac{Y_2 + Y_1}{2} - \frac{1}{2(Y_2 - Y_1)\eta} \ln \frac{\int_\Theta e^{-\eta\theta' A_T \theta + 2\eta\theta'(b_{T-1} + (Y_1 - c)x_T) - \eta\left(W + (Y_1 - c)^2\right)} d\theta}{\int_\Theta e^{-\eta\theta' A_T \theta + 2\eta\theta'(b_{T-1} + (Y_2 - c)x_T) - \eta(W + (Y_2 - c)^2)} d\theta} \\
&= \frac{Y_2 + Y_1}{2} - \frac{1}{2(Y_2 - Y_1)} \ln e^{Y_2^2 - Y_1^2 + 2c(Y_1 - Y_2) - \left(b_{T-1} + \left(\frac{Y_2 + Y_1}{2} - c\right)x_T\right)' A_T^{-1} \left(\frac{Y_2 - Y_1}{2} x_T\right)} \\
&= c + \left(b_{T-1} + \left(\frac{Y_2 + Y_1}{2} - c\right) x_T\right)' A_T^{-1} x_T, \qquad (3.4)
\end{aligned}$$

where the third equality follows from Lemma A.3. Here $W = \sum_{t=1}^{T-1} (y_t - c)^2$. Thereby we obtain Algorithm 6.

Algorithm 6 Aggregating Algorithm for Regression

Require: $a > 0$.

Initialize $b_0 = 0 \in \mathbb{R}^n$, $A_0 = aI \in \mathbb{R}^{n \times n}$.

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

Read $x_t \in \mathbb{R}^n$.

Update $A_t = A_{t-1} + x_t x_t'$.

Predict $\gamma_t = c + \left(b_{t-1} + \left(\frac{Y_2 + Y_1}{2} - c\right) x_t\right)' A_{t-1}^{-1} x_t$.

Read y_t.

Update $b_t = b_{t-1} + (y_t - c)x_t$.

end for

The incremental update of the matrix A_t^{-1} can be done efficiently by the Sherman-Morrison formula (see, e.g., Press et al., 1992, p.73), which requires $O(n^2)$ operations per step:

$$A_t^{-1} = A_{t-1}^{-1} - \frac{A_{t-1}^{-1} x_t x_t' A_{t-1}^{-1}}{1 + x_t' A_{t-1}^{-1} x_t}.$$

It is easy to check that Algorithm 6 minimizes

$$a\|\theta\|^2 + \left(\theta' x_T - \left(\frac{Y_2 + Y_1}{2} - c\right)\right)^2 + \sum_{t=1}^{T-1} (\theta' x_t - (y_t - c))^2 \tag{3.5}$$

in $\theta \in \Theta$. Indeed, by taking the derivative in θ of the quadratic form and finding $\widehat{\theta}$ where it achieves zero, we can obtain that the AAR predicts $c + \widehat{\theta}' x_T$.

3.2.2 Performance guarantee

To derive an upper bound on the cumulative loss of Algorithm 6 we use the following lemma. It states an upper bound on the loss of the AA (see Section 2.2.1). It generalizes the proof technique firstly suggested in the proof of Theorem 1 (Vovk, 2001), and will be used to handle several problems of online regression. It does not require that the experts follow linear functions of the input vectors and the loss function is the square loss.

Lemma 3.2 *Let $\Theta = \mathbb{R}^n$ be the set indexing experts. Take the initial distribution $P_0(d\theta)$ over the experts to be the Gaussian distribution* (3.3)*. Assume that the loss function $\lambda : \Omega \times \Gamma \to [0, \infty]$ is η-mixable, and denote by θ_0 some element of $\arg\min_{\theta \in \Theta} \left(L_T^\theta + a\|\theta\|^2\right)$. If, for any $T = 1, 2, \ldots$, there exists a symmetric positive definite matrix A_T such that*

$$L_T^\theta + a\|\theta\|^2 \le L_T^{\theta_0} + a\|\theta_0\|^2 + (\theta - \theta_0)' A_T (\theta - \theta_0), \tag{3.6}$$

then

$$L_T(\mathrm{AA}) \le L_T^{\theta_0} + a\|\theta_0\|^2 + \frac{1}{2\eta} \ln\det\left(\frac{1}{a} A_T\right). \tag{3.7}$$

If the Bayesian Algorithm is used, and (3.6) *holds as an equality, then* (3.7) *holds as an equality with $\eta = 1$.*

Proof By Lemma 2.1 we have

$$\begin{aligned} L_T(\mathrm{APA}) = \log_\beta \int_\Theta \beta^{L_T^\theta} P_0(d\theta) &= \log_\beta \left((a\eta/\pi)^{n/2} \int_\Theta \beta^{L_T^\theta + a\|\theta\|^2} d\theta\right) \\ &\le \log_\beta \left((a\eta/\pi)^{n/2} \beta^{L_T^{\theta_0} + a\|\theta_0\|^2} \int_\Theta \beta^{\tilde{\theta}' A_T \tilde{\theta}} d\tilde{\theta}\right), \end{aligned}$$

where $\tilde{\theta} = \theta - \theta_0$. The inequality follows from (3.6) and the fact that the functions β^z and $\log_\beta z$ are decreasing in $z > 0$, with $\beta \in [0, 1)$. The integration over $\tilde{\theta}$ is equivalent to the integration over θ. The last integral can be analytically evaluated using Lemma A.1:

$$\int_\Theta \beta^{\tilde{\theta}' A_T \tilde{\theta}} d\tilde{\theta} = \int_\Theta e^{-\eta \tilde{\theta}' A_T \tilde{\theta}} d\tilde{\theta} = \frac{\pi^{n/2}}{\sqrt{\det(\eta A_T)}}.$$

Substituting this to the upper bound for $L_T(\text{APA})$ and using the fact that $L_T(\text{AA}) \le L_T(\text{APA})$ for the mixable game concludes the proof of (3.7).

For the Bayesian algorithm $L_T(\text{BA}) = L_T(\text{APA})$ and thus if (3.6) is an equality, then there is equality in (3.7). ■

This lemma states that if the regularized cumulative loss of each expert can be bounded above by a quadratic form with its minimum in the best expert, then the loss of the AA can be bounded above by the regularized cumulative loss of the best expert plus a simple regret term. This lemma leads to the following upper bound on the cumulative square loss of the AAR.

Theorem 3.1 *For any $a > 0$, every positive integer T, every sequence of outcomes of the length T, and any $\theta \in \mathbb{R}^n$, the cumulative loss L_T of the AAR with the parameter a satisfies*

$$L_T \le L_T^\theta + a\|\theta\|^2 + \frac{(Y_2 - Y_1)^2}{4} \ln\det\left(I + \frac{1}{a}\sum_{t=1}^{T} x_t x_t'\right). \tag{3.8}$$

If, in addition, $\|x_t\|_\infty \le X$ for all t, then

$$L_T \le L_T^\theta + a\|\theta\|^2 + n\frac{(Y_2 - Y_1)^2}{4} \ln\left(1 + \frac{TX^2}{a}\right). \tag{3.9}$$

PROOF By θ_0 we denote the best expert: $\theta_0 = \arg\min_{\theta\in\Theta}\left(L_T^\theta + a\|\theta\|^2\right)$. The regularized cumulative loss of any expert θ at any step T can be expressed as

follows:

$$L_T^\theta + a\|\theta\|^2 = \sum_{t=1}^{T}(\theta' x_t - (y_t - c))^2 + a\|\theta\|^2$$
$$= \sum_{t=1}^{T}(\theta_0' x_t - (y_t - c))^2 + a\|\theta_0\|^2 + (\theta - \theta_0)' A_T (\theta - \theta_0)$$

for $A_T = aI + \sum_{t=1}^{T} x_t x_t'$. By Lemma 3.2 we obtain for any $\theta \in \Theta$

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

We take the maximum value for η, $\eta = \frac{2}{(Y_2 - Y_1)^2}$ (recall Lemma 2.5). The determinant of a symmetric positive definite matrix is upper bounded by the product of its diagonal elements (see Beckenbach and Bellman, 1961, Chapter 2, Theorem 7): $\det \left(I + \frac{1}{a} \sum_{t=1}^{T} x_t x_t' \right) \leq \left(1 + \frac{TX^2}{a} \right)^n$. This concludes the proof. ■

Interestingly, the upper bound for the AAR depends only on the size of the prediction interval but not on the location of it. It also does not depend on the concentration point of the experts.

3.3 Competing with Gaussian linear experts

In this section, we consider the problem of competing with Gaussian linear experts: experts whose predictions are the densities of normal distributions on the set of outcomes. From the statistical point of view, they predict according to the model $y = \theta' x + \epsilon$ with Gaussian noise $\epsilon \sim N(0, \sigma^2)$. Therefore, these are linear experts which additionally incorporate the information about the Gaussian nature of the noise into their predictions.

We show that Bayesian Ridge Regression can be thought of as an online algorithm competing with all the Gaussian linear experts under the logarithmic loss function. We use this representation of Bayesian Ridge Regression to derive theoretical guarantees on the square loss of Ridge Regression competing with linear experts. Our main theoretical guarantees have the form of equalities. This leads us to an upper bound on the square loss of Ridge Regression. The results of this section are described in Zhdanov and Vovk (2009).

Most of previous research in online prediction considers experts that disregard the presence of noise in observations. Some bounds for Bayesian Ridge Regression were previously derived by Kakade and Ng (2004). They prove upper bounds (perhaps because a more general problem is considered) which usually have the same logarithmic order of the regret term as the order of the bounds which we derive from our equalities for Bayesian Ridge Regression. See also Banerjee (2006) and Freund et al. (1997) for upper bounds on the loss of general Bayesian algorithms. Our approach allows us to partly improve these results for our problem.

3.3.1 Bayesian Ridge Regression as a competitive algorithm

Let the outcome set Ω be the real line $\mathbb{R}$, the prediction set Γ be the set of all measurable functions on the real line that are integrable to one, and the index set Θ for the experts be $\mathbb{R}^n$. The loss function λ is the logarithmic loss (2.20):

$$\lambda(y, \gamma) = -\ln \gamma(y),$$

where $\gamma \in \Gamma$ and $y \in \Omega$. The game follows Protocol 3 of online regression.

We consider experts whose predictions at step t are the densities of the normal distributions $N(\theta' x_t, \sigma^2)$ on the set of outcomes for some known fixed variance $\sigma^2 > 0$ (so each expert θ follows a fixed strategy, as everywhere in this chapter). In other words, each expert $\theta \in \Theta$ predicts

$$\xi_t^\theta(y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(\theta' x_t - y)^2}{2\sigma^2}}. \tag{3.10}$$

We denote the cumulative logarithmic loss of the expert θ at the step T by L_T^θ and the cumulative logarithmic loss of the learner at the step T by L_T. We use the Bayesian Algorithm (Algorithm 3) to mix these experts and obtain a competitive algorithm.

We take the initial distribution $N\left(0, \frac{\sigma^2}{a} I\right)$ on the experts with some $a > 0$:

$$P_0(d\theta) = \left(\frac{a}{2\sigma^2\pi}\right)^{n/2} \exp\left(-\frac{a}{2\sigma^2}\|\theta\|^2\right) d\theta.$$

We will prove that in this setting the prediction of the Bayesian Algorithm is equal to the prediction of Bayesian Ridge Regression. The prediction γ_T of the Bayesian Algorithm is expressed as follows:

$$\gamma_T(y) = \int_\Theta \xi_T^\theta(y) P_{T-1}^*(d\theta). \tag{3.11}$$

Here the normalized weights distribution $P_{T-1}^*(d\theta)$ is obtained after the normalization of the unnormalized weights (2.21): $P_{T-1}(d\theta) = P_0(d\theta) \prod_{t=1}^{T-1} \xi_t^\theta(y_t)$.

First we need to introduce some notation. For $t = 1, 2, \ldots$, let X_t be the $t \times n$ matrix of row vectors $x_1', \ldots, x_t'$, Y_t be the column vector of outcomes $y_1, \ldots, y_t$, and $A_t = X_t' X_t + aI$. The Bayesian Ridge Regression algorithm predicts at each step T the normal distribution $N(\gamma_T, \sigma_T^2)$ with the mean and variance given by

$$\gamma_T = Y_{T-1}' X_{T-1} A_{T-1}^{-1} x_T, \quad \sigma_T^2 = \sigma^2 x_T' A_{T-1}^{-1} x_T + \sigma^2 \tag{3.12}$$

for some $a > 0$ and the known noise variance σ^2.

Lemma 3.3 *In our setting the prediction* (3.11) *of the Bayesian Algorithm is the prediction density of Bayesian Ridge Regression in the notation of* (3.12)*:*

$$\gamma_T(y) = \frac{1}{\sqrt{2\pi\sigma_T^2}} e^{-\frac{(\gamma_T - y)^2}{2\sigma_T^2}}. \tag{3.13}$$

PROOF The prediction

$$\gamma_T(y) = \int_\Theta \xi_T^\theta(y) P_{T-1}^*(d\theta) = \frac{\int_{\mathbb{R}^n} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(\theta' x_T - y)^2}{2\sigma^2}} \prod_{t=1}^{T-1} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(\theta' x_t - y_t)^2}{2\sigma^2}} P_0(d\theta)}{\int_{\mathbb{R}^n} \prod_{t=1}^{T-1} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(\theta' x_t - y_t)^2}{2\sigma^2}} P_0(d\theta)}$$

is formally equal to the density of the predictive distribution of the Bayesian Gaussian linear model, and so equality (3.13) is true (see Bishop, 2006, Section 3.3.2). ∎

Remark 3.1 From the probabilistic point of view Lemma 3.3 is usually explained in the following way (Hoerl and Kennard, 2000). The posterior distribution $P_{T-1}^*(\theta)$ is $N(A_{T-1}^{-1} X'_{T-1} Y_{T-1}, \sigma^2 A_{T-1}^{-1})$. The conditional distribution of $\theta' x_T$ given the training examples is then $N(Y'_{T-1} X_{T-1} A_{T-1}^{-1} x_T, \sigma^2 x'_T A_{T-1}^{-1} x_T)$, and so the predictive distribution is $N(Y'_{T-1} X_{T-1} A_{T-1}^{-1} x_T, \sigma^2 x'_T A_{T-1}^{-1} x_T + \sigma^2)$.

To derive the theoretical guarantee on the logarithmic loss of Bayesian Ridge Regression we utilize the fact the loss of the APA (2.5) can be transformed to the regularized cumulative loss of the best expert θ and a regret term (following Lemma 3.2).

Theorem 3.2 *For any sequence $x_1, y_1, x_2, y_2, \ldots$, the cumulative logarithmic loss L_T of the Bayesian Ridge Regression algorithm predicting* (3.13) *at any step T can be expressed as*

$$L_T = \min_\theta \left(L_T^\theta + \frac{a}{2\sigma^2} \|\theta\|^2 \right) + \frac{1}{2} \ln \det \left(I + \frac{1}{a} \sum_{t=1}^{T} x_t x_t' \right). \tag{3.14}$$

If $\|x_t\|_\infty \le X$ for any $t = 1, 2, \ldots$, then

$$L_T \le \min_\theta \left(L_T^\theta + \frac{a}{2\sigma^2} \|\theta\|^2 \right) + \frac{n}{2} \ln \left(1 + \frac{TX^2}{a} \right). \tag{3.15}$$

PROOF By θ_0 we denote the best expert: $\theta_0 = \arg\min_{\theta\in\Theta}\left(L_T^\theta + \frac{a}{2\sigma^2}\|\theta\|^2\right)$. The regularized cumulative loss of any expert θ at any step T can be expressed as

$$\begin{aligned} L_T^\theta + \frac{a}{2\sigma^2}\|\theta\|^2 &= \sum_{t=1}^{T} -\ln\frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{(\theta' x_t - y)^2}{2\sigma^2}} + \frac{a}{2\sigma^2}\|\theta\|^2 \\ &= \sum_{t=1}^{T} -\ln\frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{(\theta_0' x_T - y)^2}{2\sigma^2}} + \frac{a}{2\sigma^2}\|\theta_0\|^2 + (\theta-\theta_0)' A_T(\theta-\theta_0) \end{aligned}$$

for $A_T = \frac{1}{2\sigma^2}\left(aI + \sum_{t=1}^{T} x_t x_t'\right)$. By the second part of Lemma 3.2 we obtain for any $\theta \in \Theta$

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

As we already noticed, the determinant of a symmetric positive definite matrix is upper bounded by the product of its diagonal elements (see Beckenbach and Bellman, 1961, Chapter 2, Theorem 7): $\det\left(I + \frac{1}{a}\sum_{t=1}^{T} x_t x_t'\right) \le \left(1 + \frac{TX^2}{a}\right)^n$. This concludes the proof. ■

This theorem shows that the Bayesian Ridge Regression algorithm can be thought of as an online algorithm successfully competing with all the Gaussian linear models under the logarithmic loss function. A bound similar to (3.15) on the logarithmic loss of Bayesian Ridge Regression is proven by Kakade and Ng (2004) using a different method; the only difference in the bound is that the authors assume that the 2-norm of x_t is bounded by 1.

3.3.2 Ridge Regression as a competitive algorithm

In this section we let the outcome set Ω and the prediction set Γ be the real line $\mathbb{R}$, and the index set Θ for the experts be $\mathbb{R}^n$. The loss function λ is the square loss (2.24):

$$\lambda(y,\gamma) = (\gamma - y)^2,$$

where $\gamma \in \Gamma$ and $y \in \Omega$. The game follows Protocol 3. The linear experts predict (3.2) with $c = 0$.

We use the Ridge Regression algorithm (RR) for the learner:

Algorithm 7 Online Ridge Regression

Require: $a > 0$.

Initialize $b_0 = 0 \in \mathbb{R}^n$, $A_0 = aI \in \mathbb{R}^{n \times n}$.

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

Read $x_t \in \mathbb{R}^n$.

Predict $\gamma_t = b'_{t-1} A_{t-1}^{-1} x_t$.

Read y_t.

Update $A_t = A_{t-1} + x_t x'_t$.

Update $b_t = b_{t-1} + y_t x_t$.

end for

As we can see, the difference between the RR and the AAR is that the AAR updates the matrix A_t before making its prediction, whereas the RR does it after making its prediction.

Following this algorithm the learner's prediction at step T can be written as

$$\gamma_T = \left(\sum_{t=1}^{T-1} y_t x_t \right)' \left(aI + \sum_{t=1}^{T-1} x_t x'_t \right)^{-1} x_T.$$

It is easy to check that Algorithm 7 minimizes

$$a\|\theta\|^2 + \sum_{t=1}^{T-1} (\theta' x_t - y_t)^2 \tag{3.16}$$

in $\theta \in \Theta$. Indeed, by taking the derivative in θ of the quadratic form and finding $\widehat{\theta}$ where it achieves zero, we can obtain that the RR predicts $\widehat{\theta}' x_T$.

We prove the following theoretical guarantee for the square loss of the learner following Ridge Regression. It states an equality rather than inequality without even assuming that the outcomes are bounded. Since it is an equality, it unites upper and lower bounds on the loss. It appears that all natural bounds on the square loss of Ridge Regression can be deduced from our theorem; we

give some examples below. In the case when the input vectors and outcomes are not restricted in any way, like for our Theorem 3.3, it is possible to prove certain loss bounds for the Gradient Descent (see Cesa-Bianchi et al., 1996). The bounds are proved on the cumulative loss of a Gradient Descent based algorithm competing with linear experts: the algorithm predicting γ_t^{GD} at step t achieves the upper bound

$$\sum_{t=1}^{T}(\gamma_t^{GD} - y_t)^2 \le 9 \min_{\theta\in\mathbb{R}^n}\left(\sum_{t=1}^{T}(\theta' x_t - y_t)^2 + \max_{t=1,\dots,T}\|x_t\|_2^2\|\theta\|^2\right). \tag{3.17}$$

We will derive a bound of this type from our main theorem below.

Theorem 3.3 *The Ridge Regression algorithm for the learner with $a > 0$ satisfies, at any step T,*

$$\sum_{t=1}^{T}\frac{(\gamma_t - y_t)^2}{1 + x_t' A_{t-1}^{-1} x_t} = \min_{\theta\in\mathbb{R}^n}\left(\sum_{t=1}^{T}(\theta' x_t - y_t)^2 + a\|\theta\|^2\right). \tag{3.18}$$

PROOF Let us rewrite the cumulative logarithmic losses L_T and L_T^θ in (3.14) using the expression for σ_t^2 given by (3.12) and (3.10):

$$\begin{aligned}
L_T &= -\sum_{t=1}^{T}\ln\left(\frac{1}{\sqrt{2\pi\sigma_t^2}}e^{-\frac{(\gamma_t - y)^2}{2\sigma_t^2}}\right) \\
&= \frac{1}{2}\ln\left((2\pi\sigma^2)^T\prod_{t=1}^{T}(1 + x_t' A_{t-1}^{-1} x_t)\right) + \frac{1}{2\sigma^2}\sum_{t=1}^{T}\frac{(\gamma_t - y_t)^2}{1 + x_t' A_{t-1}^{-1} x_t}, \\
L_T^\theta &= -\ln\left(\frac{1}{(2\pi\sigma^2)^{T/2}}e^{-\frac{1}{2\sigma^2}\sum_{t=1}^{T}(\theta' x_t - y_t)^2}\right) \\
&= \frac{T}{2}\ln(2\pi\sigma^2) + \frac{1}{2\sigma^2}\sum_{t=1}^{T}(\theta' x_t - y_t)^2.
\end{aligned}$$

Substituting these expression into (3.14), we have:

$$\frac{1}{2}\ln\prod_{t=1}^{T}(1+x_t'A_{t-1}^{-1}x_t)+\frac{1}{2\sigma^2}\sum_{t=1}^{T}\frac{(\gamma_t-y_t)^2}{1+x_t'A_{t-1}^{-1}x_t}$$
$$=\frac{1}{2\sigma^2}\min_{\theta}\left(\sum_{t=1}^{T}(\theta'x_t-y_t)^2+a\|\theta\|^2\right)+\frac{1}{2}\ln\det\left(I+\frac{1}{a}\sum_{t=1}^{T}x_tx_t'\right).$$

Equation (3.18) follows from Lemma A.6. Note that σ^2 cancelled out; this is natural as Ridge Regression (unlike Bayesian Ridge Regression) does not depend on σ. ■

An equivalent equality is also obtained (but well hidden) in the proof of Theorem 4.6 in Azoury and Warmuth (2001). Our proof is clearer and emphasizes the role of the equality. The power of the equality for Ridge Regression can be best appreciated by looking at the range of its implications, both known and new. For example, Corollary 3.1 answers the question asked by several researchers (for example, Vovk, 2001) whether Ridge Regression has a relative loss bound with the regret term of the order $\ln T$ under the square loss function, where T is the number of steps and the outcomes are assumed bounded.

Note that the part $x_t'A_{t-1}^{-1}x_t$ in the denominator is usually close to zero for large t. We obtain an upper bound in the form which is more familiar from online prediction literature.

Corollary 3.1 *Assume $|y_t|\le Y$ for all t, clip the predictions of Ridge Regression to $[-Y,Y]$, and denote them by γ_t^Y. Then*

$$\sum_{t=1}^{T}(\gamma_t^Y-y_t)^2\le\min_{\theta}\left(\sum_{t=1}^{T}(\theta'x_t-y_t)^2+a\|\theta\|^2\right)+4Y^2\ln\det\left(I+\frac{1}{a}\sum_{t=1}^{T}x_tx_t'\right). \tag{3.19}$$

PROOF We first clip the predictions of Ridge Regression to $[-Y,Y]$ in Theorem 3.3. In this case the loss at each step can only become smaller, and so the equality transforms to an inequality. Since all the outcomes also lie in $[-Y,Y]$,

the maximum square loss at each step is $4Y^2$. We have the following relations:

$$\frac{1}{1+x_t'A_{t-1}^{-1}x_t} = 1 - \left(\frac{x_t'A_{t-1}^{-1}x_t}{1+x_t'A_{t-1}^{-1}x_t}\right) \text{ and } \frac{x_t'A_{t-1}^{-1}x_t}{1+x_t'A_{t-1}^{-1}x_t} \le \ln(1+x_t'A_{t-1}^{-1}x_t).$$

The last inequality holds because $x_t'A_{t-1}^{-1}x_t$ is non-negative due to the positive definiteness of the matrix A_{t-1}. Thus we can use $\frac{b}{1+b} \le \ln(1+b), b \ge 0$ (it holds at $b=0$, then take the derivatives of both sides). For the equality

$$\sum_{t=1}^{T} \ln(1+x_t'A_{t-1}^{-1}x_t) = \ln\det\left(I + \frac{1}{a}\sum_{t=1}^{T} x_t x_t'\right)$$

see Lemma A.6. ∎

The bound (3.19) is exactly the bound obtained in Vovk (2001, Theorem 4) for the algorithm merging linear experts with predictions clipped to $[-Y, Y]$, which does not have a closed-form description and so is less interesting than clipped Ridge Regression. The bound (3.8) for the AAR has Y^2 in place of $4Y^2$. The regret term in (3.19) has the logarithmic order in T if $\|x_t\|_\infty \le X$ for all t, because, as we already noticed, the determinant of a positive definite matrix is bounded by the product of its diagonal elements (see Beckenbach and Bellman, 1961, Chapter 2, Theorem 7):

$$\ln\det\left(I + \frac{1}{a}\sum_{t=1}^{T} x_t x_t'\right) \le n \ln\left(1 + \frac{TX^2}{a}\right). \tag{3.20}$$

From our Theorem 3.3, we can also deduce Theorem 11.7 of Cesa-Bianchi and Lugosi (2006), which is somewhat similar to our corollary. That theorem implies (3.19) when Ridge Regression's predictions happen to be in $[-Y, Y]$ without clipping (but this is not what Corollary 3.1 asserts). In Theorem 4.6 of Azoury and Warmuth (2001) the same upper bound as in Corollary 3.1 is proven.

The upper bound (3.19) does not hold if the coefficient 4 is replaced by any number less than $\frac{3}{2\ln 2} \approx 2.164$, as can be seen from the example given in Theorem 3 of Vovk (2001), where the left-hand side of (3.19) is $4T + o(T)$, the minimum in the right-hand side is at most T, $Y = 1$, and the logarithm

is $2T \ln 2 + O(1)$. It is also known that there is no algorithm achieving (3.19) with the coefficient less than 1 instead of 4 even in the case where $\|x_t\|_\infty \leq X$ for all t (see Vovk, 2001, Theorem 2).

If the outcomes are not necessarily bounded, it is possible to prove an upper bound without the logarithmic part on the cumulative square loss of Ridge Regression.

Corollary 3.2 *If $\|x_t\|_2 \leq Z$ for all t, then the Ridge Regression algorithm for the learner with $a > 0$ satisfies, at any step T,*

$$\sum_{t=1}^{T} (\gamma_t - y_t)^2 \leq \left(1 + \frac{Z^2}{a}\right) \min_{\theta \in \mathbb{R}^n} \left(\sum_{t=1}^{T} (\theta' x_t - y_t)^2 + a\|\theta\|^2 \right). \tag{3.21}$$

PROOF Qazaz et al. (1997) showed that $1 + x_t' A_j^{-1} x_t \leq 1 + x_t' A_i^{-1} x_t$ for $j \geq i$. We take $i = 0$ and obtain $1 + x_t' A_{t-1}^{-1} x_t \leq 1 + Z^2/a$ for any t in Theorem 3.3.■

This bound is better than the bound in Corollary 3.1 of Kakade and Ng (2004), which has an additional regret term of logarithmic order in time. If we take $a := Z^2/8$, we obtain the bound which is slightly better than (3.17) in that it has the coefficient $Z^2/8$ instead of Z^2 before the norm. On the other hand, the choice of a requires the knowledge of Z a priori to apply Ridge Regression.

Asymptotic properties of the Ridge Regression algorithm can be further studied using Corollary A.1 in Kumon et al. (2009). It states that when $\|x_t\|_2 \leq 1$ for all t, then $x_t' A_{t-1}^{-1} x_t \to 0$ as $t \to \infty$. It is clear that we can replace $\|x_t\|_2 \leq 1$ for all t by $\sup_t \|x_t\|_2 < \infty$. The following corollary states that if there exists a very good expert (asymptotically), then Ridge Regression also predicts very well. If there is no such a good expert, Ridge Regression performs asymptotically as well as the best regularized expert.

Corollary 3.3 *Let γ_t be the predictions output by the Ridge Regression algorithm with parameter $a > 0$. Suppose $\sup_t \|x_t\|_2 < \infty$.*

1. *If*

$$\exists \theta \in \mathbb{R}^n : \sum_{t=1}^{\infty} (\theta' x_t - y_t)^2 < \infty, \tag{3.22}$$

then

$$\sum_{t=1}^{\infty}(\gamma_t - y_t)^2 < \infty.$$

2. *If*

$$\forall \theta \in \mathbb{R}^n : \sum_{t=1}^{\infty}(\theta' x_t - y_t)^2 = \infty, \tag{3.23}$$

then

$$\lim_{T\to\infty} \frac{\sum_{t=1}^{T}(\gamma_t - y_t)^2}{\min_{\theta\in\mathbb{R}^n}\left(\sum_{t=1}^{T}(\theta' x_t - y_t)^2 + a\|\theta\|^2\right)} = 1. \tag{3.24}$$

PROOF **Part 1.** Suppose that the condition (3.22) holds. Then the right-hand side of (3.18) is bounded by a constant (independent of T). By Corollary A.1 in Kumon et al. (2009), the denominators in the left-hand side converge to 1 as $t \to \infty$ and so are bounded. Therefore, the sequence $\sum_{t=1}^{T}(\gamma_t - y_t)^2$ remains bounded as $T \to \infty$.

Part 2. Suppose that the condition (3.23) holds and the right-hand side of (3.18) is bounded above by a constant C. Then for each T there exists θ_T such that

$$\sum_{t=1}^{T}(\theta_T' x_t - y_t)^2 + a\,\|\theta_T\|^2 \le C.$$

It follows that each θ_T belongs to the closed ball with centre 0 and of radius $\sqrt{C/a}$. This ball is a compact set, and thus the sequence θ_T has a subsequence that converges to some $\tilde{\theta}$. For each T_0 we have $\sum_{t=1}^{T_0}(\tilde{\theta}' x_t - y_t)^2 \le C$, because otherwise we would have $\sum_{t=1}^{\hat{T}}(\theta_{\hat{T}}' x_t - y_t)^2 > C$ for a large enough $\hat{T}$ in the subsequence. Therefore, we have arrived at a contradiction: $\sum_{t=1}^{\infty}(\tilde{\theta}' x_t - y_t)^2 \le C < \infty$. Thus if the condition (3.23) holds then right-hand side of (3.18) cannot be bounded above by a constant.

Once we know that the right-hand side of (3.18) tends to ∞ as $T \to \infty$ and the denominators on the left-hand side tend to 1 (this is true by Corollary A.1 in Kumon et al., 2009), (3.24) becomes intuitively plausible since, as far as the conclusion (3.24) is concerned, we can ignore the finite number of ts for which the denominator $1 + x_t' A_{t-1}^{-1} x_t$ is significantly different from 1. However, we will give a formal argument.

The inequality ≥ 1 in (3.24) is clear from (3.18) and $1 + x_t' A_{t-1}^{-1} x_t \geq 1$. We shall prove the inequality ≤ 1 now. Choose a small $\epsilon > 0$. Then starting from some $t = T_0$ we have that the denominators $1 + x_t' A_{t-1}^{-1} x_t$ are less than $1 + \epsilon$. Thus, for $T > T_0$,

$$\begin{aligned}
\sum_{t=1}^{T} (\gamma_t - y_t)^2 &= \sum_{t=1}^{T_0} (\gamma_t - y_t)^2 + \sum_{t=T_0+1}^{T} (\gamma_t - y_t)^2 \\
&\leq \sum_{t=1}^{T_0} (\gamma_t - y_t)^2 + (1+\epsilon) \sum_{t=1}^{T} \frac{(\gamma_t - y_t)^2}{1 + x_t' A_{t-1}^{-1} x_t} \\
&= \sum_{t=1}^{T_0} (\gamma_t - y_t)^2 + (1+\epsilon) \min_{\theta \in \mathbb{R}^n} \left(\sum_{t=1}^{T} (\theta' x_t - y_t)^2 + a \left\| \theta \right\|^2 \right).
\end{aligned}$$

This implies that the left-hand side of (3.24) with lim replaced by lim sup does not exceed $1+\epsilon$, and it remains to recall that ϵ can be taken arbitrarily small.■

3.4 Regression with pointing prediction intervals under discounted square loss

In this section we consider a new setting for online linear regression, more general than in Section 3.2. We look at the two important generalizations: first, in this setting we are allowed to discount the cumulative square loss of the learner and the experts with time when measuring their overall performance. Second, we allow all the outcomes at different steps to belong to different intervals. The learner is given new bounds, upper and lower, for the outcome at each step. We will refer to the process of transforming prediction intervals into point predictions as *pointing* the prediction intervals, and this is what our algorithm does. We show that the two generalizations which we consider in this section closely relate to each other.

It is possible to apply a modification of the Aggregating Algorithm (Algorithm 1) to mix linear predictors in this framework. However, this modification is motivated by defensive forecasting (Algorithm 2), and thus we describe the solution which uses the defensive forecasting technique rather than the Aggregating Algorithm.

The pointing prediction intervals setting can be applied, for example, when the change scale of a sequence of outcomes depends on the current value and volatility of the sequence. As another example, there may be a clear trend in data which does not depend on the input vectors; this may happen even after data preprocessing. As a third example, we may use our algorithm on top of other algorithms whose predictions are intervals for the outcome at the next step (see, e.g., Vovk et al., 2005) to give point predictions, somewhat analogously to boosting (see Freund and Schapire, 1997). We show that in practice these predictions can be more precise than just naive prediction of the centres of the intervals.

From the Bayesian perspective, pointing intervals is quite common: it can be considered as giving an exact prediction in case when a distribution over the outcomes is inferred (and confidence intervals are given). Whereas predicting the mean of the distribution may be optimal under certain conditions (for example, when the process follows probabilistic assumptions and the expecta-

tion w.r.t. the assumed probability measure of the square loss is minimized, see Chapter 4 of Hamilton, 1994), other algorithms may be needed when the conditions are not valid.

If the prediction intervals are constant and known in advance, and the cumulative loss is not discounted, the performance guarantees that we prove for our algorithm coincide with the guarantees in Theorem 3.1. On the other hand, when all the outcomes have large values but small deviation from each other, our algorithm has stronger performance guarantees.

Discounted loss is widely used in finance, online tracking, and other applications (see Gardner, 2006, for the review of exponential smoothing), as we already noticed before. With the absence of competitive properties, exponential smoothing for online prediction was considered already by Muth (1960). A different approach leading to very similar models which uses discounted least squares was proposed by Brown (1963).

It can be beneficial to use discounted loss when the best expert slowly changes over time: when discounting earlier losses, the algorithm will try to adjust to the new best expert faster. As Freund and Hsu (2008) notice, the use of discounted cumulative loss represents an alternative to the "tracking the best expert" framework of Herbster and Warmuth (1998). If the best expert changes after some steps (as in the tracking framework), the algorithm which 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.

Even the logarithmic regret term in the cumulative loss of the algorithms competing with the large classes of experts can be considered to provide too slow convergence for some applications. The upper bound on the exponentially discounted loss has approximately constant order of the regret term in time, which can be beneficial for these applications.

Discounted loss in the online prediction framework was analyzed by Freund and Hsu (2008) in the setting where the predictions are not given, and only the losses of the experts are known. The goal of the learner is to assign weights distribution over the experts at each step such that his expected loss w.r.t. this distribution is less than or close to the loss of the best expert. We

use the square loss function to measure the quality of predictions and prove an upper bound for the weighted cumulative loss of the learner in terms of the weighted cumulative loss of any regularized linear predictor. Since we consider a much more specific setting, it is natural that the order of the regret term for our bounds is better. We are not aware of any other works which consider exponentially discounted square loss in the online competitive prediction framework.

Cesa-Bianchi and Lugosi (2006, § 2.11) discuss another kind of discounting. Their framework allows them to give guarantees only at one moment T chosen in advance (see Theorem 2.8 in Cesa-Bianchi and Lugosi, 2006).

An approach to prove the results for discounted loss reformulated in terms of the Aggregating Algorithm is described in Chernov and Zhdanov (2010).

3.4.1 The prediction protocol and performance guarantees

We modify Protocol 3 to allow new prediction intervals at each step. The outcome set $\Omega_t = [Y_{t,1}, Y_{t,2}]$ is announced by reality at the beginning of each prediction step t. Let the prediction set Γ be the real line $\mathbb{R}$, the index set Θ for the experts be $\mathbb{R}^n$, and the set of input vectors $\mathbf{X}$ be a subset of $\mathbb{R}^n$. The loss function λ is the square loss (2.24):

$$\lambda(y, \gamma) = (\gamma - y)^2,$$

where $\gamma \in \Gamma$ and $y \in \Omega_t$. The prediction process follows Protocol 4.

Protocol 4 Stepwise-bounded online regression

for $t = 1, 2, \ldots$ **do**
  Reality announces $x_t \in \mathbf{X}$ and non-empty $\Omega_t = [Y_{t,1}, Y_{t,2}] \subseteq \mathbb{R}$.
  Experts announce $\xi_t^\theta \in \Gamma_t$.
  Learner predicts $\gamma_t \in \Gamma_t$.
  Reality announces $y_t \in \Omega_t$.
end for

We discount the cumulative loss of the learner and the experts with the factors $\alpha_t \in [0,1]$ after each step (analogously to the end of Section 2.2.2). Our algorithm competes with the linear experts predicting $\xi_t^\theta = \theta' x_t$ at the step t, in other words we take $c = 0$ in (3.2). This class of experts can be easily generalized to $c_t + \theta' x_t d_t$, where $c_t, d_t \in \mathbb{R}$ are constants announced by reality at the beginning of step t, if for a particular task these scaled linear experts are more appropriate. This is done by analogy with the approach of Section 3.2.

We need some preparation to prove a performance guarantee for an algorithm for the learner. The sequence of functions

$$Q_t^\theta(y_t, \gamma_t) := e^{\eta_t\left((\gamma_t - y_t)^2 - (\theta' x_t - y_t)^2\right)}$$

is forecast-continuous and has the defensive property by Lemma 2.6 with $\eta_t \in \left(0, \frac{2}{(Y_{t,2} - Y_{t,1})^2}\right]$, $y_t \in \{Y_{t,1}, Y_{t,2}\}$, and $\gamma_t \in \mathbb{R}$. We take the maximum value for η_t. Therefore, by Lemma 2.3, the sequence of functions

$$Q_T(y_1, \gamma_1, \dots, y_{T-1}, \gamma_{T-1}, y, \gamma) := \int_\Theta \prod_{t=1}^{T-1} \left(Q_t^\theta(y_t, \gamma_t)\right)^{\prod_{i=t}^{T-1} \alpha_i} Q_T^\theta(y, \gamma) P_0(d\theta)$$

is forecast-continuous and has the defensive property for $\eta_t \le \frac{2}{(Y_{t,2} - Y_{t,1})^2}$, $y \in \{Y_{T,1}, Y_{T,2}\}$, and $\gamma \in \mathbb{R}$. By Lemma 2.2 there exists $\gamma_T \in [Y_{t,1}, Y_{t,2}]$ such that $Q_T \le 1$ for all $y \in \{Y_{T,1}, Y_{T,2}\}$.

The following lemma generalizes Lemma 2.2 for the case when the outcome set is the full interval: $\Omega_t = [Y_{t,1}, Y_{t,2}]$.

Lemma 3.4 *If γ_T is such that $Q_T(y_1, \gamma_1, \dots, y_{T-1}, \gamma_{T-1}, y, \gamma_T) \le 1$ for all $y \in \{Y_{T,1}, Y_{T,2}\}$, then $Q_T(y_1, \gamma_1, \dots, y_{T-1}, \gamma_{T-1}, y, \gamma_T) \le 1$ for all $y \in [Y_{T,1}, Y_{T,2}]$.*

PROOF Note that any $y \in [Y_{T,1}, Y_{T,2}]$ can be represented as $y = uY_{T,2} + (1 - u)Y_{T,1}$ for some $u \in [0,1]$. Thus

$$\begin{aligned}(\zeta_1 - y)^2 - (\zeta_2 - y)^2 &= \zeta_1^2 - \zeta_2^2 - 2y(\zeta_1 - \zeta_2) \\ &= u[(\zeta_1 - Y_{T,2})^2 - (\zeta_2 - Y_{T,2})^2] + (1 - u)[(\zeta_1 - Y_{T,1})^2 - (\zeta_2 - Y_{T,1})^2]\end{aligned}$$

for any $\zeta_1, \zeta_2 \in \mathbb{R}$. Due to the convexity of the exponent function, we have for any $\eta_T \geq 0$

$$e^{\eta_T[(\zeta_1-y)^2-(\zeta_2-y)^2]} \leq u e^{\eta_T[(\zeta_1-Y_{T,2})^2-(\zeta_2-Y_{T,2})^2]} + (1-u) e^{\eta_T[(\zeta_1-Y_{T,1})^2-(\zeta_2-Y_{T,1})^2]}.$$

Thus

$$Q_T^\theta(y, \gamma_T) \leq u Q_T^\theta(Y_{T,2}, \gamma_T) + (1-u) Q_T^\theta(Y_{T,1}, \gamma_T),$$

and therefore

$$\begin{aligned} Q_T(y_1, \gamma_1, \ldots, y_{T-1}, \gamma_{T-1}, y, \gamma_T) &\leq u Q_T(y_1, \gamma_1, \ldots, y_{T-1}, \gamma_{T-1}, Y_{T,2}, \gamma_T) \\ &\quad + (1-u) Q_T(y_1, \gamma_1, \ldots, y_{T-1}, \gamma_{T-1}, Y_{T,1}, \gamma_T) \leq 1, \end{aligned}$$

where the second inequality follows from the condition of the lemma. ∎

Denote by X the $T \times n$ matrix consisting of the rows of the input vectors $x_1', \ldots, x_T'$. We shall denote $w_{t,T} := \eta_t \prod_{i=t}^{T-1} \alpha_i$ with $\eta_t = \frac{2}{(Y_{t,2}-Y_{t,1})^2}$. We also have that $W_T := \mathrm{diag}(w_{1,T}, w_{2,T}, \ldots, w_{T,T})$ is the diagonal matrix $T \times T$. We prove the following upper bound for the learner's loss.

Theorem 3.4 *For any $a > 0$, there exists a prediction strategy for the learner achieving, for every T and for any linear predictor $\theta \in \mathbb{R}^n$,*

$$\begin{aligned} \sum_{t=1}^T w_{t,T}(\gamma_t - y_t)^2 \leq \sum_{t=1}^T w_{t,T}(\theta' x_t - y_t)^2 + a\|\theta\|^2 \\ + \frac{1}{2} \ln \det \left(\frac{X' W_T X}{a} + 1 \right). \quad (3.25) \end{aligned}$$

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

$$\begin{aligned} \sum_{t=1}^T w_{t,T}(\gamma_t - y_t)^2 \leq \sum_{t=1}^T w_{t,T}(\theta' x_t - y_t)^2 + a\|\theta\|^2 \\ + \frac{n}{2} \ln \left(\frac{Z^2 \sum_{t=1}^T w_{t,T}}{a} + 1 \right). \quad (3.26) \end{aligned}$$

PROOF We have by Lemma 3.4 that Algorithm 2 predicts γ_t, $t = 1, \ldots, T$, such that

$$Q_T = \int_\Theta \prod_{t=1}^{T} e^{w_{t,T}\left((\gamma_t - y_t)^2 - (\theta' x_t - y_t)^2\right)} P_0(d\theta) \le 1.$$

We take the Gaussian initial weights distribution over the experts with a parameter $a > 0$:

$$P_0(d\theta) = (a/\pi)^{n/2} e^{-a\|\theta\|^2} d\theta.$$

Thus, extracting the loss of the learner from the integral in the previous inequality, we obtain

$$e^{\sum_{t=1}^{T} w_{t,T}(\gamma_t - y_t)^2} (a/\pi)^{n/2} \int_\Theta e^{-\sum_{t=1}^{T} w_{t,T}(\theta' x_t - y_t)^2 - a\|\theta\|^2 + B_0 - B_0} d\theta \le 1.$$

Here $B_0 = \min_{\theta \in \Theta} \left(\sum_{t=1}^{T} w_{t,T}(\theta' x_t - y_t)^2 + a\|\theta\|^2 \right)$ is the loss of the best regularized predictor. In the exponent we have a horizontal shift of a quadratic form by θ. Directly evaluating the integral following Lemma A.1, we get

$$e^{\sum_{t=1}^{T} w_{t,T}(\gamma_t - y_t)^2} e^{-B_0} (a/\pi)^{n/2} \sqrt{\frac{\pi^n}{\det\left(aI + \sum_{t=1}^{T} w_{t,T} x_t x_t'\right)}} \le 1.$$

It is easy to see that $\sum_{t=1}^{T} w_{t,T} x_t x_t' = X' W_T X$. When we take natural logarithms of both parts, we obtain (3.25). We then bound the determinant of the positive definite matrix by the product of its diagonal elements (see Theorem 7 in Chapter 2 of Beckenbach and Bellman, 1961) and obtain (3.26). ∎

Let us consider the special case when there is no discounting ($\alpha_t = 1$ for all t). Then this theorem coincides with the upper bound for the Aggregating Algorithm for Regression (Theorem 3.1) if the prediction intervals do not depend on t: $Y_{t,2} = Y_2$ and $Y_{t,1} = Y_1$.

On the other hand, if the outcomes lie in the tight tube $y_t \in [C_t - \delta, C_t + \delta]$ for some large C_t and small δ, the length of the interval for the outcomes is just 2δ, and so the regret term becomes much smaller than if we used the AAR with $Y = \sup_t C_t + \delta$. Indeed, dividing both parts of the inequality (3.26) by $\eta_t = \frac{1}{2\delta^2}$ we obtain the regret term $n\delta^2 \ln\left(\frac{X^2 T}{2\delta^2 a} + 1\right)$. At the

same time, Theorem 3.1 states the upper bound for the AAR with the regret term $n(\sup_t C_t + \delta)^2 \ln\left(\frac{X^2 T}{a} + 1\right)$ for the same a.

We now consider another special case when the prediction intervals and the discounting coefficients do not depend on t: $Y_{t,2} = Y$, $Y_{t,1} = -Y$, and $\alpha_t = \alpha \in [0, 1]$. The following corollary easily follows from Theorem 3.4 (we use the strategy from this theorem with $a := a/\eta = 2Y^2 a$).

Corollary 3.4 *Assume that $\|x_t\|_\infty \le Z$ for all t. For any $a > 0$, there exists a prediction strategy for the learner achieving, for every T and for any linear predictor $\theta \in \mathbb{R}^n$,*

$$\sum_{t=1}^{T} \alpha^{T-t}(\gamma_t - y_t)^2 \le \sum_{t=1}^{T} \alpha^{T-t}(\theta' x_t - y_t)^2 + a\|\theta\|^2 + nY^2 \ln\left(\frac{Z^2 \sum_{t=1}^{T} \alpha^{T-t}}{a} + 1\right). \quad (3.27)$$

Since $\sum_{t=1}^{T} \alpha^{T-t} = \frac{1-\alpha^{T-1}}{1-\alpha}$ if $\alpha \neq 1$, we can see that the regret is constant $nY^2 \ln\left(\frac{Z^2}{a(1-\alpha)} + 1\right)$. When the losses are discounted, the role of the time is played by the sum of the discounted factors: $\sum_{t=1}^{T} \alpha^{T-t}$. Effectively, only a finite number of recent losses is used to assess the quality of the algorithm.

3.4.2 Derivation of the algorithm

In the proof of Theorem 3.4 we use Algorithm 2 as the strategy for the learner. We will show that Algorithm 8 is a modified version of it, and it is more efficient than the direct use of Algorithm 2 (which requires numerical integration). It requires $O(n^3)$ operations per step, even though usually the process can be accelerated by applying efficient algorithms for solving systems of linear equations. We call this algorithm the Pointing Prediction Intervals Regression algorithm (PPIR).

Algorithm 8 Pointing Prediction Intervals Regression

Initialize $b_0 = 0 \in \mathbb{R}^n$, $A_0 = aI \in \mathbb{R}^n \times \mathbb{R}^n$, $\alpha_0 = 1$.

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

Read new $x_t \in \mathbb{R}^n$, and $[Y_{t,1}, Y_{t,2}] \subseteq \mathbb{R}$, $Y_{t,1} < Y_{t,2}$.

Calculate $\eta_t = \frac{2}{(Y_{t,2} - Y_{t,1})^2}$.

Update $z_t = b_{t-1} + \eta_t \frac{Y_{t,2} + Y_{t,1}}{2} x_t$, $A_t = aI + \alpha_{t-1}(A_{t-1} - aI) + \eta_t x_t x_t'$.

Predict $\gamma_t = z_t' A_t^{-1} x_t$.

Read new y_t and $\alpha_t \in [0, 1]$.

Update $b_t = \alpha_t(b_{t-1} + y_t \eta_t x_t)$.

end for

The most time-consuming operation is the inversion of the updated matrix A_t. If there is no discounting ($\alpha_t = 1$ for all t), this operation can be done by the Sherman-Morrison formula (see Section 2.10 of Press et al., 1992):

$$A_t^{-1} = (A_{t-1} + \eta_t x_t x_t')^{-1} = A_{t-1}^{-1} - \frac{(A_{t-1}^{-1} x_t)(A_{t-1}^{-1} x_t)'}{1/\eta_t + x_t A_{t-1}^{-1} x_t},$$

which requires only $O(n^2)$ operations.

It is easy to check that Algorithm 8 minimizes

$$a\|\theta\|^2 + \eta_T(\theta' x_T - (Y_{T,2} + Y_{T,1})/2)^2 + \sum_{t=1}^{T-1} w_{t,T}(\theta' x_t - y_t)^2 \tag{3.28}$$

in $\theta \in \Theta$. Indeed, by taking the derivative in θ of the quadratic form and finding $\widehat{\theta}$ where it achieves zero, we can obtain that the PPIR predicts $\widehat{\theta}' x_T$.

If the discounting is present, or if smaller and smaller intervals are announced with time, then the coefficients for the recent losses increase in comparison with the coefficients for the older losses. The dependency of predictions on the past data decreases with time, similarly to the suggestion made by Busuttil (2008) for financial applications (the WeCKAAR).

Lemma 3.4 provides the existence of the appropriate predictions at each step and gives a way to find them. Unfortunately, this way requires the time-consuming search for a fixed point and numerical integration when the experts predict according to the linear functions. We have shown in Section 2.2.2 that

the Defensive Forecasting algorithm finds the prediction γ_T satisfying (2.14):

$$(\gamma_T - Y_{T,2})^2 - g_T(Y_{T,2}) = (\gamma_T - Y_{T,1})^2 - g_T(Y_{T,1}), \tag{3.29}$$

where $g_t(y)$ is defined from (2.13):

$$g_T(y) = -\frac{1}{\eta_T} \ln \int_\Theta e^{-\eta_T(\theta' x_T - y)^2} \prod_{t=1}^{T-1} e^{-w_{t,T}(\theta' x_t - y_t)^2} P_0(d\theta)$$

for any $y \in [Y_{T,1}, Y_{T,2}]$.

It is possible that the calculated prediction does not lie in the prediction interval: $\gamma_T \notin [Y_{T,1}, Y_{T,2}]$, in which case the algorithm may just truncate it to the closest point in this segment. The upper bound holds for both truncated and not truncated predictions. The solution of (3.29) in γ_T is given by

$$\begin{aligned}
\gamma_T &= \frac{Y_{T,2} + Y_{T,1}}{2} - \frac{g(Y_{T,2}) - g(Y_{T,1})}{2(Y_{T,2} - Y_{T,1})} \\
&= \frac{Y_{T,2} + Y_{T,1}}{2} \\
&\quad - \frac{1}{2(Y_{T,2} - Y_{T,1})\eta_T} \ln \frac{\int_\Theta e^{-\theta' A_T \theta + 2\theta'\left(b_{T-1} + \eta_T Y_{T,1} x_T\right) - \left(\sum_{t=1}^{T-1} w_{t,T} y_t^2 + \eta_T Y_{T,1}^2\right)} d\theta}{\int_\Theta e^{-\theta' A_T \theta + 2\theta'\left(b_{T-1} + \eta_T Y_{T,2} x_T\right) - \left(\sum_{t=1}^{T-1} w_{t,T} y_t^2 + \eta_T Y_{T,2}^2\right)} d\theta} \\
&= \frac{Y_{T,2} + Y_{T,1}}{2} \\
&\quad - \frac{1}{2(Y_{T,2} - Y_{T,1})\eta_T} \ln e^{\eta_T \left(Y_{T,2}^2 - Y_{T,1}^2 - \left(b_{T-1} + \eta_T \left(\frac{Y_{T,2} + Y_{T,1}}{2}\right) x_T\right)' A_T^{-1} \left(\frac{Y_{T,2} - Y_{T,1}}{2} x_T\right)\right)} \\
&= \left(b_{T-1} + \eta_T \left(\frac{Y_{T,2} + Y_{T,1}}{2}\right) x_T\right)' A_T^{-1} x_T,
\end{aligned} \tag{3.30}$$

where the third equality follows from Lemma A.3. Here we have $A_T = aI + \sum_{t=1}^{T-1} w_{t,T} x_t x_t' + \eta_T x_T x_T'$ and $b_{T-1} = \sum_{t=1}^{T-1} w_{t,T} y_t x_t$.

3.4.3 Experiments with pointing intervals

In our experiments we do not take discounting into account and only investigate the empirical properties of the pointing part of the algorithm. This is because pointing algorithms are less frequently used than the algorithms which

are capable to discount the past data and their empirical properties are less known. We run Algorithm 8 on several artificial and real-world data sets.

Artificial data sets

We imitate time series with different properties, such as a constant term, linear trend, and a seasonal component. Denoting time steps by $t = 1, 2, \ldots, T$, where $T = 300$, and outcomes by y_t, we generate the following data sets:

$$\begin{aligned} D_1 &: y_t = \xi_t, \\ D_2 &: y_t = 3 + \xi_t, \\ D_3 &: y_t = 0.01t + \xi_t, \\ D_4 &: y_t = -0.01t + \xi_t, \\ D_5 &: y_t = \sin(t/20) + \xi_t. \end{aligned} \tag{3.31}$$

Each sequence is corrupted by noise ξ_t. We use 3 types of noise: Gaussian noise $\xi_t \sim N(0, 0.1)$, signed power-law $\xi_t \sim \pm\mathrm{St}(2.4)$ (student-distributed variable with the positive sign with probability 1/2), and correlated Gaussian $\xi_t = 0.8\xi_{t-1} + \eta, \eta \sim N(0, 1)$. All the noise values $\xi_1, \ldots, \xi_T$ are divided by a constant so that the variance of the noise is equal to 0.1. The coefficients in the noise distributions are chosen so that the resulting data sets represent well given classes of noises. In other words, noise values for non-Gaussian distributions should differ significantly from the Gaussian noise, and at the same time they should not be too extreme. Examples of the data sets are presented in Figure 3.1.

We need a training set to find the best parameters for the algorithms which we compare and a test set to calculate the mean square error and assess the performance. We divide each sequence into two equal parts: training set and test set. The coefficients of the functions for y_t were chosen such that training sets are good representatives for the test set (this is especially important for the case of sinus) and the sequences are not extreme.

We analyse sequences (3.31) as time series. In time series there are no explicit input vectors attached to the outcomes. However, we can take vectors

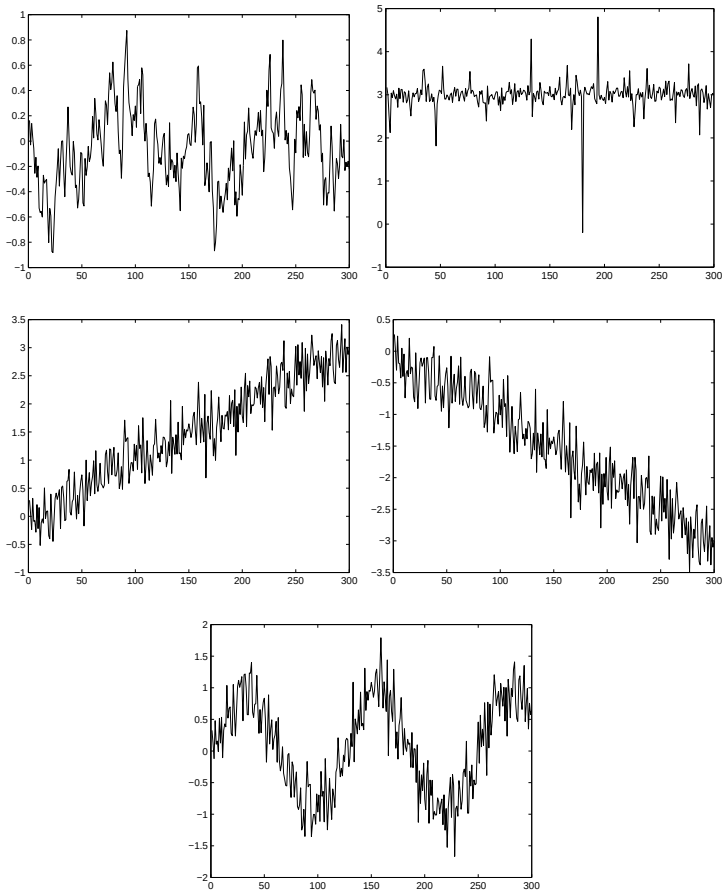


Figure 3.1: Examples of the data sets: D_1 with the Gaussian correlated noise on the left top, D_2 with the power-law noise next right, then upward trend, downward trend, and sinus. The last three are corrupted by Gaussian noise.

consisting of previous observations and use them as input vectors for our algorithms. We chose to use one previous observation: our experiments showed that using more observations improves the absolute performance of all the algorithms, but does not affect their relative performance. In order to assess the quality of predictions, we calculate the cumulative square loss over the test set and divide it by the number of examples (obtaining the Mean Square Error, MSE).

Since our data sets are randomly generated, we need to look at several runs in order to collect reliable statistics. We run algorithms 1000 times for each

time series generating different noise values each time.

We apply three online algorithms: the Ridge Regression algorithm, the Aggregating Algorithm for Regression, and the PPIR. Theoretical guarantees for the cumulative square loss of the online Ridge Regression in the case when the outcomes are uniformly bounded are given by Corollary 3.1. Theoretical guarantees for the cumulative square loss of the Aggregating Algorithm for Regression in the case when the outcomes are uniformly bounded are given by Theorem 3.1. The parameters a for each of the algorithms are chosen on the training set.

The PPIR requires values of $Y_{t,1}, Y_{t,2}$ at each step. We use the Bayesian Ridge Regression predictions to get them (see Section 3.3.1). The Bayesian Ridge Regression algorithm is based on the assumption that the underlying process is of the form $y_t = \theta' x_t + \zeta_t$, where $\zeta_t \sim N(0, \sigma^2)$ is the Gaussian noise with some variance σ^2, and uses the Gaussian prior on models $\theta \sim N(0, (1/a)I)$. In this case the Bayesian Ridge Regression prediction at step T is the mean of the conditional distribution $p(y_T | x_T, x_{T-1}, y_{T-1}, \ldots, x_1, y_1)$ (and corresponds to the prediction of online Ridge Regression):

$$\gamma_T^{RR} = \left(\sum_{t=1}^{T-1} x_t y_t \right)' A_{T-1}^{-1} x_t$$

for $A_{T-1} = aI + \sum_{t=1}^{T-1} x_t x_t'$. A confidence interval for y_t will then be given using $\delta_T^2 = \sigma^2(1 + a x_T' A_{T-1}^{-1} x_T)$: $Y_{T,1} = \gamma_T^{RR} - c\delta_T, Y_{T,2} = \gamma_T^{RR} + c\delta_T$. We choose the coefficient $c\sigma$ such that our algorithm suffers the least loss over the training set.

Table 3.1 gives the number of times the AAR and the PPIR (second raw) suffer less loss over the test set than the RR and the AAR (first column) out of total 1000 Runs for different data sets and different noise distributions. It is interesting that the RR is outperformed everywhere by the AAR or the PPIR. The AAR works well on most of the data sets. As we could expect, the PPIR outperforms all the algorithms on the potentially unbounded data sets with trends D_3, D_4 because it gives its future prediction in accordance with the previous ones using prediction intervals. The fact that the AAR beats other algorithms on the second data set D_2 seems surprising: the AAR moves the

predictions of the RR towards zero, and the mean of the outcomes is equal to 3.

One can extract the measure of statistical significance from the difference between the algorithms in Table 3.1. Testing the hypothesis that one of the algorithms beats another one with probability $\frac{1}{2}$ gives p-values less than 5% if the number in the table is more than 527 or less than 473, and p-values less than 1% if the number in the table is more than 538 or less than 462.

Table 3.1: Number of times the AAR and the PPIR suffer less loss than the RR and the AAR.

	Gaussian		Power-law		Correlated	
	AAR	PPIR	AAR	PPIR	AAR	PPIR
D_1						
RR	759	663	711	667	542	449
AAR	0	668	0	655	0	458
D_2						
RR	1000	90	939	174	887	274
AAR	0	0	0	64	0	111
D_3						
RR	214	1000	415	980	13	921
AAR	0	955	0	707	0	988
D_4						
RR	182	1000	405	981	15	926
AAR	0	970	0	713	0	985
D_5						
RR	988	360	940	380	949	219
AAR	0	48	0	130	0	55

Table 3.2 gives the means of the mean square errors over 1000 runs and their standard deviations. As we can see, most of the differences between the errors of the algorithms are not very large, but as it was shown in Table 3.1 these differences are stable.

Table 3.2: The means and standard deviations of Mean Square Errors over 1000 runs.

	Gaussian		Power-law		Correlated	
	MSE	DEV	MSE	DEV	MSE	DEV
D_1						
RR	0.0995	0.0081	0.0993	0.0324	0.0377	0.0063
AAR	0.0995	0.0081	0.0993	0.0323	0.0377	0.0063
PPIR	0.0994	0.0081	0.0992	0.0323	0.0377	0.0063
D_2						
RR	0.1975	0.0225	0.1984	0.0657	0.0419	0.0077
AAR	0.1959	0.0222	0.1965	0.0644	0.0417	0.0076
PPIR	0.1975	0.0225	0.1984	0.0658	0.0419	0.0077
D_3						
RR	0.2059	0.0215	0.2091	0.0606	0.0426	0.0078
AAR	0.2068	0.0208	0.2077	0.0565	0.0434	0.0078
PPIR	0.2044	0.0217	0.2061	0.0627	0.0426	0.0078
D_4						
RR	0.2060	0.0215	0.2090	0.0607	0.0426	0.0078
AAR	0.2069	0.0208	0.2078	0.0565	0.0434	0.0078
PPIR	0.2045	0.0217	0.2060	0.0629	0.0426	0.0078
D_5						
RR	0.1843	0.0192	0.1843	0.0192	0.0423	0.0075
AAR	0.1834	0.0190	0.1834	0.0190	0.0421	0.0074
PPIR	0.1842	0.0193	0.1842	0.0193	0.0423	0.0075

Real-world data sets

For the second experiment, we use three data sets containing air quality information. The *ozone* data set[1] contains 366 observations of Los Angeles ozone pollution levels in 1976. Each observation is one day. The problem is to predict the daily maximum one-hour-average ozone reading using other 12 variables (features). We exclude three variables which define the date of the observations and 12 observations with missed ozone readings.

[1]This data set is taken from the BLSS Data Library, and accessible here: `ftp://ftp.stat.berkeley.edu/pub/users/breiman/`.

The *airquality* data set[2] contains 154 observations of air quality measurements in New York, May to September 1973. Each observation is one day. The problem is to predict the mean ozone concentration using other 5 variables. We exclude two variables which define the date of the observations and 38 observations with missed ozone readings.

The *NO2* data set[3] contains 500 observations from a road air pollution study collected by the Norwegian Public Roads Administration. The predicted variable is the hourly values of the logarithm of the concentration of NO2 (particles), measured at Alnabru in Oslo, Norway, between October 2001 and August 2003. Seven predictor variables include the logarithm of the number of cars per hour, temperatures, wind speed and direction, hour of day, and the date when the observation was taken. We exclude the date variable, but kept the hour of day variable because the observations in the data set do not come hour by hour, there are many hours missing.

All the observations are sorted by their date and time. We normalize all the features and outcomes to have zero mean and standard deviation one over the first half of each data set (we use it as a training set). We also add an additional bias feature 1 to all the observations.

In this study the intervals for the PPIR are given by the Ridge Regression Confidence Machine (RRCM); see Section 2.3 of Vovk et al. (2005). This interval predictor is based on Ridge Regression. It takes a ridge parameter a_c for Ridge Regression and a significance level parameter $\epsilon \in (0, 1]$. If the data are independent identically distributed and the significance level is ϵ, this algorithm produces a prediction interval for the outcome such that the actual outcome belongs to this interval with the probability $1-\epsilon$. When applied in the on-line mode, the RRCM makes errors at different observations independently. For example, if $\epsilon = 0.1$ then around 90% of the intervals predicted by the RRCM for different input vectors contain the outcomes for these vectors. If the interval for an outcome is infinite, we shrink it to the interval $[-10^5, 10^5]$. We use the intervals predicted by the RRCM as the input parameters for the

[2]This data set is taken from the R data set package, `http://stat.ethz.ch/R-manual/R-devel/library/datasets/html/00Index.html`.

[3]This data set is taken from the StatLib Archive, `http://lib.stat.cmu.edu/datasets/`.

PPIR at each step. We also take the centres of the intervals as the point predictions of the RRCM. By comparing the predictions of our algorithm with these predictions we can say whether the achieved precision is due to the good interval prediction only, or also due to the capability of our algorithm to utilize these intervals.

The parameter a_c is chosen such that the RRCM suffers the least square loss on the training set with the best significance level ϵ. We then fix the parameter a_c and compare the performance of the algorithms for different values of ϵ. We find the ridge parameters for the Ridge Regression, for the AAR, and for the PPIR such that these algorithms suffer the least cumulative square loss on the training set. Note that in online prediction one does not need a validation set: at each step in the training set the prediction is made on a new example.

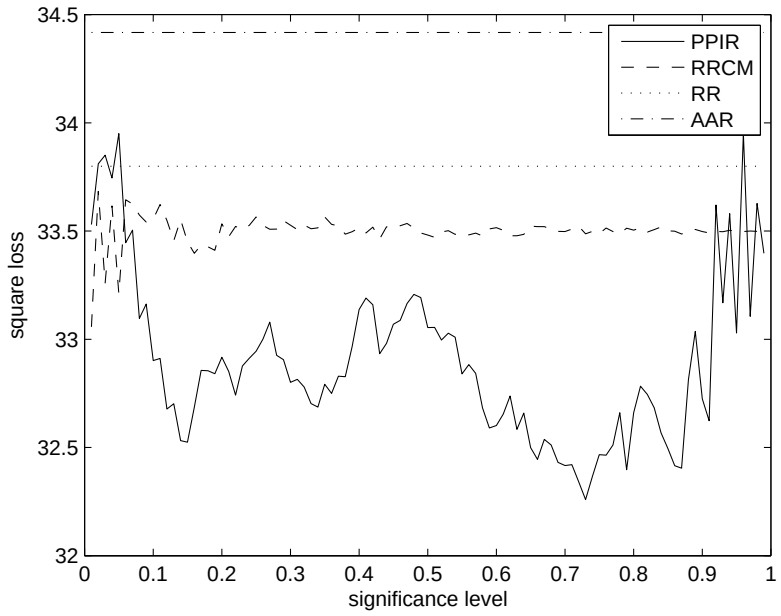


Figure 3.2: Cumulative square losses over the test set of the *ozone* data of different algorithms for different significance levels.

Figure 3.2 shows the cumulative square loss of different algorithms over the test set (the second half) of the *ozone* data for different significance levels ϵ. The ridge for the RRCM, $a_c = 1$, was chosen on the training set. Clearly,

the RR and the AAR do not depend on the significance level and the lines for them are constant. It is interesting to note that the best significance level for the RRCM is very close to 0, whereas the best significance level for the PPIR based on the RRCM is larger. The PPIR outperforms all other algorithms when the significance level ϵ lies in the large interval $[0.06, 0.91]$. We pay particular attention to the small values for the significance level, because they correspond to the reasonable reliability of the intervals given by the RRCM.

The PPIR does not perform as well on the *airquality* data set but still outperforms all the algorithms if the significance level ϵ lies in the intervals $[0.11, 0.24]$ and $[0.34, 0.62]$ or equals 0.01 or 0.03.

On the *NO2* data, the PPIR outperforms the RR and the RRCM if the significance level ϵ belongs to the intervals $[0.04, 0.07]$ and $[0.11, 0.17]$ or equals 0.02. The AAR outperforms all other algorithms on this data set. It is worth noting that excluding one vector from the *NO2* data set which contains an outlier feature value makes the PPIR perform much better: it beats all the algorithms if the significance level is in the range of $[0.02, 0.07]$ and it performs better than the RR and the RRCM if the significance level belongs to $[0.02, 0.23]$.

3.5 Generalized linear models

In this section, we develop an algorithm competitive with the benchmark class of generalized linear models under square loss function. Generalized linear models have been found very useful in statistical analysis (see McCullagh and Nelder, 1989) to solve bounded regression problems. Classification problems are often solved by means of these models as well.

We use the Aggregating Algorithm (Algorithm 1) somewhat similarly to the AAR described in Section 3.2. Whereas the AAR only covers the class of linear experts, our new algorithm also covers other popular classes of experts, which are more efficient in that their predictions always belong to the interval $[Y_1, Y_2]$ assumed to contain the label that is being predicted. When specialized to the case of linear experts, our general loss bound coincides with the known optimal bound for the AAR. A disadvantage of our algorithm is that we need to know $[Y_1, Y_2]$ *a priori* to be able to apply it (as for the PPIR described in Section 3.4).

The problems which cover in part generalized linear models from the perspective of online convex optimization are analyzed in Hazan et al. (2007). Our algorithm can be applied to some non-convex functions, which is impossible for the methods of online convex optimization even for a compact set of experts.

Generalized linear models are popular in Bayesian statistics for solving classification problems (see Relevance Vector Machines in Bishop, 2006, Section 7.2.3). From the competitive prediction prospective, Bayesian mixtures are analogous to the Aggregating Algorithm competing under the logarithmic loss function. Upper bounds on the logarithmic loss are proved by Kakade and Ng (2004), Kivinen and Warmuth (2001), and Banerjee (2007) using different approaches. In this section, we prove upper bounds on the square loss, which is more often used in practice to compare the performances of different algorithms.

This problem does not appear to be analytically tractable. Therefore, we develop a prediction algorithm using the Markov chain Monte Carlo method, which is shown to be fast and reliable in many cases. Monte Carlo methods are

well known in the Bayesian community (Neal, 2003). They are also applied by, for example, Dalalyan and Tsybakov (2009), to explore exponential weighting schemes in problems with high dimension of the input vectors.

We give suggestions about choosing the parameters of our algorithm and perform experiments with it on a toy data set and two real world ozone level data sets. The results of this section are described in Zhdanov and Vovk (2010).

3.5.1 Performance guarantee

We consider Protocol 3 with $\Omega = [Y_1, Y_2]$, $\Gamma = \mathbb{R}$, and the square loss (2.24):

$$\lambda(y, \gamma) = (\gamma - y)^2,$$

where $\gamma \in \Gamma$ and $y \in \Omega$. Signals x_t come from a set $\mathbf{X} \subseteq \mathbb{R}^n$.

The learner wishes to compete with the class of *generalized linear experts* indexed by $\theta \in \Theta = \mathbb{R}^n$. Expert θ's prediction at step t is denoted ξ_t^θ and is equal to

$$\xi_t^\theta = Y_1 + (Y_2 - Y_1)\sigma(\theta' x_t). \tag{3.32}$$

Here $\sigma : \mathbb{R} \to \mathbb{R}$ is a fixed *activation function*. We have $\sigma : \mathbb{R} \to [0, 1]$ in all the cases except linear regression (see below). If the range of the function σ is $[0, 1]$, the experts necessarily give predictions from $[Y_1, Y_2]$.

Assume that the function

$$b(u, z) := \left(\frac{d\sigma(z)}{dz}\right)^2 + (\sigma(z) - u)\frac{d^2\sigma(z)}{dz^2} \tag{3.33}$$

is uniformly bounded: $b := \sup_{u \in [0,1], z \in \mathbb{R}} |b(u, z)| < \infty$. We will see below that this assumption holds for the most popular generalized linear regression models. We prove the following upper bound on the learner's loss.

Theorem 3.5 *Let $a > 0$. There exists a prediction strategy for the learner such that for any positive integer T, for any sequence of outcomes of length T,*

and any $\theta \in \mathbb{R}^n$, the cumulative loss L_T of the learner satisfies

$$L_T \leq L_T^\theta + a\|\theta\|^2 + \frac{(Y_2 - Y_1)^2}{4} \ln \det \left(I + \frac{b(Y_2 - Y_1)^2}{a} \sum_{t=1}^{T} x_t x_t' \right), \qquad (3.34)$$

where $b := \sup_{u \in [0,1], z \in \mathbb{R}} |b(u,z)|$ and $b(u,z)$ is defined by (3.33). If, in addition, $\|x_t\|_\infty \leq X$ for all t, then

$$L_T \leq L_T^\theta + a\|\theta\|^2 + \frac{n(Y_2 - Y_1)^2}{4} \ln \left(1 + \frac{b(Y_2 - Y_1)^2 X^2 T}{a} \right). \qquad (3.35)$$

PROOF The regularized cumulative loss of the expert θ at the step T can be expressed as

$$Q(\theta) := \sum_{t=1}^{T} \left((Y_2 - Y_1)\sigma(\theta' x_t) + Y_1 - y_t \right)^2 + a\|\theta\|^2.$$

Because of the second addend in the definition of $Q(\theta)$, $Q(\theta) \to \infty$ as $\|\theta\| \to \infty$. Therefore $\min_\theta Q(\theta)$ is attained at some point. Let θ_0 be a point where it is attained, and thus $\nabla Q(\theta_0) = 0$. We use Taylor expansion of $Q(\theta)$ in the point θ_0:

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

where ϕ is a convex combination of θ_0 and θ. Here H is the Hessian matrix of $Q(\theta)$, the matrix of its second derivatives. By δ_i^j we denote the Kronecker delta. The second partial derivative of Q is expressed as follows:

$$\begin{aligned} \frac{\partial^2 Q}{\partial\theta_i \partial\theta_j} &= 2a\delta_i^j + 2(Y_2 - Y_1)^2 \\ &\quad \cdot \sum_{t=1}^{T} \left(\frac{\partial\sigma(\theta' x_t)}{\partial\theta_i} \frac{\partial\sigma(\theta' x_t)}{\partial\theta_j} - \left(\frac{y_t - Y_1}{Y_2 - Y_1} - \sigma(\theta' x_t) \right) \frac{\partial^2\sigma(\theta' x_t)}{\partial\theta_i \partial\theta_j} \right) \\ &= 2a\delta_i^j + 2(Y_2 - Y_1)^2 \sum_{t=1}^{T} x_{t,i} x_{t,j} b\left(\frac{y_t - Y_1}{Y_2 - Y_1}, \theta' x_t \right). \end{aligned}$$

It is clear now that the matrix $H(\phi)$ can be represented as follows:

$$H(\phi) = 2aI + 2(Y_2 - Y_1)^2 X' \Gamma(\phi) X,$$

where X is the design matrix $T \times n$ consisting of the rows of the input vectors $x_1', \ldots, x_T'$. Here $\Gamma(\phi)$ is the diagonal $T \times T$ matrix which has the coefficients $b(u_1, \phi' x_1), \ldots, b(u_T, \phi' x_T)$ on the diagonal (with $u_i = \frac{y_i - Y_1}{Y_2 - Y_1}$, $i = 1, \ldots, T$). Since $\Gamma(\phi)$ is a symmetric matrix, we can see (Harville, 1997, Theorem 21.5.6) that

$$\psi' \Gamma(\phi) \psi \le \psi' \lambda_{\max} \psi \tag{3.36}$$

for any $\psi \in \mathbb{R}^n$, where $\lambda_{\max}$ is the supremum over maximum eigenvalues of $\Gamma(\phi)$. Since $b(u_t, \phi' x_t)$ is uniformly bounded, we have $\lambda_{\max} \le b$.

We can take $\psi = X(\theta - \theta_0)$ and obtain from (3.36) that

$$Q(\theta) \le Q(\theta_0) + (\theta - \theta_0)'(aI + b(Y_2 - Y_1)^2 X' X)(\theta - \theta_0).$$

By Lemma 3.2 we obtain that for any $\theta \in \Theta$ the loss of the Aggregating Algorithm satisfies

$$L_T \le L_T^\theta + a\|\theta\|^2 + \frac{1}{2\eta} \ln \det \left(I + \frac{b(Y_2 - Y_1)^2}{a} \sum_{t=1}^{T} x_t x_t' \right).$$

We take the maximum value for η, $\eta = \frac{2}{(Y_2 - Y_1)^2}$ (recall Lemma 2.5). The determinant of a symmetric positive definite matrix is upper bounded by the product of its diagonal elements (see Beckenbach and Bellman, 1961, Chapter 2, Theorem 7):

$$\det \left(I + \frac{b(Y_2 - Y_1)^2}{a} \sum_{t=1}^{T} x_t x_t' \right) \le \left(1 + \frac{b(Y_2 - Y_1)^2 T X^2}{a} \right)^n.$$

This concludes the proof. ∎

The order of the regret term in bound (3.35) is logarithmic in the number of steps. It matches the order of the best bounds for the linear regression problem under square loss proved in Theorem 3.1 and for the classification prob-

lem using generalized linear regression experts under logarithmic loss proved by Kakade and Ng (2004).

The prediction strategy achieving (3.34) is formulated as Algorithm 9 (page 138), calculated with the number of iterations $M \to \infty$; we also call Algorithm 9 the Aggregating Algorithm for Generalized Linear Models (AAGLM). In Section 3.5.3, we derive it and describe its parameters. Even though Algorithm 9 is an online algorithm, it is easy to apply it in the batch setting: it suffices to consider each example in the test set as the next example after the training set.

3.5.2 Examples of the models and performance guarantees

Now we give some examples of generalized linear models and bounds on the losses of the corresponding algorithms.

Linear regression

We have for linear regression

$$\sigma(z) = \frac{z}{Y_2 - Y_1} - Y_1, \quad z \in \mathbb{R} \tag{3.37}$$

(so that the experts predict $\xi_t^\theta = \theta' x_t$). The derivatives are equal to

$$\frac{d\sigma(z)}{dz} = \frac{1}{Y_2 - Y_1}$$

and

$$\frac{d^2\sigma(z)}{dz^2} = 0.$$

Using these expressions in the derivatives of σ in (3.33) we obtain

$$b(u, z) = \frac{1}{(Y_2 - Y_1)^2}.$$

Using $b = \frac{1}{(Y_2-Y_1)^2}$ in (3.35), we have the following corollary for the linear regression experts.

Corollary 3.5 *There exists a prediction strategy for the learner achieving*

$$L_T \leq L_T^\theta + a\|\theta\|^2 + \frac{n(Y_2 - Y_1)^2}{4} \ln\left(1 + \frac{X^2 T}{a}\right)$$

for any expert $\theta \in \mathbb{R}^n$ predicting according to (3.32) *and* (3.37)*.*

As we can see, the upper bound is the same as the upper bound (3.9) for the AAR. This bound is known to have the best possible order (see Vovk, 2001, Theorem 2) for the linear experts.

Logistic regression

We have for logistic regression

$$\sigma(z) = \frac{1}{1 + e^{-z}}, \quad z \in \mathbb{R}. \tag{3.38}$$

The derivatives are equal to

$$\frac{d\sigma(z)}{dz} = \sigma(z)(1 - \sigma(z))$$

and

$$\frac{d^2\sigma(z)}{dz^2} = \sigma(z)(1 - \sigma(z))(1 - 2\sigma(z)).$$

Using these expressions in the derivatives of σ in (3.33), we obtain

$$b(u, z) = \sigma^2(z)(1 - \sigma(z))^2 + (\sigma(z) - u)\sigma(z)(1 - \sigma(z))(1 - 2\sigma(z)).$$

We have $|b(u, z)| \leq b < \frac{5}{64}$. Using this in (3.35), we have the following corollary for logistic regression experts.

Corollary 3.6 *Let $a > 0$. There exists a prediction strategy for the learner*

achieving

$$L_T \leq L_T^\theta + a\|\theta\|^2 + \frac{n(Y_2 - Y_1)^2}{4} \ln\left(1 + \frac{5(Y_2 - Y_1)^2 X^2 T}{64a}\right)$$

for any expert $\theta \in \mathbb{R}^n$ *predicting according to* (3.32) *and* (3.38).

Probit regression

We have for probit regression

$$\sigma(z) = \Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-v^2/2} dv, \quad z \in \mathbb{R}, \tag{3.39}$$

where Φ is the cumulative distribution function of the normal distribution with zero mean and unit variance. The derivatives are equal to

$$\frac{d\sigma(z)}{dz} = \frac{1}{\sqrt{2\pi}} e^{-z^2/2}$$

and

$$\frac{d^2\sigma(z)}{dz^2} = -\frac{z}{\sqrt{2\pi}} e^{-z^2/2}.$$

Using these expressions in the derivatives of σ in (3.33), we obtain

$$b(u, z) = \frac{1}{2\pi} e^{-z^2} - (\sigma(z) - u) \frac{z}{\sqrt{2\pi}} e^{-z^2/2}.$$

We have $|b(u, z)| \leq b < \frac{25}{128}$. Using this in (3.35), we have the following corollary for probit regression experts.

Corollary 3.7 *Let* $a > 0$. *There exists a prediction strategy for the learner achieving*

$$L_T \leq L_T^\theta + a\|\theta\|^2 + \frac{n(Y_2 - Y_1)^2}{4} \ln\left(1 + \frac{25(Y_2 - Y_1)^2 X^2 T}{128a}\right)$$

for any expert $\theta \in \mathbb{R}^n$ *predicting according to* (3.32) *and* (3.39).

Complementary log-log regression

We have for the complementary log-log regression

$$\sigma(z) = 1 - \exp(-\exp(z)), \quad z \in \mathbb{R}. \tag{3.40}$$

When the argument z of the complementary log-log function $1-\exp(-\exp(z))$ approaches minus infinity, this function is similar to the logistic function $\frac{1}{1+e^{-z}}$ and tends to zero. When the argument approaches infinity, the function tends to one more quickly than the logistic function. This can be used in problems with asymmetry in outcomes. The derivatives of $\sigma(z)$ are equal to

$$\frac{d\sigma(z)}{dz} = e^z(1-\sigma(z))$$

and

$$\frac{d^2\sigma(z)}{dz^2} = e^z(1-\sigma(z))(1-e^z).$$

Using these expressions in the derivatives of σ in (3.33), we obtain

$$b(u,z) = e^{2z}(1-\sigma(z))^2 + e^z(1-\sigma(z))(1-e^z).$$

We have $|b(u,z)| \le b < \frac{17}{64}$. Using this in (3.35), we have the following corollary for complementary log-log regression experts.

Corollary 3.8 *Let $a > 0$. There exists a prediction strategy for the learner achieving*

$$L_T \le L_T^\theta + a\|\theta\|^2 + \frac{n(Y_2-Y_1)^2}{4} \ln\left(1 + \frac{17(Y_2-Y_1)^2X^2T}{64a}\right)$$

for any expert $\theta \in \mathbb{R}^n$ predicting according to (3.32) *and* (3.40).

3.5.3 Derivation of the prediction algorithm

Our prediction algorithm differs from many algorithms commonly used to fit a generalized linear model (see, for example, McCullagh and Nelder, 1989).

First, instead of fitting the data with the best parameter θ (as in logistic regression) it uses the regularization parameter $a > 0$ preventing θ to be too large, and thus preventing overfitting to a certain extent. Second, it looks to minimize the square loss instead of the logarithmic loss. To predict at step T, it works with the function

$$\sum_{t=1}^{T-1}(\xi_t^\theta - y_t)^2 + a\|\theta\|^2 \tag{3.41}$$

with ξ_t^θ from (3.32). Third, it does not look for the best regularized expert θ, but at each prediction step it mixes all the experts using the Aggregating Algorithm (Algorithm 1), thus preventing overfitting even further.

We apply the AA with the Gaussian initial distribution over experts:

$$P_0(d\theta) = (a\eta/\pi)^{n/2}\exp(-a\eta\|\theta\|^2)d\theta \tag{3.42}$$

for some $a > 0$. We use the substitution function (2.26) to give predictions. It allows us to use generalized predictions (2.3) calculated from unnormalized weights (2.2) and with factor $(a\eta/\pi)^{n/2}$ omitted. Normalization is avoided because calculating the normalizing constant is a computationally inefficient operation for our task. By Lemma 2.5 we take the maximum value for η, $\eta = \frac{2}{(Y_2-Y_1)^2}$. We denote (using (3.32))

$$w_T(\theta) := \exp\left(-a\eta\|\theta\|^2 - \eta\sum_{t=1}^{T}\left(\xi_t^\theta - y_t\right)^2\right). \tag{3.43}$$

Algorithm 9 is based on the MCMC technique of numerical integration in (2.3) at $y = Y_1$ and $y = Y_2$; Andrieu et al. (2003) give a good MCMC survey. The function $e^{-\eta(\xi_t^\theta - y)^2}$ needs to be integrated with respect to the unnormalized posterior distribution $P_{t-1}(d\theta)$. We choose to use the simple Metropolis sampling to sample θ from the posterior.

Metropolis sampling from a distribution $\mathcal{P}$ is an iterative process, with the initial value θ^0 and a simple proposal distribution. We choose the Gaussian proposal distribution $N(0, \sigma^2)$ with the parameter σ^2 chosen on the data. At

each iteration $i = 1, \ldots, M$, the update for θ is sampled from the proposal distribution:

$$\theta^i = \theta^{i-1} + \zeta^i, \;\; \zeta^i \sim N(0, \sigma^2).$$

The updated θ^i is accepted with the probability $\min\left(1, \frac{f_{\mathcal{P}}(\theta^i)}{f_{\mathcal{P}}(\theta^{i-1})}\right)$. Here $f_{\mathcal{P}}(\theta)$ is the value of the density function for the distribution $\mathcal{P}$ at the point θ. By accepting and rejecting the updates, the values of the parameter θ move closer to the value where the maximum of the density function $f_{\mathcal{P}}$ is achieved. Thus sampling is performed from the area with high density of $\mathcal{P}$ and covers the tails of it only occasionally. Therefore numerical integration using sampling with MCMC is often more efficient than the usual Monte Carlo sampling from the uniform distribution. Sometimes the updates are accepted even if they do not move the next θ closer to the maximum (this happens when $\frac{f_{\mathcal{P}}(\theta^i)}{f_{\mathcal{P}}(\theta^{i-1})} < 1$). This may allow the algorithm to move away from the local maxima of the density function.

It is common when using the MCMC approach to have a "burn-in" stage, at which the integral is not calculated, but the algorithm is looking for the best "locality" for θ. This stage is used to avoid the error accumulated while the algorithm is still looking for the correct location of the main mass of the distribution. Instead of that, at each prediction step t we take the new starting point θ^0 for the Metropolis sampling to be the last point θ^M achieved on the previous step $t - 1$.

In the case when the dimension n of input vectors is large and the sampling is not very efficient, one can use more advanced techniques, such as adaptive sampling or Slice sampling (Neal, 2003). However, when we tried several versions of Slice sampling, the convergence speed on our data sets was slower than for Metropolis sampling.

The function (3.41) is not necessarily convex in θ, so it may have several local minima. Thus we cannot use the Laplace approximation for the integral and obtain reliable Iteratively Reweighted Least Square estimation of θ, the common approach to give predictions when working with generalized linear models. The MCMC approach to calculating similar integrals for Bayesian prediction models was analyzed by Neal (1999). It is stated there that it

takes $O(n^3)$ operations to calculate a general integral.

Algorithm 9 AAGLM

Require: Bounds for the outcomes Y_1, Y_2,
maximum number of MCMC iterations $M > 0$,
standard deviation $\sigma > 0$,
regularization coefficient $a > 0$.

Calculate $\eta := \frac{2}{(Y_2 - Y_1)^2}$.
Initialize $\theta_0^M := 0 \in \Theta$.
for $t = 1, 2, \ldots$ **do**
  $G_{t,1} := 0$, $G_{t,2} := 0$.
  Define $w_{t-1} : \Theta \to [0, \infty)$ by (3.43) with $t - 1$ in place of T.
  Read $x_t \in \mathbb{R}^n$.
  Initialize $\theta_t^0 := \theta_{t-1}^M$.
  for $m = 1, 2, \ldots, M$ **do**
    $\theta^* := \theta_t^{m-1} + N(0, \sigma^2 I)$.
    if $w_{t-1}(\theta^*) \geq w_{t-1}(\theta_t^{m-1})$ **then**
      $\theta_t^m := \theta^*$.
    else
      Flip a coin with success probability $w_{t-1}(\theta^*)/w_{t-1}(\theta_t^{m-1})$.
      if success **then**
        $\theta_t^m := \theta^*$.
      else
        $\theta_t^m := \theta_t^{m-1}$.
      end if
    end if
    $G_{t,1} := G_{t,1} + e^{-\eta(\xi_t^{\theta_t^m} - Y_1)^2}$, $G_{t,2} := G_{t,2} + e^{-\eta(\xi_t^{\theta_t^m} - Y_2)^2}$.
  end for
  Output prediction $\gamma_t := \frac{1}{2}\left(Y_2 + Y_1 + \frac{\ln G_{t,2} - \ln G_{t,1}}{\eta(Y_2 - Y_1)}\right)$.
  Read $y_t \in [Y_1, Y_2]$.
end for

3.5.4 Experiments

In this section we investigate empirical properties of our algorithm on toy and real data sets and suggest ways to choose the parameters for it.

Toy data set

In this experiment we aim to emphasize the main properties of competitive online algorithms: how they behave if the data follow the assumptions of one of the experts, and when the data fail to do so; how quickly online algorithms adjust to the substantial changes in the properties of the data.

Consider the following online classification problem. Let $x_t \in \mathbb{R}$ be the input vectors $x_1 = -50, x_2 = -49.9, \ldots, x_T = 100$, real numbers from -50 to 100 with step 0.1. Let the outcomes y_t be

$$y_t = \begin{cases} 1 & \text{if } x_t < -10 \text{ or } 10 < x_t < 50, \\ 0 & \text{otherwise.} \end{cases}$$

We add the bias one as the second component of each input vector (input vector dimension becomes equal to 2). We try to predict this sequence online step by step by Algorithm 9 competing with logistic experts. The result is presented on Figure 3.3. We also show the best fitted logistic predictor achieving the minimum cumulative square loss. Notice three following interesting properties of the picture.

The predictions of the AAGLM tend to the predictions of the best logistic regression fitted to the whole data as T becomes large. This matches the fact that the mean loss of the AAGLM converges to the mean loss of the best logistic regression; see Corollary 3.6.

When $x \in [-10, 50)$ both algorithms suffer large loss. Corollary 3.6 ensures that the AAGLM does not suffer much more than the best logistic regression.

During each period, the AAGLM tries to fit a logistic regression function in this and the previous periods. The dependence on the previous periods is due to the fact that the trained AAGLM is equivalent to the untrained AAGLM which starts predicting with initial weights distribution $P_t^*(d\theta)$, where t is the number of steps in the previous periods.

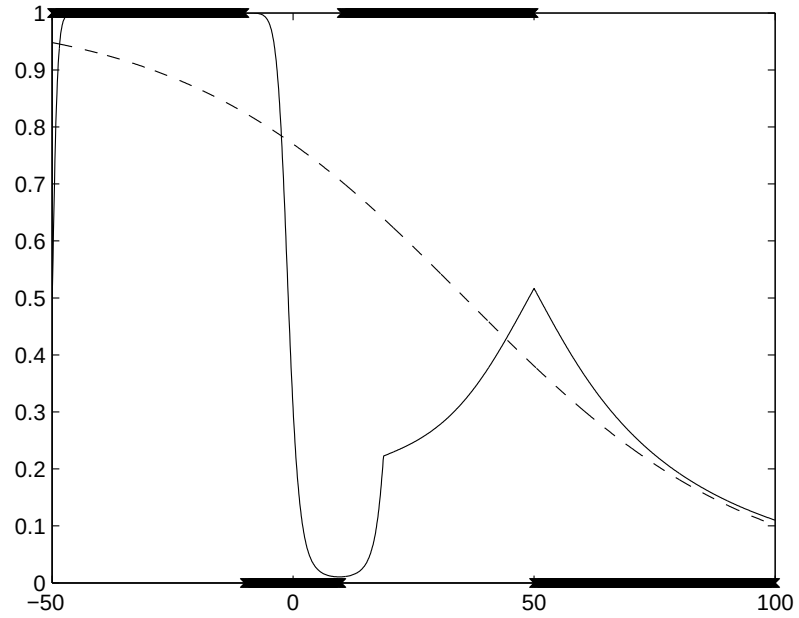


Figure 3.3: Sequential predictions of the AAGLM algorithm for the two-class classification problem. The dashed curve is the predictions of the best logistic regression (under square loss) on all the data. The horizontal axis contains the input vectors, the vertical axis contains the outcomes and the predictions of the outcomes by the two algorithms.

In order for expert-based algorithms to predict well on a certain type of data, the best expert should suffer small loss on these data. If the sequence of outcomes has several regimes which rarely switch from one to another, like in our figure, "tracking the best expert" (Herbster and Warmuth, 1998; Vovk, 1999) may be a more suitable framework.

The parameters of the AAGLM used in these studies are $Y_1 = 0, Y_2 = 1$, $M = 1000$ (we did not use "burn-in" stage), $\sigma = 0.00001$, $a = 10^{-100}$. Increasing the regularization coefficient a leads to the regularization towards 0.5 (as expected). Increasing M accelerates somewhat the reaction in the very beginning of each turn, but the main trend does not change. Too low value for σ leads to slower convergence. Too high value of this parameter forces the algorithm to fluctuate between two classes, and never find a stable solution (this is expected as well since with large σ the numerical integration becomes less precise). It is common when applying the MCMC technique to use the fol-

lowing rule of thumb to determine the value of the parameter σ: the rejection rate should be 30–70%.

Ozone data set

We perform empirical studies on two real-world data sets described by Zhang and Fan (2008).

Data sets The data sets contain various meteorology and ozone data for the Houston, Galveston, and Brazoria (HGB) area in Texas, USA, day by day for 7 years, 1998–2004. We use both one-hour (ozone1) and eight-hour (ozone8) ozone data sets: they contain the the daily maximum of 1 hour (ozone1) ozone concentration and the daily maximum of the average over 8 consecutive hours (ozone8) ozone concentration. Each observation is one day. Each observation has 72 features of various measures of air pollutant and meteorological information for the target area in the study. Each observation is assigned the label 1 (we say its outcome is equal to 1) if the ozone level exceeds the danger threshold, which is 120 parts per billion (ppb) for ozone1 and 80 ppb for ozone8; otherwise, the observation has the label 0 (outcome is equal to 0). The data are collected online, so online prediction algorithms are more appropriate for the study than batch algorithms. They are able to predict ozone levels day by day incorporating the information from all the previous days. Therefore we consider online two-class classification problems.

Zhang and Fan (2008) showed that all 72 features may be relevant to the prediction problem, and thus we decided to use all of them to train our algorithms. We replace the missing values of the features by the mean of the available values of them from the first year (we use the first year data as our training set).

The data sets are very skewed: the number of positive examples is very low (73 for ozone1 and 160 for ozone8 out of 2534 observations). It can be expected that for such data sets complementary log-log (comlog) regression performs better than logistic or probit regression (as explained in the description of the comlog activation function). Indeed, the square loss suffered by logistic regression trained on the whole data set is 48.4178 for ozone1 and 96.2846

for ozone8. At the same time, the square loss suffered by comlog regression trained on the whole data set is 46.7815 and 94.8399 respectively. Thus we use the comlog activation function (3.40) in this experimental study.

Algorithms and results We normalize all the features to have mean zero and maximum absolute value one over the first year. We also add an additional bias feature 1 to all the examples.

We compare different algorithms in two regimes: the online regime and the (incremental) batch regime. In the online regime the algorithms are retrained as soon as a new observation is obtained. In the batch regime the algorithms are only retrained yearly on all the past data. Zhang and Fan (2008) suggested this regime as the most realistic for meteorologists (they did not consider the online regime though).

For the online regime the AAGLM parameters M, σ, a, and the length of the burn-in stage are chosen to suffer the least square loss over the first year. We choose σ from the range $10^{-k}, 5{\cdot}10^{-k}$, $k = 0, 1, \dots, 5$. We choose a from the range $10^{-k}, 5 \cdot 10^{-k}$, $k = -1, 0, 1, \dots, 10$. The best parameters are $M = 2500$, $\sigma = 0.01$, $a = 0.1$, and the length of the burn-in stage is 2000. It is interesting to note that for both data sets the best parameters are the same up to our precision. This may mean that the best parameters for the algorithm depend mostly on the input vectors, and not on the outcomes (because the input vectors are the same for our data sets).

The batch regime of the AAGLM can be understood as just one step of the online algorithm repeated for each new test example. During the training stage the batch algorithm does not calculate the integral but saves the values of θ obtained at each iteration $m = 1, 2, \dots, M$. At the prediction stage, the algorithm calculates the integral using the saved values of θ computed on the iterations between $B < M$ and M, where B is the length of the burn-in stage. We choose the same parameters $\sigma = 0.01$, $a = 0.1$ as for the online version, and $B = 5000$, $M = 15000$ to ensure good convergence. There are no theoretical guarantees for the batch setting.

The first algorithm with which we compare the AAGLM is the online comlog regression minimizing logarithmic loss (standard generalized regres-

sion model): at each step t it uses all the previous steps to find the best parameter $\hat{\theta}$ and then gives its prediction $\sigma(\hat{\theta}' x_t)$ according to this parameter. In terms familiar from the online prediction literature, it corresponds to the Follow the Best Expert predictor under the logarithmic loss function, the natural competitor to our algorithm.

In the batch regime, in the beginning of each year, the best parameter $\tilde{\theta}$ is found on all the previous years. All the predictions in the following year are made using this $\tilde{\theta}$.

We also calculate the performance of the linear Support Vector Machine (SVM) and the SVM with the RBF kernel, implemented by Chang and Lin (2001). The SVM with the RBF kernel showed the best performance on ozone8 in a different framework (Zhang and Fan, 2008). In the online regime the SVMs predict only one next outcome at each step and retrain after the actual outcome is announced. The parameter of the kernel and the parameter C are chosen to achieve the least square loss over the first year. Note that in the online regime one does not need the validation set: the training set at each step does not include the next test example at which prediction is made. Thus the risk of overfitting is less than if the testing was done on the training set.

In the batch regime, at the beginning of each year, the SVMs are retrained on all the previous years. All the predictions in the following year are made using these trained SVMs. The parameter of the kernel and the parameter C are chosen using 5-fold cross-validation: all the data from the previous years are randomly separated into 5 parts. Four parts are used to train the SVM, the fifth part is used for testing: the square loss is calculated over the fifth part. This procedure repeats 5 times with different combinations of the parts. The parameters of the SVMs are chosen to suffer the least average square loss.

Since the number of positive examples is small, it makes sense to compare the precision of the predicting algorithms with the precision of the zero predictor: the predictor which always predicts low ozone concentration.

The minimal square loss which is suffered by the comlog regression model trained on the whole data set is equal to 28.0001 for ozone1 and 75.0001 for ozone8. This loss is unrealistic since we do not know all the observations when start predicting, and included here to specify the limitations of the generalized

regression model. The square loss of the online comlog regression over this period is equal to 105.1823 and 186.6147 respectively. The square loss of the AAGLM over this period is equal to 66.2727 and 120.8483 respectively. At the same time, the upper bounds on its loss from Corollary 3.8 have the values 323.9632 for ozone1 and 371.3681 for ozone8. The zero predictor suffers the square loss 73 and 160 respectively.

Table 3.3 contains average mean square errors for different algorithms over the last 6 years (2–7) in the data sets.

Table 3.3: Average MSEs of different algorithms over the last 6 years of ozone1/ozone8.

Algorithm	MSE online	MSE batch
AAGLM	8.2159/15.3288	8.2163/15.7552
SVM rbf	9.7607/14.8806	9.4497/15.9679
SVM linear	8.8624/16.1532	8.8500/16.5264
Comlog	14.4578/24.2686	13.8096/27.9474
Zeros	9.6667/21.3333	9.6667/21.3333

Figure 3.4 presents *precision* (the number of correctly identified high ozone days divided by the total number of the predicted high ozone days) and *recall* (the number of correctly identified high ozone days divided by the total number of the actual high ozone days) for different threshold values calculated for the last 6 years of ozone8. It contains information for the four algorithms applied in the online regime. The area under the curve for the AAGLM is larger than the area under the curve for the online comlog regression and under the curve for the linear SVM. This shows the superiority of our algorithm over these competitors for the classification task. We can also see that there is a point on the curve for the online comlog regression where the reduction of the recall does not lead to the increase in the precision. This means that the threshold becomes larger than the predicted probability of many high-ozone days.

As we can see from the figures in Table 3.3, the algorithms in the online

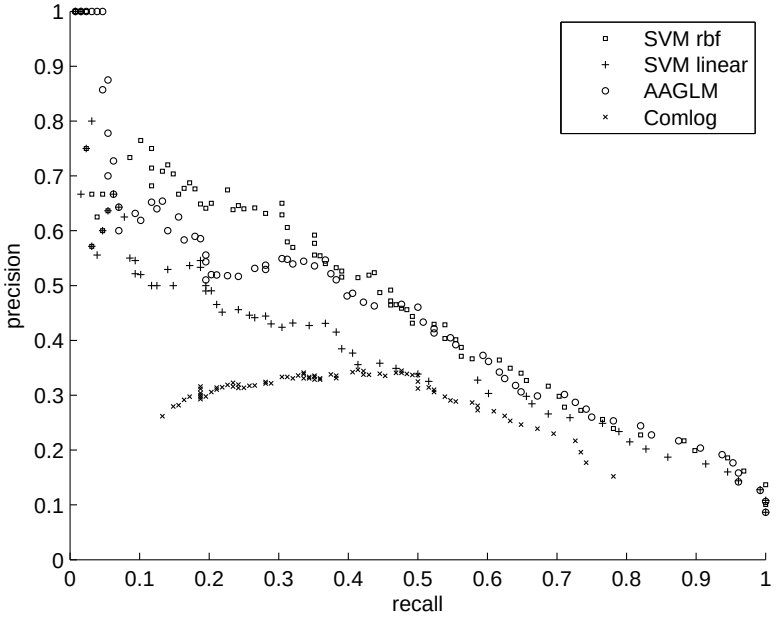


Figure 3.4: Precision-recall curve for different threshold values for the algorithms applied in the online regime on ozone8.

regime perform better than the same algorithms in the batch regime on ozone8 and usually worse on ozone1.

We can also see that the AAGLM significantly outperforms the simple zero predictor and the comlog predictor. Unlike the comlog predictor, the AAGLM is developed to work with the square loss and achieves better empirical performance in terms of the square loss. It performs a little worse on ozone8 than the SVM with the RBF kernel. It is possible to apply kernelization to the AAGLM as well (for the kernelization of standard generalized linear regression models see Cawley et al., 2007), as we will show in Section 4.5; the kernelized algorithm may achieve better performance. On ozone1 the AAGLM outperforms all the algorithms including SVMs.

The disadvantage of our algorithm against the competitors is in its training speed. Increasing the training speed of our algorithm is an interesting area of future research. It would be also interesting to apply our classifier to other data sets and find extreme data sets where the theoretical guarantee is tight.

3.6 Linear probability forecasting

In this section, we introduce a generalization of the Aggregating Algorithm for Regression described in Section 3.2 to the case when the dimension of the outcomes is more than two. Our generalization is intended to deal with probability forecasting, when the outcome and the predictions at each step are probability distributions on a finite number of events. This problem can be thought of as a multi-class classification problem if each outcome is associated with one of the events.

In this section, we consider asymmetric linear experts. They predict a set of numbers; each of these numbers should approximate the corresponding component of the outcome. One component of their prediction has the meaning of a remainder. In practice this situation is quite common. For example, in the previous chapter we considered the problem of predicting the results football matches. In that scenario either one team wins or the other, and the remainder is draw. As another example, we may analyse precious metal alloys and look for a description of the following kind: an alloy has 40% of gold, 35% of silver, and some addition (e.g., copper and palladium). Our algorithm also can be used in financial applications: it is common to try to predict the direction of the price: the price can go up, down, or stay close to the current value.

Kivinen and Warmuth's (2001) work includes the case when possible outcomes lie in a more than 2-dimensional simplex and the algorithms give probability predictions; their algorithm competes with all logistic regression functions. They use the relative entropy loss function $\mathcal{L}$ and obtain a regret term of the order $O(\sqrt{\mathcal{L}_T(\theta)})$ which is upper unbounded in the worst case. They also consider the standard approach to apply online linear regression for each component of the outcome and thus make predictions of multi-dimensional outcomes. The upper bound on the loss is also easy to derive. We consider a slightly different problem: first, we aim to give probability predictions; second, not all of the components of the outcome have the same role, one of them is a remainder. Our algorithms can be easily kernelized (see Section 4.6).

In online convex optimization (see Hazan et al., 2007, for the bounds with the logarithmic regret term) one allows the loss functions to be unknown before

the prediction is made, but requires the learner's prediction to be the weighted average of the expert's predictions. Our way of giving predictions is different. It allows us to give probability predictions in the situation when the experts do not necessarily provide probabilities.

Our setting also slightly relates to the Optimal Portfolio selection problem (Cover, 1991). In this problem predictions come from a probability simplex: they are considered as a capital allocation among a set of assets. There are no outcomes, and the loss function is the minus logarithm of the scalar product between the prediction and the vector of the changes in the prices of the assets.

The upper bounds for classification problems are often proved in terms of the number of mistakes which are made by the learner (see, for example, Cesa-Bianchi et al., 2005). In this situation the learner is forced to give exact predictions about the assignment of the label to the input vector.

The quality of the predictions in our setting is assessed by the Brier loss. We develop two algorithms to solve the problem of multi-dimensional prediction. The first algorithm applies the AAR to predict each coordinate of the outcome separately, and then combines these predictions in a certain way to get probability prediction. The other algorithm is designed to give probability predictions directly. We derive upper bounds on the losses of both algorithms and come to an unexpected conclusion that the component-wise algorithm is better than the second one asymptotically, but worse in the beginning of the prediction process. Their performance on benchmark data sets is very similar. The results of this section are described in Zhdanov and Kalnishkan (2010b) and in more details in Zhdanov and Kalnishkan (2009).

3.6.1 Framework

We are interested in the generalisation of the Brier game (Brier, 1950), which was considered in Section 2.4. The set of outcomes $\Omega = \mathcal{P}(\Sigma)$ is the set of all probability measures on a finite set Σ with d elements, the set of predictions $\Gamma := \{(\gamma^1, \dots, \gamma^d) : \sum_{i=1}^d \gamma^i = 1, \gamma^i \in \mathbb{R}\}$ is a hyperplane in d-dimensional space containing all the outcomes, and the quality of predictions is measured by the Brier loss (2.29). The set of input vectors $\mathbf{X} \subseteq \mathbb{R}^n$ is a subset of the

Euclidean space. The game of prediction follows Protocol 3.

We develop algorithms which are capable of competing with all linear functions $\xi_t = (\xi_t^1, \dots, \xi_t^d)'$ of the input vectors $x_1, x_2, \dots \in \mathbf{X}$:

$$\begin{aligned} \xi_t^1 &= 1/d + (\theta^1)' x_t \\ &\dots \\ \xi_t^{d-1} &= 1/d + (\theta^{d-1})' x_t \\ \xi_t^d &= 1 - \xi^1 - \dots - \xi^{d-1} = 1/d - \left(\sum_{i=1}^{d-1} \theta^i \right)' x_t, \end{aligned} \tag{3.44}$$

where $\theta^i \in \mathbb{R}^n$, $i = 1, \dots, d-1$. In the model (3.44), the last component of the prediction is calculated from the other components. We use the notation $(Y; Z)$ for the vertical concatenation of two column vectors Y and Z. We denote each expert by $\theta = (\theta^1; \dots; \theta^{d-1})$ and the indexing set for the experts by $\Theta = \mathbb{R}^{n(d-1)}$. The prediction of the expert θ can be represented as $\xi_t = \xi_t(\theta)$. By L_T we denote the cumulative loss of the learner at the step T, and by L_T^θ we denote the cumulative loss of the expert θ at this step.

3.6.2 Derivation of the algorithms

In this section we describe how we apply the Aggregating Algorithm (Algorithm 1) to mix experts and make predictions. In order to apply the AA we first need to ensure that the game is mixable, see (2.4).

Proof of mixability

Let us denote the set of d probability measures concentrated in points of Σ by $\mathcal{R}(\Sigma)$. Lemma 2.7 ensures that the Brier game with finite number of outcomes is mixable iff $\eta \in (0, 1]$. This statement is valid for the experts which give predictions lying inside the probability simplex $\mathcal{P}(\Sigma)$ and the outcomes which belong to $\mathcal{R}(\Sigma)$.

We need to prove that there exists a prediction satisfying (2.4) for our experts (3.44) (who can give predictions outside of the probability simplex) and our outcome set Ω (the whole probability simplex, not only its vertices).

Lemma 3.6 describes the first part, but first we need to state an additional statement. The following lemma shows that any vector from $\mathbb{R}^d$ can be projected into simplex without increasing the Brier loss. Recall that $\beta = e^{-\eta}$.

Lemma 3.5 *For any $\xi = (\xi^1, \dots, \xi^d) \in \mathbb{R}^d$ there exists $\zeta = (\zeta^1, \dots, \zeta^d) \in \mathcal{P}(\Sigma)$ such that for any $y \in \Omega$, we have $\lambda(y, \zeta) \le \lambda(y, \xi)$.*

PROOF The Brier loss of a prediction γ is the square Euclidean distance between γ and the actual outcome y. The proof follows from the fact that $\mathcal{P}(\Sigma)$ is a convex and closed subset of $\mathbb{R}^d$. ∎

Lemma 3.6 *Let $P(d\theta)$ be any probability distribution on Θ. Then for any $\eta \in (0, 1]$ and any $\xi(\theta) \in \mathbb{R}^d$ there exists $\gamma \in \Gamma$ such that for any $y \in \mathcal{R}(\Sigma)$, we have*

$$\lambda(y, \gamma) \le \log_\beta \int_\Theta \beta^{\lambda(y, \xi(\theta))} P(d\theta).$$

PROOF By Lemma 3.5 for any experts' prediction $\xi(\theta)$ we can find $\zeta(\theta) \in \mathcal{P}(\Sigma)$ such that its loss does not exceed the loss of the experts: $\lambda(y, \zeta(\theta)) \le \lambda(y, \xi(\theta))$ for any $y \in \mathcal{R}(\Sigma)$. Thus we have

$$\log_\beta \int_\Theta \beta^{\lambda(y, \zeta(\theta))} P(d\theta) \le \log_\beta \int_\Theta \beta^{\lambda(y, \xi(\theta))} P(d\theta)$$

for any $y \in \mathcal{R}(\Sigma)$. We can take the same prediction $\gamma \in \Gamma$ that satisfies the necessary inequality with ζ instead of ξ. By Lemma 2.7 such prediction exists for any $\eta \in (0, 1]$ (or $\beta \in [e^{-1}, 1)$). ∎

We now prove that we can use the same substitution function and the same learning rate parameter η as for the case of finite number of possible outcomes. Such a function is described in Proposition 2.1. This is an extension of Lemma 4.1 from Haussler et al. (1998).

Lemma 3.7 *Let $P(d\theta)$ be a probability distribution on Θ and put*

$$f(y) = \log_\beta \int_\Theta \beta^{\lambda(y, \xi(\theta))} P(d\theta)$$

for every $y \in \Omega$. Then if γ is such a prediction that $\lambda(z,\gamma) \le f(z)$ for any $z \in \mathcal{R}(\Sigma)$ then $\lambda(y,\gamma) \le f(y)$ for any $y \in \Omega$.

PROOF Let us take the unit vector basis $z\{1\},\dots,z\{d\}$ in $\mathbb{R}^d$. It is easy to see that

$$\begin{aligned}
\lambda(y,\xi(\theta)) - \lambda(y,\gamma) &= \sum_{j=1}^{d}(\xi^j(\theta)-y^j)^2 - \sum_{j=1}^{d}(\gamma^j-y^j)^2 \\
&= \sum_{j=1}^{d}(\xi^j(\theta))^2 - (\gamma^j)^2 - 2y^j(\xi^j(\theta)-\gamma^j) \\
&= \sum_{i=1}^{d} y^i \left[\sum_{j=1}^{d}(\xi^j(\theta))^2 - (\gamma^j)^2 - 2z\{i\}^j(\xi^j(\theta)-\gamma^j)\right] \\
&= \sum_{i=1}^{d} y^i \left[\sum_{j=1}^{d}(\xi^j(\theta)-z\{i\}^j)^2 - \sum_{j=1}^{d}(\gamma^j-z\{i\}^j)^2\right] \\
&= \sum_{i=1}^{d} y^i[\lambda(z\{i\},\xi(\theta)) - \lambda(z\{i\},\gamma)]
\end{aligned}$$

for any $y \in \Omega$. We also have that $\lambda(y,\gamma) - f(y) \le 0$ is equivalent to $\int_\Theta \beta^{\lambda(y,\xi(\theta))-\lambda(y,\gamma)}P(d\theta) \le 1$. Due to the convexity of the exponent function and the fact the the inequality holds for any $z \in \mathcal{R}(\Sigma)$, we have

$$\int_\Theta \beta^{\lambda(y,\xi(\theta))-\lambda(y,\gamma)}P(d\theta) = \int_\Gamma \beta^{\sum_{i=1}^{d} y^i[\lambda(z\{i\},\xi(\theta))-\lambda(z\{i\},\gamma)]}P(d\theta) \le \sum_{i=1}^{d} y^i = 1. \quad \blacksquare$$

Algorithm for multidimensional outcomes

Let us denote the ith possible outcome from $\mathcal{R}(\Sigma)$ by $y\{i\}$, $i = 1,\dots,d$. The figure brackets $\{i\}$ will mean that the ith vector out of d vectors is taken. We take the initial weights distribution P_0 over the experts to have the Gaussian density with a parameter $a > 0$:

$$P_0(d\theta) = (a\eta/\pi)^{n(d-1)/2}e^{-a\eta\|\theta\|^2}d\theta.$$

Instead of taking the integral (2.3) to calculate the generalized prediction, we obtain a shifted generalised prediction G by calculating $G_T^i = g_T(y\{i\}) - g_T(y\{d\})$. We use the substitution function from Proposition 2.1, which is irrespective to this shift. Each component of $G_T = (G_T^1, \dots, G_T^d)$ corresponds to one of the possible outcomes from $\mathcal{R}(\Sigma)$ and $G_T^d = 0$. Other components for $i = 1, \dots, d-1$ are expressed as follows:

$$G_T^i = \log_\beta \frac{\beta^{g_T(y\{i\}) + \sum_{t=1}^{T-1} g_t(y_t)}}{\beta^{g_T(y\{d\}) + \sum_{t=1}^{T-1} g_t(y_t)}} = -\frac{1}{\eta} \ln \frac{\int_\Theta e^{-\eta Q(\theta, y\{i\})} d\theta}{\int_\Theta e^{-\eta Q(\theta, y\{d\})} d\theta},$$

where by $Q(\theta, y)$ we denote the quadratic form:

$$\begin{aligned} Q(\theta, y) &= L_{T-1}^\theta + \lambda(y, \xi_T(\theta)) + a\|\theta\|^2 \\ &= \sum_{t=1}^{T-1} \sum_{i=1}^{d} (\xi_t^i(\theta) - y_t^i)^2 + \sum_{i=1}^{d} (\xi_T^i(\theta) - y^i)^2 + a\|\theta\|^2. \quad (3.45) \end{aligned}$$

Here $y_t = (y_t^1, \dots, y_t^d)$ are the actual outcomes on the steps before T and $y = (y^1, \dots, y^d) \in \Omega$ is the possible outcome at the step T.

Let C_T be the $n \times n$ matrix $C_T = \sum_{t=1}^{T} x_t x_t'$. The quadratic form $Q(\theta, y)$ can be separated into a quadratic part, a linear part, and a remainder: $Q(\theta, y) = Q_1(\theta, y) + Q_2(\theta, y) + Q_3(\theta, y)$. We have that

$$\begin{aligned} Q_1(\theta, y) &= \sum_{t=1}^{T} \left(\sum_{i=1}^{d-1} (\theta^i)' x_t x_t' \theta^i + \left(\sum_{i=1}^{d-1} \theta^i \right)' x_t x_t' \left(\sum_{i=1}^{d-1} \theta^i \right) \right) + a\|\theta\|^2 \\ &= \theta' \left(aI + \begin{pmatrix} 2C_T & \cdots & C_T \\ \vdots & \ddots & \vdots \\ C_T & \cdots & 2C_T \end{pmatrix} \right) \theta = \theta' A_T \theta \end{aligned}$$

is the quadratic part of $Q(\theta, y)$. Here A_T is a square matrix with $n(d-1)$ rows. The linear part is equal to

$$Q_2(\theta, y) = \theta' h_T - 2 \sum_{i=1}^{d-1} (y^i - y^d)(\theta^i)' x_T,$$

where $h_T^i = -2\sum_{t=1}^{T-1}(y_t^i - y_t^d)x_t$, $i = 1, \ldots, d-1$, make up a big vector $h_T = (h_T^1; \ldots; h_T^{d-1}) \in \mathbb{R}^{n(d-1)}$. The remainder is equal to

$$Q_3(\theta, y) = \sum_{t=1}^{T-1}\sum_{i=1}^{d}(y_t^i - 1/d)^2 + \sum_{i=1}^{d}(y^i - 1/d)^2.$$

Therefore G_T^i can be calculated using Lemma A.3 as follows:

$$G_T^i = -(b_T\{i\})' A_T^{-1} z_T\{i\} \tag{3.46}$$

for $i = 1, \ldots, d-1$ and $G_T^d = 0$. Here we have

$$b_T\{i\} = h_{T-1} + (x_T; \ldots; x_T; \mathbf{0}; x_T; \ldots; x_T) \in \mathbb{R}^{n(d-1)},$$
$$z_T\{i\} = (-x_T; \ldots; -x_T; -2x_T; -x_T; \ldots; -x_T),$$

where the zero-vector $\mathbf{0} \in \mathbb{R}^n$ and $-2x_T$ are placed in the i-th blocks. It is now possible to apply the substitution function from Proposition 2.1 to get predictions. We call our algorithm the mAAR (the multi-dimensional Aggregating Algorithm for Regression). It is described as Algorithm 10.

Component-wise algorithm

In this section we describe the component-wise algorithm. It gives predictions for each component of the outcome separately, and then combines them in a special way to give probability prediction.

We consider a different (more wide) class of experts than (3.44): the components of their predictions have symmetry. Each expert $\theta \in \mathbb{R}^{nd}$ predicts

$$\xi_t^i = 1/d + (\theta^i)' x_t, \quad i = 1, \ldots, d, \tag{3.47}$$

i.e. they use $c = 1/d$ in (3.2) for each component of their prediction.

The component-wise Aggregating Algorithm for Regression (cAAR) calcu-

lates its preliminary prediction by the formula (3.4):

$$\gamma_T^i = \frac{1}{2} + \left(\sum_{t=1}^{T-1} y_t^i x_t + \frac{d-2}{2d} x_T \right)' \left(aI + \sum_{t=1}^{T} x_t x_t' \right)^{-1} x_T, \quad i = 1, \dots, d.$$

In this formula the outcomes y_t^i are taken to be equal to the ith components of the real outcomes. The algorithm then projects the prediction vector onto the prediction simplex such that the loss does not increase. We use the projection algorithm suggested by Michelot (1986) and described as Algorithm 11.

Algorithm 10 The mAAR for the Brier game

Require: $a > 0$.

Initialize $h_0^i = \mathbf{0} \in \mathbb{R}^n$, $i = 1, \dots, d-1$, and $C_0 = \mathbf{0} \in \mathbb{R}^{n \times n}$.

Set $h_0 = (h_0^1; \dots; h_0^{d-1}) \in \mathbb{R}^{n(d-1)}$.

for $t = 1, 2, \dots$ **do**

Read $x_t \in \mathbf{X}$.

Update $C_t = C_{t-1} + x_t x_t'$, $A_t = aI + \begin{pmatrix} 2C_t & \cdots & C_t \\ \vdots & \ddots & \vdots \\ C_t & \cdots & 2C_t \end{pmatrix}$.

Set $b_t\{i\} = h_{t-1} + (x_t; \dots; x_t; \mathbf{0}; x_t; \dots; x_t)$, where $\mathbf{0}$ is a zero-vector from $\mathbb{R}^n$ placed at the ith position, $i = 1, \dots, d-1$.

Set $z_t\{i\} = (-x_t; \dots; -x_t; -2x_t; -x_t; \dots; -x_t)$, where $-2x_t'$ is placed at the ith position, $i = 1, \dots, d-1$.

Calculate $G^i = -(b_t\{i\})' A_t^{-1} z_t\{i\}$, $G^d = 0$, $i = 1, \dots, d-1$.

Solve $\sum_{i=1}^d (s - G^i)^+ = 2$ in $s \in \mathbb{R}$.

Predict $\gamma_t \in \mathcal{P}(\Omega)$ with $\gamma_t^i = (s - G^i)^+ / 2$, $i = 1, \dots, d$.

Read y_t.

Update $h_t^i = h_{t-1}^i - 2(y_t^i - y_t^d) x_t$, $h_t = (h_t^1; \dots; h_t^{d-1})$.

end for

Algorithm 11 Projection of a point from $\mathbb{R}^d$ onto probability simplex.

Initialize $I = \emptyset$, $x = \mathbf{1} \in \mathbb{R}^d$.
Let γ_T be the prediction vector and $|I|$ be the dimension of the set I.
while 1 **do**
  $\gamma_T := \gamma_T - \frac{\sum_{i=1}^{d} \gamma_T^i - 1}{d - |I|}$.
  $\gamma_T^i := 0$ for all $i \in I$.
  if $\gamma_T^i \geq 0$ for all $i = 1, \ldots, d$ **then**
    break.
  end if
  $I := I \bigcup \{i : \gamma_T^i < 0\}$.
  if $\gamma_T^i < 0$ for some i **then**
    $\gamma_T^i := 0$.
  end if
end while

3.6.3 Performance guarantees

We derive upper bounds on the cumulative loss of the mAAR and of the cAAR predicting in the same framework.

Loss bound for the cAAR

The loss of the component-wise algorithm by one component is bounded as in Theorem 3.1. When we use the component-wise algorithm to predict each component separately, its Brier loss is bounded as in the following theorem.

Theorem 3.6 *If $\|x_t\|_\infty \leq X$ for all t then for any $a > 0$, every positive integer T, every sequence of outcomes of the length T, and any $\theta \in \mathbb{R}^{n(d-1)}$, the loss L_T of the cAAR with the parameter a satisfies*

$$L_T \leq L_T^\theta + da\|\theta\|^2 + \frac{nd}{4} \ln\left(1 + \frac{TX^2}{a}\right). \tag{3.48}$$

Proof Let us first consider the experts (3.47) and sum the upper bounds (3.9) with $Y_2 = 1$, $Y_1 = 0$ for each of the components of the outcome. In the last

component we can take $\theta^d = -\sum_{i=1}^{d-1} \theta^i$ instead of any expert and use the Cauchy inequality

$$\left\| \sum_{i=1}^{d-1} \theta^i \right\|^2 \le (d-1) \sum_{i=1}^{d-1} \|\theta^i\|^2$$

to compete with the experts (3.44). Thus we obtain the right-hand side of (3.48). The probability prediction of the cAAR is given by Algorithm 11 such that the Brier loss does not increase (see Michelot, 1986, Theorem 3.1). This completes the proof. ∎

Loss bound for the mAAR

The upper bound for the loss of Algorithm 10 is given by the following theorem.

Theorem 3.7 *For any $a > 0$, every positive integer T, every sequence of outcomes of the length T, and any $\theta \in \mathbb{R}^{n(d-1)}$, the cumulative loss L_T of the mAAR with the parameter $2a$ satisfies*

$$L_T \le L_T^\theta + 2a\|\theta\|^2 + \frac{1}{2} \ln \det \left(I + \frac{1}{2a} \begin{pmatrix} 2C_T & \cdots & C_T \\ \vdots & \ddots & \vdots \\ C_T & \cdots & 2C_T \end{pmatrix} \right). \tag{3.49}$$

If, in addition, $\|x_t\|_\infty \le X$ for all t, then

$$L_T \le L_T^\theta + 2a\|\theta\|^2 + \frac{n(d-1)}{2} \ln \left(1 + \frac{TX^2}{a} \right). \tag{3.50}$$

Proof Let by θ_0 denote the best expert: $\theta_0 = \arg\min_{\theta \in \Theta} \left(L_T^\theta + 2a\|\theta\|^2 \right)$. The regularized cumulative loss of the expert θ at the step T can be expressed using the decomposition (3.45) as follows:

$$\sum_{t=1}^{T} \sum_{i=1}^{d} (\xi_t^i(\theta) - y_t^i)^2 + 2a\|\theta\|^2$$

$$= \sum_{t=1}^{T} \sum_{i=1}^{d} (\xi_t^i(\theta_0) - y_t^i)^2 + 2a\|\theta_0\|^2 + (\theta - \theta_0)' A_T (\theta - \theta_0)$$

for $A_T = 2aI + \begin{pmatrix} 2C_T & \cdots & C_T \\ \vdots & \ddots & \vdots \\ C_T & \cdots & 2C_T \end{pmatrix}$. By Lemma 3.2, we obtain for any $\theta \in \Theta$

$$L_T \leq L_T^\theta + 2a\|\theta\|^2 + \frac{1}{2\eta} \ln \det \left(\frac{A_T}{2a} \right).$$

We take the maximum value for η, $\eta = 1$. The determinant of a symmetric positive definite matrix is upper bounded by the product of its diagonal elements (see Beckenbach and Bellman, 1961, Chapter 2, Theorem 7): $\det \left(\frac{A_T}{2a} \right) \leq \left(1 + \frac{TX^2}{a} \right)^{n(d-1)}$. This completes the proof. ∎

We can derive a slightly better upper bound: in the determinant of A_T one should subtract the second block row from the first one and then add the first block column to the second one, then repeat this $d-2$ times with other rows and columns.

Proposition 3.1 *In the conditions of Theorem 3.7, the loss L_T of the mAAR with the parameter a satisfies*

$$L_T \leq L_T^\theta + a\|\theta\|_2^2 + \frac{n(d-2)}{2} \ln \left(1 + \frac{TX^2}{a} \right) + \frac{n}{2} \ln \left(1 + \frac{TX^2 d}{a} \right) \qquad (3.51)$$

for any $\theta \in \mathbb{R}^{n(d-1)}$.

The upper bound (3.50) is worse asymptotically in d than the bound (3.48) of the component-wise algorithm, but it is better in the beginning, especially when the norm of the best expert $\|\theta\|$ is large. This can happen in the important case when the dimension of the input vectors is larger than the size of the prediction set: $n >> T$.

Using methods similar to the ones described by Vovk (2001), it is possible to prove lower bounds for the regret term of the order $O\left(\frac{d-1}{d} \ln T\right)$ for the case of the linear model (3.44). We can say that the order of our upper bounds is optimal. Multiplicative constants may possibly be improved though.

3.6.4 Experiments

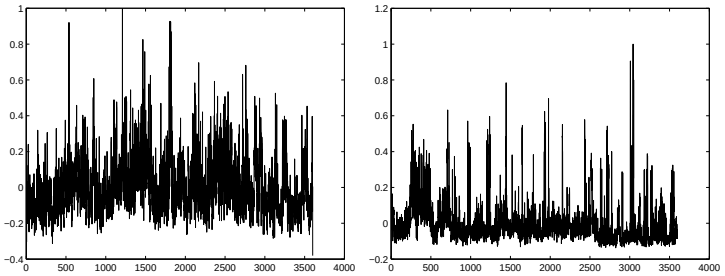
We run our algorithms on six real world time series data sets. In the time series, we consider there are no input vectors attached to the outcomes. However, we can take the vectors consisting of previous observations (we shall take ten of those) and use them as the input vectors. Data set DEC-PKT[4] contains an hours worth of all wide-area traffic between Digital Equipment Corporation and the rest of the world. Data set LBL-PKT-4[4] consists of observations of another hour of traffic between the Lawrence Berkeley Laboratory and the rest of the world. We transformed both the data sets in such a way that each observation is the number of packets in the corresponding network during a fixed time interval of one second. The other four datasets[5] (C4,C9,E5,E8) relate to transportation data. Two of them (C9,C11) contain low-frequency monthly traffic measures. Two of them (E5,E8) contain high-frequency day traffic measures. On each of these data sets the following operations were performed: subtraction of the mean value and division by the maximum absolute value. The resulting time series are shown in Figure 3.5.

We use ten previous observations as an input vector for tested algorithms at each prediction step. We consider the following 3-class classification problem. We predict whether the next value in a time series will be more than the previous value plus a precision parameter ϵ, less than that value, or will lie in the 2ϵ tube around the previous value. The precision ϵ for each time seres is chosen to be the median of all the changes in them. In order to assess the quality of predictions, we calculate the cumulative square loss at the last two thirds of each time series (test set) and divide it by the number of examples (MSE). Since we are considering the online setting, we could calculate the cumulative loss from the beginning of each time series. However, our approach is not sensitive to starting effects. It allows us to choose the ridge parameter a fairly on the training set. It also allows us to compare the performance of our algorithms with batch algorithms, which would be normally used to solve this problem.

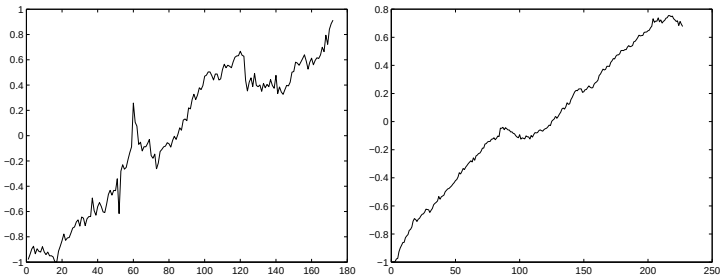
The square loss on the test set takes into account the quality of an algorithm

[4]Data sets can be found http://ita.ee.lbl.gov/html/traces.html.

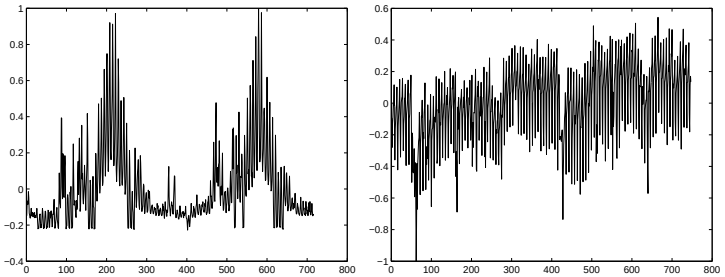
[5]Data sets can be found http://www.neural-forecasting-competition.com/index.htm.



(a) DEC-PKT series (b) LBL-PKT series



(c) C4 series (d) C9 series



(e) E5 series (f) E8 series

Figure 3.5: Time series from 6 data sets.

only at the very end of the prediction process, and does not look at the quality during the process. We introduce another quality measure: at each step in the test set we calculate the MSE of an algorithm until this step. After all the steps, we average these MSEs and call the average AMSE. Clearly, if one algorithm is better than another on the whole test set (its total MSE is smaller) but was often worse on many parts of the test set (total MSEs of many parts of the set is larger), this measure takes it into account.

We compare the performance of our algorithms with the multinomial logistic regression (mLog), because it is a standard classification algorithm which

gives probability predictions:

$$\gamma^i_{\text{mLog}} = \frac{e^{\theta^i x}}{\sum_{i=1}^d e^{\theta^i x}}$$

for all the components of the outcome $i = 1, \dots, d$. In our case $d = 3$. Here parameters $\theta^1, \dots, \theta^d$ are estimated from the training set. We apply this algorithm in two regimes: batch regime, where the algorithm learns only on the training set and is tested on the test set (and thus θ is not updated on the test set); and in the online regime, where at each step new parameters θ are found, and only one next outcome is predicted. The second regime is more fair to compare with online algorithms, but the first one is standard and faster. In both regimes logistic regression does not have theoretical guarantees on the square loss.

We also compare our algorithms with the simple predictor predicting the average of the ten previous outcomes (it thus always gives probability predictions).

We are not aware of other efficient algorithms for online probability prediction, and thus use the logistic regression and simple predictor as the only baselines. Component-wise algorithms which could be used for online prediction (e.g., Gradient Descent, Kivinen and Warmuth 1997, Ridge Regression, Hoerl and Kennard 2000) have to use the normalization by Algorithm 11. Thus they have to be applied in a different way than they are described in the corresponding papers, and cannot be fairly compared with our algorithms.

The ridge for our algorithms is chosen to achieve the best MSE on the training set: the first third of each series. The results are shown in Table 3.4. In this table cAAR and mAAR state for the derived algorithms, mLog states for the logistic regression, mLogOnline states for the online logistic regression, and Simple stands for the simple average predictor. We highlight the most precise algorithms for different data sets. We also show the time needed to make predictions on the whole data set. The algorithms were implemented in Matlab R2007b and run on the laptop with 2Gb RAM and processor Intel Core 2, T7200, 2.00GHz.

As we can see from the table, our online methods 4 out of 6 times perform

better than the batch method. Online logistic regression performs well, but it is very slow. Our algorithms perform similar to each other and comparable to the online logistic regression, but they are much faster.

Table 3.4: The square losses and prediction time (sec) of different algorithms.

Set	Algorithm	MSE	AMSE	Time
DEC-PKT	cAAR	0.45906	0.45822	0.578
	mAAR	0.45906	0.45822	1.25
	mLog	0.46107	0.46265	0.375
	mLog Online	**0.45751**	0.45762	2040.141
	Simple	0.58089	0.57883	0
LBL-PKT	cAAR	0.48147	0.479	0.579
	mAAR	0.48147	0.479	1.266
	mLog	0.47749	0.47482	0.391
	mLog Online	**0.47598**	0.47398	2403.562
	Simple	0.57087	0.5657	0.016
C4	cAAR	0.64834	0.65447	0.015
	mAAR	**0.64538**	0.65312	0.062
	mLog	0.76849	0.77797	0.016
	mLog Online	0.68164	0.7351	4.328
	Simple	0.69037	0.69813	0.016
C9	cAAR	**0.63238**	0.64082	0.015
	mAAR	0.63338	0.64055	0.063
	mLog	0.97718	0.91654	0.031
	mLog Online	0.71178	0.75558	10.625
	Simple	0.6509	0.65348	0
E5	cAAR	0.34452	0.34252	0.078
	mAAR	0.34453	0.34252	0.219
	mLog	0.31038	0.30737	1.109
	mLog Online	**0.30646**	0.30575	446.578
	Simple	0.58212	0.58225	0
E8	cAAR	0.29395	0.29276	0.078
	mAAR	0.29374	0.29223	0.25
	mLog	0.31316	0.30382	0.109
	mLog Online	**0.27982**	0.27068	83.125
	Simple	0.69691	0.70527	0.016

Chapter 4

Online regression in Hilbert spaces

In this chapter we describe online regression framework in Hilbert spaces. The well-known in machine learning kernel trick applied to methods for linear regression allows us to develop methods working with much wider classes of regression functions: the classes of functions from Reproducing Kernel Hilbert Spaces.

Even though the theory of Reproducing Kernel Hilbert Spaces is known for mathematicians for a relatively long time (Aronszajn, 1950), its popularity in machine learning (following work of Aizerman et al., 1964) is probably due to the development of statistical learning theory (Vapnik, 1995) and the applications of Support Vector learning (Schölkopf and Smola, 2002). Bayesian analysis often regards kernel methods from the perspective of Gaussian non-parametric methods (Rasmussen and Williams, 2006).

The first analysis of an online prediction algorithm which is competitive with a large class of functions is due to Cesa-Bianchi et al. (1996).

4.1 Online regression in Hilbert spaces

Online regression in Hilbert spaces follows the same Protocol 3 which we considered earlier for online linear regression. The difference lies in the class of experts with which the learner tries to compete.

If the set of input vectors lies in a Euclidean space, $\mathbf{X} \subseteq \mathbb{R}^n$, then we can consider simple linear predictors of the form $\theta \in \mathbb{R}^n$ which given a signal $x \in \mathbf{X}$ make predictions $\theta' x$. This approach was used in the previous chapter. The use of the linear methods in the real world is limited though, because they can only model simple dependencies. Even though generalized linear models, which we considered in Section 3.5, work with non-linear experts, it is still a narrow class. One solution to this problem could be to map the data to some high dimensional feature space and then find a simple solution there. Some of the implementations of this idea can lead to the curse of dimensionality: the situation where both the computational and generalisation performance degrade as the number of features grow. On the contrary, kernel methods can be used to make a linear algorithm operate in feature space without the inherent complexities.

Definition 4.1 (Kernel as a Dot Product in Feature Space) Given a mapping $\phi : \mathbf{X} \mapsto \mathcal{H}$, where $\mathcal{H}$ is a Hilbert space, *kernel* is defined as a function $k : \mathbf{X} \times \mathbf{X} \to \mathbb{R}$ such that

$$k(x, y) = \langle \phi(x), \phi(y) \rangle_{\mathcal{H}}.$$

Definition 4.2 (Reproducing Kernel Hilbert Space (RKHS)) A *Reproducing Kernel Hilbert Space (RKHS)* on a set $\mathbf{X}$ is a Hilbert space $\mathcal{F}$ of real valued functions on X such that the evaluation functional $f \in \mathcal{F} \mapsto f(x)$ is continuous for each $x \in X$.

By the Riesz Representation theorem, for every $x \in \mathbf{X}$ there exists a function $k_x \in \mathcal{F}$ such that

$$f(x) = \langle k_x, f \rangle$$

for all $f \in \mathcal{F}$. The *reproducing kernel* of $\mathcal{F}$ is the function $k : \mathbf{X} \times \mathbf{X} \mapsto \mathbb{R}$ such that

$$k(x, y) = \langle k_x, k_y \rangle = k_x(y) = k_y(x).$$

The continuity requirement is essential for regression tasks: it ties the norm of the functions from a functional Hilbert space with their predictions. If two functions are close w.r.t. norm, they should give similar predictions.

An equivalent definition of the kernel will also be useful in many derivations.

Definition 4.3 (Kernel as a Symmetric Positive Semi-Definite Matrix) A *kernel* is any function $k : \mathbf{X} \times \mathbf{X} \mapsto \mathbb{R}$ that is symmetric

$$k(x, y) = k(y, x)$$

for all $y, x \in \mathbf{X}$, and positive semi-definite

$$\sum_{i,j=1}^{\ell} c_i c_j k(x_i, x_j) \geq 0$$

for all $\ell \geq 1$, all $c_i, c_j \in \mathbb{R}$, and all $x_i, x_j \in \mathbf{X}$.

These three definitions are equivalent since a function $k(x, y) : \mathbf{X} \times \mathbf{X} \mapsto \mathbb{R}$ can be represented in the form $\langle \phi(x), \phi(y) \rangle$ iff k is the reproducing kernel of an RKHS iff k is symmetric and positive semi-definite. For every kernel there exists a unique RKHS $\mathcal{F}$ such that k is the reproducing kernel of $\mathcal{F}$. For more information on kernels and RKHS see, for example, Aronszajn (1950) and Schölkopf and Smola (2002, Chapter 2).

In order to kernelize a linear algorithm, one usually formulates 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*. As we will see below, it is sometimes necessary to be able to derive the kernelized algorithm from the linear one without using the kernel trick.

Standard Kernels

There are several kernels which are generally advised to use for practical purposes if the specific form is not known.

The Linear Kernel The linear kernel (dot product) is the simplest of kernels and is used in linear algorithms

$$k(x, y) = \langle x, y \rangle.$$

Clearly, the mapping used is the identity function, therefore the input set lies directly in the feature space.

The Polynomial Kernel The polynomial kernel is a generalization of the linear kernel given by

$$k(x, y) = (1 + \langle x, y \rangle)^d.$$

This kernel maps the elements of the vectors into the space spanned by all their monomials (products of features) up to and including the dth degree.

The Radial Basis Function Kernel The radial basis function (RBF) kernel is calculated by

$$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right).$$

RBF kernel maps the input set onto the surface of an infinite dimensional unit hypersphere, because by construction $\|\phi(x)\| = \sqrt{k(x, x)} = 1$ for all $x \in X$. The parameter σ controls the amount of smoothing of the decision surface in the space with input vectors. Large σ leads to a smooth surface, small σ leads to a more flexible surface.

4.2 Kernelized Aggregating Algorithm for Regression

In this section we present the Kernelized Aggregating Algorithm for Regression (KAAR) described by Gammerman et al. (2004). It competes with all functions from the RKHS given by a kernel for the case when the outcome set Ω is a bounded interval $[Y_1, Y_2]$, $Y_1 < Y_2$.

Kivinen and Warmuth (2004) consider online learning in RKHS using Gradient Descent, and also derive mistake bounds for an online classification algorithm.

4.2.1 Derivation of the algorithm

Let the outcome set Ω be the bounded interval $[Y_1, Y_2]$, the prediction set Γ be the real line $\mathbb{R}$, and the loss function be the square loss (2.24): $\lambda(y, \gamma) = (\gamma - y)^2$ with $y \in \Omega$, $\gamma \in \Gamma$. Let also the set $\mathbf{X}$ of input vectors be a subset of $\mathbb{R}^n$, $\mathbf{X} \subseteq \mathbb{R}^n$. Each expert predicts according to a function $f \in \mathcal{F}$ at the step t:

$$\xi_t^\theta = c + f(x_t) \tag{4.1}$$

for some $c \in \mathbb{R}$, where $\mathcal{F}$ is the Reproducing Kernel Hilbert Space corresponding to a kernel function $k(x, z)$, $x, z \in \mathbf{X}$. We say that $\mathcal{F}$ indexes experts, and each expert is associated with a function $f \in \mathcal{F}$. Here c in the experts may be needed because we usually take the distribution over the experts centred at 0, and c shifts the centre.

We apply the Aggregating Algorithm for Regression to compete with these experts. Gammerman et al. (2004) first applied it for the case $Y_1 = -Y$, $Y_2 = Y$, $Y > 0$, symmetric experts with $c = 0$, and separable RKHS. We first need to rewrite the formula (3.4) for the predictions of the AAR in the form that involves only dot products.

We introduce some notation. Let

$$\begin{array}{lll} k(x,y) & \text{be} & \text{the given kernel function,} \\ \mathbf{K}_T & \text{be} & \text{the matrix of kernel values, } \mathbf{K}_T = \{k(x_i,x_j)\}_{i,j=1}^T, \\ \mathbf{k}_T & \text{be} & \text{the last column of this matrix, } \mathbf{k}_T = \{k(x_i,x_T)\}_{i=1}^T, \\ \mathbf{Y}_T & \text{be} & \text{the column vector of outcomes, } \mathbf{Y}_T = (y_1,\dots,y_T)'. \end{array} \tag{4.2}$$

Let also $\mathbf{Y}_T - c$ be the difference between the vector $\mathbf{Y}_T$ and the column vector of constant values c of the same length. When we write $\mathbf{Z} = (\mathbf{V};\mathbf{Y})$ ($\mathbf{Z} = (\mathbf{V}',\mathbf{Y}')$) we mean that the column (row) vector $\mathbf{Z}$ is obtained by concatenating two column vectors $\mathbf{V},\mathbf{Y}$ vertically (row vectors $\mathbf{V}',\mathbf{Y}'$ horizontally).

Proposition 4.1 *Predictions* (3.4) *of the AAR can be represented as*

$$\gamma_T = c + \left(\mathbf{Y}_{T-1} - c; \frac{Y_2+Y_1}{2} - c\right)' (aI + \mathbf{K}_T)^{-1}\mathbf{k}_T \tag{4.3}$$

for the linear kernel $k(x,y) = \langle x,y\rangle$, the unit $T\times T$ matrix I, and $a > 0$.

Proof Let by X denote the design matrix $T\times n$ consisting of the rows of the input vectors $x_1',\dots,x_T'$. Then we have $aI + \mathbf{K}_T = aI + XX'$ and $\mathbf{k}_T = Xx_T$. By Lemma A.4 we obtain $(aI+XX')^{-1}Xx_T = X(aI+X'X)^{-1}x_T$. It is easy to see that

$$\left(\mathbf{Y}_{T-1} - c; \frac{Y_2+Y_1}{2} - c\right)' X = \left(\sum_{t=1}^{T-1} x_t(y_t - c) + \left(\frac{Y_2+Y_1}{2} - c\right)x_T\right)',$$

and thus we obtain formula (3.4) from (4.3). ∎

Following this proposition the KAAR can use other kernel functions to give its predictions. In the next section we prove that if another kernel function is used, the KAAR is competitive with all the functions from the RKHS corresponding to the kernel.

4.2.2 Performance guarantee

Let $\mathcal{F}$ be the RKHS corresponding to the kernel $k(x,y)$. The following lemma presents a general approach used to prove theoretical guarantees for kernelized

algorithms.

Let predicting algorithm $\mathcal{A}$ be such that it uses input vectors only in the form of dot products. We say that $\gamma_t^{\mathcal{A}}$ are the predictions of $\mathcal{A}$, and $\gamma_t^{\mathcal{K}\mathcal{A}}$ are the predictions of $\mathcal{A}$ obtained by replacing dot products $\langle x_i, x_j \rangle$ of the input vectors by the values of the kernel $k(x_i, x_j)$.

Let a theoretical guarantee for the algorithm $\mathcal{A}$ at the step T depend on the predictions $\gamma_t^{\mathcal{A}}$, the actual outcomes y_t, the values $\theta' x_t$ for an expert $\theta \in \mathbb{R}^n$, its complexity $\|\theta\|_2$, parameters of $\mathcal{A}$, and the dot products of the input vectors $\langle x_i, x_j \rangle$, $i, j = 1, \ldots, T$. Then we say that its *dual form* at the step T depends in the same way on the predictions $\gamma_t^{\mathcal{K}\mathcal{A}}$ the actual outcomes y_t, the values $\sum_{i=1}^T c_i k(x_t, x_i)$ for an expert $c \in \mathbb{R}^T$, its complexity $\left\| \sum_{i=1}^T c_i k(\cdot, x_i) \right\|_{\mathcal{F}}$, the same parameters of $\mathcal{A}$, and the kernel values $k(x_i, x_j)$, respectively.

Lemma 4.1 *Take algorithm $\mathcal{A}$ predicting in Protocol 3. If there is a theoretical guarantee for the algorithm $\mathcal{A}$, then the dual form of the guarantee holds even if any other kernel than the dot product is used.*

Proof It suffices to prove that for each $T \in \{1, 2, \ldots\}$ and each sequence $(x_1, y_1, \ldots, x_T, y_T) \in (\mathbf{X} \times \mathbb{R})^T$, the dual form of the guarantee is valid. Fix such T and $(x_1, y_1, \ldots, x_T, y_T)$. Fix an isomorphism between the linear span of $k_{x_1}, \ldots, k_{x_T}$ obtained from the Riesz Representation theorem, and $\mathbb{R}^{\tilde{T}}$, where $\tilde{T} \le T$ is the dimension of the linear span of $k_{x_1}, \ldots, k_{x_T}$. Let $\tilde{x}_1, \ldots, \tilde{x}_T \in \mathbb{R}^{\tilde{T}}$ be the images of $k_{x_1}, \ldots, k_{x_T}$, respectively, under this isomorphism. Then $k(\cdot, x_i) = \langle \cdot, \tilde{x}_i \rangle$, thus the kernels in the dual guarantee transform to the dot products. The values $\sum_{i=1}^T c_i k(x_j, x_i)$, $j = 1, \ldots, T$, for any expert $c \in \mathbb{R}^T$ are equal to the values $\langle \tilde{x}_j, \theta \rangle$ for the expert $\theta = \sum_{i=1}^T c_i \tilde{x}_i \in \mathbb{R}^{\tilde{T}}$. The complexities of the experts are equal as well due to the isomorphism:

$$\|\theta\|_2^2 = \left\| \sum_{i=1}^T c_i \tilde{x}_i \right\|_{\mathbb{R}^{\tilde{T}}}^2 = \left\| \sum_{i=1}^T c_i k_{x_i} \right\|_{\mathcal{F}}^2 .$$

The algorithm $\mathcal{A}$ applied to the transformed input vectors $\tilde{x}_1, \ldots, \tilde{x}_T$ is equivalent to the initial algorithm applied to the input vectors $x_1, \ldots, x_T$ (with the use of kernels), thus they give the same predictions. They have the same theoretical guarantees on their losses, thus the guarantee with kernels is valid. ■

The following theorem plays an important role in machine learning. It shows that given a finite set of pairs $(x_1, y_1), \dots, (x_T, y_T)$, the best predictor among all the functions from the given RKHS can be represented as a finite linear combination of the kernel functions based on the input vectors.

Theorem 4.1 (Representer theorem; see Schölkopf and Smola, 2002, Theorem 4.2) *Denote by $g : [0, \infty) \to \mathbb{R}$ a strictly monotonic increasing function. Assume $\mathbf{X}$ is an arbitrary set, and $\mathcal{F}$ is a Reproducing Kernel Hilbert Space of functions on $\mathbf{X}$ with the given kernel $k : \mathbf{X}^2 \to \mathbb{R}$. Assume we also have a positive integer T and an arbitrary loss function $c : (\mathbf{X} \times \mathbb{R}^2)^T \to \mathbb{R} \bigcup \{\infty\}$. Then each minimizer $f \in \mathcal{F}$ of*

$$c\left((x_1, y_1, f(x_1)), \dots, (x_T, y_T, f(x_T))\right) + g(\|f\|_{\mathcal{F}})$$

admits a representation of the form

$$f(x) = \sum_{i=1}^{T} \alpha_i k(x, x_i)$$

for any $x \in \mathbf{X}$ and real numbers $\alpha_i, i = 1, \dots, T$.

By L_T^f denote the cumulative loss of the expert $f \in \mathcal{F}$: $L_T^f = \sum_{t=1}^{T} (f(x_t) - y_t)^2$. The KAAR has the following upper bound on its cumulative square loss.

Theorem 4.2 *For any $a > 0$, every positive integer T, every sequence of outcomes of the length T, and any $f \in \mathcal{F}$, the cumulative square loss L_T of the KAAR with the parameter a satisfies*

$$L_T \le L_T^f + a\|f\|^2 + \frac{(Y_2 - Y_1)^2}{4} \ln \det \left(I + \frac{1}{a} \mathbf{K}_T \right). \tag{4.4}$$

Proof Following Lemma 4.1 we first need to prove that the upper bound (3.8) for the AAR can be expressed in such a way that all the input vectors appear only in dot products. Let us take the dot product kernel $k(x_i, x_j) = \langle x_i, x_j \rangle$.

Then the predictions of the AAR are equal to the predictions of the KAAR for all $t = 1, \dots, T$ by Proposition 4.1, and thus the left-hand sides of (3.8)

and (4.4) are the same. The regret terms

$$\frac{(Y_2 - Y_1)^2}{4} \ln \det \left(I + \frac{1}{a} \mathbf{K}_T \right) = \frac{(Y_2 - Y_1)^2}{4} \ln \det \left(I + \frac{1}{a} \sum_{t=1}^{T} x_t x_t' \right)$$

are the same due to Lemma A.5. The loss of any expert is upper bounded by the loss of the best expert achieving the minimal loss. The regularized cumulative losses of the best experts are the same by the Representer Theorem. Thus we can apply Lemma 4.1 to obtain the upper bound (4.4) for any kernel other than the dot product. ■

The order of the regret term in (4.4) is not clear on the face of it. We show that it has the order $O(\sqrt{T})$ in many cases. We will use the notation $c_{\mathcal{F}}^2 = \sup_{x \in \mathbf{X}} k(x, x)$ and assume $c_{\mathcal{F}}^2 < \infty$.

Corollary 4.1 *In the conditions of Theorem 4.2 and if the number of steps T is known in advance, the KAAR can achieve*

$$L_T \le L_T^f + \left(\|f\|^2 + \frac{(Y_2 - Y_1)^2}{4} \right) c_{\mathcal{F}} \sqrt{T} \tag{4.5}$$

for any $f \in \mathcal{F}$.

Proof The determinant of a symmetric positive definite matrix is upper bounded by the product of its diagonal elements (see Beckenbach and Bellman, 1961, Chapter 2, Theorem 7) and thus

$$\ln \det \left(I + \frac{1}{a} \mathbf{K}_T \right) \le T \ln \left(1 + \frac{c_{\mathcal{F}}^2}{a} \right) \le T \frac{c_{\mathcal{F}}^2}{a}$$

If we know the number of steps T in advance, then we can choose a specific value for a; let $a = c_{\mathcal{F}} \sqrt{T}$. ■

If we do not know the number of steps in advance, it is possible to achieve a similar bound using the Aggregating Algorithm with a suitable initial weights distribution over the parameter a (see Vovk, 2005).

Seeger et al. (2008) analyze the order of $\ln \det \left(I + \frac{1}{a} \mathbf{K}_T \right)$ using other methods. For example, they show that if the input vectors are i.i.d. and the RBF kernel is used, the expected order of the regret term is $O(\log^n T)$.

4.3 Kernelized Ridge Regression

In this section we further develop the results obtained in Section 3.3. The Bayesian Algorithm which we use in this section becomes an analogue of non-parametric Bayesian methods.

Upper bounds on the logarithmic loss of Bayesian non-parametric algorithms (including our bound for the Kernelized Bayesian Ridge Regression) are derived by Kakade et al. (2005) and further analyzed by Seeger et al. (2008) using other methods. On the other hand, we provide a new equality on the square loss of the Kernelized Ridge Regression. Another approach to obtain the same result is described in Zhdanov and Kalnishkan (2010a).

4.3.1 Kernelized Bayesian Ridge Regression

Let the outcome set Ω be the real line $\mathbb{R}$ and the prediction set Γ be the set of all measurable functions on the real line integrable to one. The loss function λ is the logarithmic loss (2.20):

$$\lambda(y, \gamma) = -\ln \gamma(y),$$

where $\gamma \in \Gamma$ and $y \in \Omega$. The game follows Protocol 3. Input vectors x_t come from a set $\mathbf{X} \subseteq \mathbb{R}^n$.

Let $\mathcal{F}$ be the RKHS corresponding to a kernel function $k(x, z)$, $x, z \in \mathbf{X}$. We also use the notation (4.2). In addition, by X_T we denote the design matrix $T \times n$ consisting of the rows of the input vectors $x_1', \ldots, x_T'$.

Each expert $f \in \mathcal{F}$ predicts at the step t the normal distribution on the set of outcomes with the mean $f(x_t)$ and the variance σ^2 which is assumed to be known. In other words, each expert $f \in \mathcal{F}$ predicts

$$\xi_t^f(y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(f(x_t)-y)^2}{2\sigma^2}}. \tag{4.6}$$

Mixing these experts directly using the Bayesian Algorithm would correspond to taking a Gaussian process initial distribution over the infinite-dimensional class $\mathcal{F}$ of experts. Even though we will use this approach later in Section 4.5,

here we utilize the convenient properties of the kernel trick introduced in Section 4.2.

At step T the Kernelized Bayesian Ridge Regression algorithm (KBRR) predicts the normal density on outcomes with the mean $\mu_T = \mathbf{Y}'_{T-1}(aI + \mathbf{K}_{T-1})^{-1}\mathbf{k}_{T-1}$ and variance $\sigma^2 + \sigma^2(k(x_T, x_T) - \mathbf{k}'_{T-1}(aI + \mathbf{K}_{T-1})^{-1}\mathbf{k}_{T-1})/a$. We denote by L_T the cumulative logarithmic loss, over the first T steps, of the algorithm, and by L_T^f we denote the cumulative logarithmic loss of the expert $f \in \mathcal{F}$ over these steps.

Theorem 4.3 *For any sequence $x_1, y_1, x_2, y_2, \ldots$, the cumulative logarithmic loss of the kernelized Bayesian Ridge Regression algorithm at any step T can be expressed as*

$$L_T = \min_{f \in \mathcal{F}} \left(L_T^f + \frac{a}{2\sigma^2}\|f\|_{\mathcal{F}}^2 \right) + \frac{1}{2} \ln\det\left(I + \frac{1}{a}\mathbf{K}_T \right). \tag{4.7}$$

Proof Following Lemma 4.1 we need to prove that the theoretical guarantee (3.14) for the BRR can be expressed in such a way that all the input vectors appear only in dot products. Let us take the dot product kernel $k(x_i, x_j) = \langle x_i, x_j \rangle$.

Using Lemma A.4 we obtain that the Bayesian Ridge Regression algorithm gives the same predictions as the kernelized Bayesian Ridge Regression algorithm:

$$\begin{aligned}\mathbf{Y}'_{t-1}(aI + \mathbf{K}_{t-1})^{-1}\mathbf{k}_{t-1} &= \mathbf{Y}'_{t-1}(aI + X_{t-1}X'_{t-1})^{-1}X_{t-1}x_t \\ &= \mathbf{Y}'_{t-1}X_{t-1}(aI + X'_{t-1}X_{t-1})^{-1}x_t.\end{aligned}$$

Furthermore, we have by Lemma A.5 that

$$\ln\det\left(I + \frac{1}{a}\mathbf{K}_T \right) = \ln\det\left(I + \frac{1}{a}X_T X'_T \right) = \ln\det\left(I + \frac{1}{a}\sum_{t=1}^{T} x_t x'_t \right).$$

The regularized cumulative losses of the best experts are the same by the Representer Theorem. Thus we can apply Lemma 4.1 to obtain (4.7) for any kernel other than the dot product. ∎

This theorem is also proven by Kakade et al. (2005) for $a = \sigma^2$ using a different approach.

4.3.2 Kernelized Ridge Regression

In this section we prove bounds on the square loss of the Kernelized Ridge Regression algorithm (KRR). At step T it predicts the mean of the predictive distribution of the Kernelized Bayesian Ridge Regression: $\gamma_T = \mathbf{Y}'_{T-1}(aI + \mathbf{K}_{T-1})^{-1}\mathbf{k}_{T-1}$. The following theorem is an analogue of Theorem 3.3 for Ridge Regression.

Theorem 4.4 *The Kernelized Ridge Regression algorithm for the learner with $a > 0$ satisfies, at any step T,*

$$\sum_{t=1}^{T} \frac{(\gamma_t - y_t)^2}{1 + \frac{1}{a}(k(x_t, x_t) - \mathbf{k}'_{t-1}(aI + \mathbf{K}_{t-1})^{-1}\mathbf{k}_{t-1})} = \min_{f \in \mathcal{F}} \left(\sum_{t=1}^{T} (f(x_t) - y_t)^2 + a\|f\|^2_{\mathcal{F}} \right). \quad (4.8)$$

PROOF Following Lemma 4.1 we need to prove that the theoretical guarantee (3.18) for the RR can be expressed in such a way that all the input vectors appear only in dot products. Let us take the dot product kernel $k(x_i, x_j) = \langle x_i, x_j \rangle$.

Since the predictions of the Bayesian Ridge Regression algorithm and the kernelized algorithm are the same, the means of the predictive distributions are the same. Thus the KRR gives the same predictions as the RR. Furthermore, by Lemma A.4 we have the equality of the denominators:

$$\begin{aligned} 1 + \frac{1}{a}(k(x_t, x_t) - \mathbf{k}'_{t-1}(aI + \mathbf{K}_{t-1})^{-1}\mathbf{k}_{t-1}) \\ = 1 + \frac{1}{a}(x'_t(I - X'_{t-1}(aI + X_{t-1}X'_{t-1})^{-1}X_{t-1})x_t) \\ = 1 + \frac{1}{a}(x'_t(aI + X'_{t-1}X_{t-1})^{-1}((aI + X'_{t-1}X_{t-1}) - X'_{t-1}X_{t-1})x_t) \\ = 1 + x'_t(aI + X'_{t-1}X_{t-1})^{-1}x_t. \end{aligned}$$

The regularized cumulative losses of the best experts are the same by the Representer Theorem. Thus we can apply Lemma 4.1 to obtain (4.8) for any kernel other than the dot product. ■

We can see from Theorem 13.3.8 of Harville (1997) that

$$\begin{aligned}\det\left(I+\frac{1}{a}\mathbf{K}_T\right) &= \det\begin{pmatrix} I+\frac{1}{a}\mathbf{K}_{T-1} & \frac{1}{a}\mathbf{k}_{T-1} \\ \frac{1}{a}\mathbf{k}'_{T-1} & 1+k(x_T,x_T)/a \end{pmatrix} \\ &= \det\left(I+\frac{1}{a}\mathbf{K}_{T-1}\right)\left(1+\frac{k(x_T,x_T)-\mathbf{k}'_{T-1}(aI+\mathbf{K}_{T-1})^{-1}\mathbf{k}_{T-1}}{a}\right),\end{aligned}$$

and so by induction we have

$$\det\left(I+\frac{1}{a}\mathbf{K}_T\right) = \prod_{t=1}^{T}\left(1+\frac{k(x_t,x_t)-\mathbf{k}'_{t-1}(aI+\mathbf{K}_{t-1})^{-1}\mathbf{k}_{t-1}}{a}\right),$$

with $\mathbf{k}'_0(aI+\mathbf{K}_0)^{-1}\mathbf{k}_0$ understood to be 0. Using this equality and following the arguments of the proof of Corollary 3.1 we obtain the following corollary from Theorem 4.4.

Corollary 4.2 *Assume $|y_t| \le Y$ for all t, clip the predictions of kernelized Ridge Regression to $[-Y,Y]$, and denote them by γ_t^Y. Then*

$$\sum_{t=1}^{T}(\gamma_t^Y - y_t)^2 \le \min_{f\in\mathcal{F}}\left(\sum_{t=1}^{T}(f(x_t)-y_t)^2 + a\|f\|_{\mathcal{F}}^2\right) + 4Y^2\ln\det\left(I+\frac{1}{a}\mathbf{K}_T\right). \tag{4.9}$$

It is possible to prove this corollary directly from Corollary 3.1 using the same argument as in the proof of Theorem 4.4.

The order of the regret term in (4.9) can be analyzed using the same arguments as in the end of Section 4.2.2.

4.4 Kernelized regression with pointing prediction intervals under discounted loss

In this section we present the Kernelized Pointing Prediction Intervals Regression algorithm (KPPIR).

4.4.1 Derivation of the algorithm

Let the outcome sets Ω_t be the bounded intervals $\Omega_t = [Y_{t,1}, Y_{t,2}]$ announced by reality at the beginning of each prediction step t. The prediction set Γ is the real line $\mathbb{R}$ and the loss function is the square loss (2.24): $\lambda(y, \gamma) = (\gamma - y)^2$ with $y \in \Omega_t$, $\gamma \in \Gamma$. The square loss is discounted with the factors $\alpha_t \in [0, 1]$ at each step.

Let $\mathcal{F}$ be the RKHS corresponding to a kernel function $k(x, z)$, $x, z \in \mathbf{X}$. We also use the notation (4.2). Recall that $\eta_t = \frac{2}{(Y_{t,2} - Y_{t,1})^2}$ and $w_{t,T} = \eta_t \prod_{i=t}^{T-1} \alpha_i$. We also denote $W_T := \operatorname{diag}(w_{1,T}, w_{2,T}, \dots, w_{T,T})$ and thus $\sqrt{W_T} = \operatorname{diag}(\sqrt{w_{1,T}}, \sqrt{w_{2,T}}, \dots, \sqrt{w_{T,T}})$. Expert $f \in \mathcal{F}$ predicts at step t

$$\xi_t^f = f(x_t). \tag{4.10}$$

Proposition 4.2 *Predictions* (3.30) *of the PPIR can be represented as*

$$\gamma_T = \left(\mathbf{Y}_{T-1}; \frac{Y_2 + Y_1}{2} \right)' \sqrt{W_T} \left(aI + \sqrt{W_T} \mathbf{K}_T \sqrt{W_T} \right)^{-1} \sqrt{W_T} \mathbf{k}_T \tag{4.11}$$

for the linear kernel $k(x, y) = \langle x, y \rangle$, the unit $T \times T$ matrix I, and $a > 0$.

Proof Let by X denote the design matrix $T \times n$ consisting of the rows of the input vectors $x_1', \dots, x_T'$. Then we have $\mathbf{K}_T = XX'$ and $\sqrt{W_T}\mathbf{k}_T = \sqrt{W_T} X x_T$. By Lemma A.4 we obtain

$$\left(aI + \sqrt{W_T} X X' \sqrt{W_T} \right)^{-1} \sqrt{W_T} X x_T = \sqrt{W_T} X (aI + X' W_T X)^{-1} x_T$$

It is easy to see that

$$\left(\mathbf{Y}_{T-1}; \frac{Y_2 + Y_1}{2}\right)' W_T X = \left(\sum_{t=1}^{T-1} w_{t,T} y_t x_t + \eta_T \left(\frac{Y_{T,2} + Y_{T,1}}{2}\right) x_T\right)'$$

and

$$X' W_T X = \sum_{t=1}^{T-1} w_{t,T} x_t x_t' + \eta_T x_T x_T'.$$

Thus we obtain formula (3.30) from (4.11). ∎

The most time-consuming operation is the inversion of the updated matrix

$$aI + \sqrt{W_T}\mathbf{K}_T\sqrt{W_T} = aI + \begin{pmatrix} w_{1,T}k(x_1,x_1) & \sqrt{w_{1,T}w_{2,T}}k(x_1,x_2) & \cdots & \sqrt{w_{1,T}w_{T,T}}k(x_1,x_T) \\ \sqrt{w_{2,T}w_{1,T}}k(x_2,x_1) & w_{2,T}k(x_2,x_2) & \cdots & \sqrt{w_{2,T}w_{T,T}}k(x_2,x_T) \\ \vdots & \vdots & \ddots & \vdots \\ \sqrt{w_{T,T}w_{1,T}}k(x_T,x_1) & \sqrt{w_{T,T}w_{2,T}}k(x_T,x_2) & \cdots & w_{T,T}k(x_T,x_T) \end{pmatrix}.$$

It is interesting to find efficient algorithms to inverse the matrix, especially in the case when the outcome sets and the discounting do not depend on t: $Y_{t,2} = Y_2$, $Y_{t,1} = Y_1$, and $\alpha_t = \alpha$ is close to zero. Then the first rows and columns in the second addend matrix become small with time, and efficient algorithms may exist to provide reliable approximations of the inverse.

4.4.2 Performance guarantees

In this section we prove an upper bound on the discounted square loss of the KPPIR.

Theorem 4.5 *For any $a > 0$, every positive integer T, every sequence of outcomes of the length T, and any $f \in \mathcal{F}$, the KPPIR with the parameter a*

satisfies

$$\sum_{t=1}^{T} w_{t,T}(\gamma_t - y_t)^2 \le \sum_{t=1}^{T} w_{t,T}(f(x_t) - y_t)^2 + a\|f\|^2 + \frac{1}{2}\ln\det\left(\frac{\sqrt{W_T}\mathbf{K}_T\sqrt{W_T}}{a} + 1\right) \quad (4.12)$$

PROOF Following Lemma 4.1 we need to prove that the upper bound (3.25) for the PPIR can be expressed in such a way that all the input vectors appear only in dot products. Let us take the dot product kernel $k(x_i, x_j) = \langle x_i, x_j \rangle$.

Then the predictions of the PPIR are equal to the predictions of the KPPIR for all $t = 1, \ldots, T$ by Proposition 4.2, and thus the left-hand sides of (3.25) and (4.12) are the same. The regret terms

$$\frac{1}{2}\ln\det\left(I + \frac{\sqrt{W_T}\mathbf{K}_T\sqrt{W_T}}{a}\right) = \frac{1}{2}\ln\det\left(I + \frac{X'W_TX}{a}\right)$$

are the same due to Lemma A.5. The loss of any expert is upper bounded by the loss of the best expert achieving the minimal loss. The regularized cumulative losses of the best experts are the same by the Representer Theorem. Thus we can apply Lemma 4.1 to obtain the upper bound (4.12) for any kernel other than the dot product. ■

The order of the regret term in (4.12) can be analyzed using the same arguments as in the end of Section 4.2.2. We would like to pay attention to one particular special case when the prediction intervals and the discounting factor do not change with time: $Y_{t,2} = Y_2$, $Y_{t,1} = Y_1$, and $\alpha_t = \alpha$. We will use the notation $c^2_{\mathcal{F}} = \sup_{x \in \mathbf{X}} k(x, x)$ and assume that $c^2_{\mathcal{F}} < \infty$.

Corollary 4.3 *In the conditions of Theorem 4.5 and given in advance any constant $\mathcal{T}$ such that $\sum_{t=1}^{T} \alpha^{T-t} \le \mathcal{T}$, one can choose parameter a such that*

the strategy in Theorem 4.5 achieves

$$\sum_{t=1}^{T} \alpha^{T-t}(\gamma_t - y_t)^2 \leq \sum_{t=1}^{T} \alpha^{T-t}(f(x_t) - y_t)^2 + \left(\frac{(Y_2 - Y_1)^2}{4} + \|f\|^2\right) c_{\mathcal{F}} \sqrt{\mathcal{T}} \quad (4.13)$$

for any $f \in \mathcal{F}$.

PROOF The determinant of a symmetric positive definite matrix is upper bounded by the product of its diagonal elements (see Beckenbach and Bellman, 1961, Chapter 2, Theorem 7) and thus

$$\ln \det\left(I + \frac{\sqrt{W_T}\mathbf{K}_T\sqrt{W_T}}{a}\right) \leq T \ln\left(1 + \frac{c_{\mathcal{F}}^2\left(\prod_{t=1}^{T} w_{t,T}\right)^{1/T}}{a}\right) \leq T\frac{c_{\mathcal{F}}^2}{a}\left(\prod_{t=1}^{T} w_{t,T}\right)^{1/T}.$$

Moreover, using the inequality between the geometric and arithmetic means, we obtain

$$\left(\prod_{t=1}^{T} w_{t,T}\right)^{1/T} = \eta\left(\prod_{t=1}^{T} \alpha^{T-t}\right)^{1/T} \leq \eta\frac{\sum_{t=1}^{T}\alpha^{T-t}}{T} \leq \eta\frac{\mathcal{T}}{T}.$$

Choosing $a = \eta c_{\mathcal{F}}\sqrt{\mathcal{T}}$, we obtain (4.13) from (4.12). ∎

The role of time in (4.13) is played by the upper bound $\mathcal{T}$ on $\sum_{t=1}^{T}\alpha^{T-t}$, similarly to other problems with discounting (e.g. Section 3.4). Note that if $\alpha < 1$, then

$$\sum_{t=1}^{T}\alpha^{T-t} = \alpha^0 + \ldots + \alpha^{T-1} = \frac{1-\alpha^T}{1-\alpha} \leq \frac{1}{1-\alpha}.$$

Thus if α is known, it is easy to find $\mathcal{T}$ for all T at once.

4.5 Kernelized generalized linear models

In this section we kernelize the algorithm described in Section 3.5 and prove upper bounds on the square loss of the algorithm competing with the functions from an RKHS. Algorithm 9 is not formulated in terms of the dot product of input vectors, and thus the technique applied in Section 4.2 is not applicable here. Instead, we approach the kernelization problem by mixing infinite-dimensional Hilbert space of experts directly using the Gaussian process prior on them, similarly to the approach of Bayesian nonparametric methods.

An approach to kernelize the algorithms working with generalized linear models under the logarithmic loss function (standard models) was suggested by Cawley et al. (2007). Their algorithm uses only dot products of input vectors.

4.5.1 Derivation of the algorithm

We consider Protocol 3 with the outcome set $\Omega = [Y_1, Y_2]$, the prediction set $\Gamma = [Y_1, Y_2]$, and the square loss (2.24): $\lambda(y, \gamma) = (\gamma - y)^2$ with $y \in \Omega$, $\gamma \in \Gamma$. Input vectors x_t come from a set $\mathbf{X} \subseteq \mathbb{R}^n$.

Let $\mathcal{F}$ be the RKHS corresponding to a kernel function $k(x, z)$, $x, z \in \mathbf{X}$. We also use the notation (4.2). Expert $f \in \mathcal{F}$ predicts at step t

$$\xi_t^f = Y_1 + (Y_2 - Y_1)\sigma(f(x_t)) \tag{4.14}$$

for the activation function $\sigma : \mathbb{R} \to \mathbb{R}$. The cumulative square loss of the expert f at the step T is denoted by L_T^f.

We mix experts (4.14) using the Aggregating Algorithm (Algorithm 1). Let the initial distribution $P_0(df)$ over the experts be a Gaussian process; see Rasmussen and Williams (2006) for the description of some mathematical foundations and practical application of Gaussian processes.

Definition 4.4 (Gaussian process) A *Gaussian process* is a collection of random variables, any finite number of which have a joint Gaussian distribution.

The realization of the random variable distributed as a Gaussian process can be thought of as an infinite vector such that each component (and set of components) of it is a realization of a Gaussian distributed random variable. Each component of the vector can be interpreted as a function of the index of the component. The distribution over functions on different domains is defined similarly. It is possible to sample from a given Gaussian process, but only to obtain the values of a finite set of the components of the vector.

The initial distribution over functions evaluated in examples $x_1, \ldots, x_T$ is expressed as follows:

$$(f(x_1), \ldots, f(x_T))' \sim N\left(0, \frac{\mathbf{K}_T}{2\eta a}\right).$$

We denote it by $P_0^T(df)$. Even though this particular normal distribution depends on the examples, it corresponds to the initial distribution over the experts, the Gaussian process, which is independent on them.

Following the usual steps in the derivation of the prediction algorithm using the AA, we obtain that the generalized prediction calculated from unnormalized weights is expressed as follows:

$$G_T(y) = \log_\beta \int \beta^{\sum_{t=1}^{T-1}(\sigma(f(x_t))-y_t)^2+(\sigma(f(x_T))-y)^2} P_0^T(df) \tag{4.15}$$

for any $y \in \Omega$. The algorithm uses the substitution function (2.26) to give its prediction:

$$\gamma_T = \frac{Y_2 + Y_1}{2} - \frac{G_T(Y_2) - G_T(Y_1)}{2(Y_2 - Y_1)}. \tag{4.16}$$

In order to calculate the integral, it is possible to use a Monte-Carlo method. That is, to sample many vectors $(f(x_1), \ldots, f(x_T))'$ from the initial distribution, and then average the values of the integrated function. We call the algorithm which uses numerical integration the KAAGLM (Kernelized Aggregating Algorithm for Generalized Linear Models). It is possible to apply the methods of MCMC to calculate the integral more efficiently similarly to as described in Section 3.5.3.

4.5.2 Performance guarantee

To derive the upper bound on the loss we again need the activation function to have the property (3.33). The function

$$b(u,z) := \left(\frac{d\sigma(z)}{dz}\right)^2 + (\sigma(z)-u)\frac{d^2\sigma(z)}{dz^2} \tag{4.17}$$

should be uniformly bounded: $b := \sup_{u\in[0,1],z\in\mathbb{R}} |b(u,z)| < \infty$. We derive the following upper bound on the cumulative square loss of the prediction strategy $\mathcal{S}_K$ which uses the asymptotical (in the number of iterations in numerical integration) KAAGLM to give its predictions.

Theorem 4.6 *Let $a > 0$. There exists a prediction strategy $\mathcal{S}_K$ for the learner such that for every positive integer T, for every sequence of outcomes of the length T, and any $f \in \mathcal{F}$, the loss L_T of the learner satisfies*

$$L_T \le L_T^f + a\|f\|^2 + \frac{(Y_2-Y_1)^2}{4}\ln\det\left(1+\frac{b(Y_2-Y_1)^2\mathbf{K}_T}{a}\right). \tag{4.18}$$

PROOF Following Lemma 4.1, we have to prove that the theoretical guarantee (3.34) can be expressed in such a way that all the input vectors appear only in dot products. We also need to prove that the kernelized strategy $\mathcal{S}_K$ gives the same prediction as the strategy $\mathcal{S}$ used in Theorem 3.5 if the kernel is the dot product.

Let us take the dot product kernel $k(x_i,x_j) = \langle x_i, x_j\rangle$. Let us denote by h_i the values $f(x_i)$, $i = 1,\dots,T$, and by h^τ the vectors $h^\tau = (h_1,\dots,h_\tau)'$ for $\tau = 1,\dots,T$. Then the integral in (4.15) at the step τ is expressed as follows:

$$\int \beta^{\sum_{t=1}^{\tau-1}(\sigma(h_t)-y_t)^2+(\sigma(h_\tau)-y)^2} P_0^\tau(dh) \tag{4.19}$$

with the Gaussian measure $P_0^\tau(dh) \sim N\left(0,\frac{\mathbf{K}_\tau}{2\eta a}\right)$. On the other hand, the initial weights distribution (3.42) over the generalized linear models is also Gaussian, $P_0(d\theta) \sim N\left(0,\frac{I}{2\eta a}\right)$, and thus there is a measure over predictions $X_\tau\theta \sim N\left(0,\frac{X_\tau X_\tau'}{2\eta a}\right)$. Here X_τ is the matrix consisting of the rows of the input vectors $x_1',\dots,x_\tau'$. We also have $\mathbf{K}_\tau = X_\tau X_\tau'$. Therefore, the measures in both

of the integrals (4.19) for $\mathcal{S}_K$ and

$$\log_\beta \int \beta^{\sum_{t=1}^{T-1}(\sigma(\theta' x_t)-y_t)^2+(\sigma(\theta' x_T)-y)^2} P_0(d\theta)$$

for $\mathcal{S}$ induce the same measure over predictions. This leads to the fact that the predictions of the strategies are the same since they utilize the integrals in the same way (4.16).

The determinants

$$\det\left(I + \frac{b(Y_2 - Y_1)^2}{a}\mathbf{K}_T\right) = \det\left(I + \frac{b(Y_2 - Y_1)^2}{a}\sum_{t=1}^{T} x_t x_t'\right)$$

are the same to Lemma A.5.

The loss of any expert is upper bounded by the loss of the best expert achieving the minimal loss. The regularized cumulative losses of the best experts are the same by the Representer Theorem. Thus we can apply Lemma 4.1 to obtain the upper bound (4.18) for any kernel other than the dot product.■

The order of the regret term in (4.18) can be analyzed using the same arguments as in the end of Section 4.2.2.

4.6 Kernelized probability forecasting

In this section, we further develop the results obtained in Section 3.6. We only consider the algorithm which gives probability predictions directly (mAAR), because the properties of the component-wise algorithm easily follow from the properties of the KAAR described in Section 4.2.

4.6.1 Derivation of the algorithm

The set of outcomes $\Omega = \mathcal{P}(\Sigma)$ is the set of all probability measures on a finite set Σ with d elements, the set of predictions $\Gamma := \{(\gamma^1, \dots, \gamma^d) : \sum_{i=1}^d \gamma^i = 1, \gamma^i \in \mathbb{R}\}$ is a hyperplane in d-dimensional space containing all the outcomes, and the quality of predictions is measured by the Brier loss (2.29). The set of input vectors $\mathbf{X} \subseteq \mathbb{R}^n$ is a subset of the Euclidean space. The game of prediction follows Protocol 3. $\mathcal{F}$ is the RKHS corresponding to a kernel function $k(x, z)$, $x, z \in \mathbf{X}$.

Our algorithm competes with the experts predicting $\xi_t(f)$ according to the following functions:

$$\begin{aligned} \xi_t^1 &= 1/d + f^1(x_t) \\ &\dots \\ \xi_t^{d-1} &= 1/d + f^{d-1}(x_t) \\ \xi_t^d &= 1 - \xi^1 - \dots - \xi^{d-1} \end{aligned} \tag{4.20}$$

with $f^1, \dots, f^{d-1} \in \mathcal{F}$. We denote the cumulative loss of the expert $f = (f^1, \dots, f^{d-1})'$ by L_T^f.

We start by rewriting the prediction of the mAAR in the dual form. We use the notation (4.2). We also denote

$$\begin{aligned} \widetilde{\mathbf{Y}}_{T-1}^i &:= -2(y_1^i - y_1^d, \dots, y_{T-1}^i - y_{T-1}^d, -1/2)', \quad i = 1, \dots, d-1, \\ \overline{\mathbf{Y}}_{T-1}^i &:= -2(y_1^i - y_1^d, \dots, y_{T-1}^i - y_{T-1}^d, 0)', \quad i = 1, \dots, d-1. \end{aligned}$$

Let us set $A_T := aI + \begin{pmatrix} 2\mathbf{K}_T & \cdots & \mathbf{K}_T \\ \vdots & \ddots & \vdots \\ \mathbf{K}_T & \cdots & 2\mathbf{K}_T \end{pmatrix}$ with the unit $T(d-1) \times T(d-1)$ matrix I.

Proposition 4.3 *On trial T values G_T^i from* (3.46) *can be represented as*

$$G_T^i = \left(\widetilde{\mathbf{Y}}_{T-1}^1;\ \cdots;\ \widetilde{\mathbf{Y}}_{T-1}^{i-1};\ \overline{\mathbf{Y}}_{T-1}^i;\ \widetilde{\mathbf{Y}}_{T-1}^{i+1};\ \cdots;\ \widetilde{\mathbf{Y}}^{d-1}\right)' \cdot A_T^{-1}\left(\mathbf{k}(x_T);\ \cdots;\ \mathbf{k}(x_T);\ 2\mathbf{k}(x_T);\ \mathbf{k}(x_T);\ \cdots;\ \mathbf{k}(x_T)\right) \quad (4.21)$$

for the linear kernel $k(x,y) = \langle x, y\rangle$.

Proof By $M_T = (x_1, \ldots, x_T)$ we denote the matrix $n \times T$ consisting of the columns of the input vectors. Let us set

$$B_T = \begin{pmatrix} 2M_T & \cdots & M_T \\ \vdots & \ddots & \vdots \\ M_T & \cdots & 2M_T \end{pmatrix}, C_T = \begin{pmatrix} M_T' & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & M_T' \end{pmatrix}.$$

Then $h_T^i = -2\sum_{t=1}^{T-1}(y_t^i - y_t^d)x_t = M_T\overline{\mathbf{Y}}_{T-1}^i$, $i = 1, \ldots, d-1$, and we can decompose

$$b_T\{i\} = (h_{T-1}^1; \ldots; h_{T-1}^{d-1}) + (x_T; \ldots, x_T; \mathbf{0}; x_T; \ldots, x_T) = C_T'\left(\widetilde{\mathbf{Y}}_{T-1}^1;\ \cdots;\ \widetilde{\mathbf{Y}}_{T-1}^{i-1};\ \overline{\mathbf{Y}}_{T-1}^i;\ \widetilde{\mathbf{Y}}_{T-1}^{i+1};\ \cdots;\ \widetilde{\mathbf{Y}}_{T-1}^{d-1}\right).$$

The matrix A_T can be represented as $A_T = aI + C_TB_T$ because $\mathbf{K}_T = M_T'M_T$. Using Lemma A.4, we obtain from (3.46)

$$G_T^i = -(b_T\{i\})'(aI + B_TC_T)^{-1}z_t\{i\} = -\left(\widetilde{\mathbf{Y}}_{T-1}^1;\ \cdots;\ \widetilde{\mathbf{Y}}_{T-1}^{i-1};\ \overline{\mathbf{Y}}_{T-1}^i;\ \widetilde{\mathbf{Y}}_{T-1}^{i+1};\ \cdots;\ \widetilde{\mathbf{Y}}_{T-1}^{d-1}\right)' \cdot (aI + C_TB_T)^{-1}C_T\left(-x_T;\ \cdots;\ -2x_T;\ \cdots; -x_T\right), \quad i = 1, \ldots, d-1,$$

and $G_T^d = 0$. Note that $\mathbf{k}_T = M_T'x_T$, thus (4.21) holds. ∎

To get predictions one can use the substitution function from Proposition 2.1. We call the resulting algorithm the multidimensional Kernelized Aggregating Algorithm for Regression (mKAAR).

4.6.2 Performance guarantee

Theorem 4.7 *For any $a > 0$, every positive integer T, every sequence of outcomes of the length T, and any $f^1, \ldots, f^{d-1} \in \mathcal{F}$, the loss L_T of the mKAAR with the parameter $2a$ satisfies*

$$L_T \le L_T^f + 2a \sum_{i=1}^{d-1} \|f_i\|_{\mathcal{F}}^2 + \frac{1}{2} \ln \det \left(I + \frac{1}{2a} \begin{pmatrix} 2\mathbf{K}_T & \cdots & \mathbf{K}_T \\ \vdots & \ddots & \vdots \\ \mathbf{K}_T & \cdots & 2\mathbf{K}_T \end{pmatrix} \right). \quad (4.22)$$

Proof Following Lemma 4.1 we need to prove that the upper bound (3.49) for the mAAR can be expressed in such a way that all the input vectors appear only in dot products. Let us take the dot product kernel $k(x_i, x_j) = \langle x_i, x_j \rangle$.

The predictions of the mAAR and of the mKAAR are equal to each other by Proposition 4.3; thus the left-hand sides of (3.49) and (4.22) are the same. The regret terms

$$\frac{1}{2} \ln \det \left(I + \frac{1}{2a} \begin{pmatrix} 2C_T & \cdots & C_T \\ \vdots & \ddots & \vdots \\ C_T & \cdots & 2C_T \end{pmatrix} \right) = \frac{1}{2} \ln \det \left(I + \frac{1}{2a} \begin{pmatrix} 2\mathbf{K}_T & \cdots & \mathbf{K}_T \\ \vdots & \ddots & \vdots \\ \mathbf{K}_T & \cdots & 2\mathbf{K}_T \end{pmatrix} \right)$$

are the same due to Lemma A.5; here $C_T = \sum_{t=1}^T x_t x_t'$. The regularized cumulative losses of the best experts are the same by the Representer Theorem Thus using Lemma 4.1 we can prove (4.22) for any kernel other than the dot product. ∎

The regret term can be analyzed using the arguments of Section 4.2.2.

Chapter 5

Online regression in Banach spaces

In this chapter we describe online regression in Banach spaces. The idea of competing with a class of functions wider than Hilbert spaces, in particular Banach spaces, is very natural. Indeed, if we want to compete with some linear space F, it should have a norm. If there is no norm, F can contain a very complicated hypothesis f (e.g., an exact fit) and it is almost impossible to compete against it. If there is a norm, we can say that this hypothesis f is probably too complicated. The norm should be somehow aligned with the evaluation functional. Indeed, we are ultimately interested in predictions $f(x)$. The norm is just a measure of complexity, and the continuity of the functional gives some connections between predictions and this measure. So if hypotheses f_1 and f_2 are close w.r.t. the norm, they should produce similar predictions. This argument leads to the fact we need the notion of Proper Banach Functional Spaces. These are functional Banach spaces with continuous evaluation functional. Moreover, there are too many types of norms, so it is not enough to only give a measure of complexity. Some geometrical property of the space is needed, some measure "to which extent the norm is good". This property can be given, for example, by the convexity module, or by the smoothness module.

The mathematical theory of functional Banach spaces is wide and well developed in many areas (see, for example, Triebel, 2006). Unfortunately, may

be due to the high complexity of the theory, there exists much less known applications of it to machine learning than for Hilbert spaces. Despite of that, a description of some of the recent achievements in this area is given by Zhang et al. (2009). Grove et al. (1997) prove mistake bounds for a linear online classification algorithm. Gentile and Littlestone (1999) introduce online regression algorithms competing with linear functions from a finite-dimensional Banach space. Vovk (2007) uses a cardinally different method and develops an algorithm working with infinite-dimensional functional Banach spaces.

5.1 Competing with Banach lattices

In this section, we consider more general sets $\mathbf{X}$ of the input vectors than subsets of $\mathbb{R}^n$ used in previous chapters. We take $\mathbf{X}$ to be a subset of an abstract normed vector space. The performance of our algorithm is compared with the performance of any vector from the dual space (we say that it predicts linearly on the input vector). Thus we consider linear regression in infinite-dimensional abstract Banach spaces.

From one side, we show that it is enough to slightly modify algorithms developed to compete with linear functions (we take the AAR, Section 3.2, as an example) to enable them to work in our framework. On the other side, this surprising result comes with the assumption that all the input vectors are known in advance. We call it semi-online setting.

We then modify our algorithm to enable it to work with finite-dimensional vectors from a domain of $\mathbb{R}^n$ as input vectors and compete with the functions belonging to a functional Sobolev space. This may lead to a wide spectrum of applications, for example prediction of Brownian motion, which almost surely belongs to a Sobolev space. The results of this section are described in Zhdanov et al. (2010).

5.1.1 Linear functions and their p-norms

Let the outcome set Ω be the interval $[-Y, Y]$ for some $Y > 0$, the prediction set Γ be the real line $\mathbb{R}$, and the loss function be the square loss (2.24): $\lambda(y, \gamma) = (\gamma - y)^2$ for $y \in \Omega$, $\gamma \in \Gamma$.

Let also the set $\mathbf{X}$ of input vectors be a subset of $\mathbb{R}^n$, $\mathbf{X} \subseteq \mathbb{R}^n$, only in this subsection. A linear expert $\theta \in \mathbb{R}^n$ predicts $\xi_t^\theta = \theta' x_t$ at the step t.

We use the Aggregating Algorithm for Regression. The regret term in the upper bound (3.9) has the logarithmic order in T but linear in n. Therefore, it is better applicable for the case of small n and large T. We shall now prove an upper bound that grows more slowly in n and depends on non-euclidian norms of θ. By $\|x\|_p$ we denote the p-norm of any vector $x \in \mathbb{R}^n$: $\|x\|_p = (|x_1|^p + \ldots + |x_n|^p)^{1/p}$. We also denote the space of n-dimensional real vectors $x = (x_1, \ldots, x_n)$ equipped with the p-norm by ℓ_p^n, $p \geq 1$. We take

q to be such that $1/p + 1/q = 1$. We use the following constants of norms equivalence.

Lemma 5.1 *Let $a \in \mathbb{R}^n$, $1 \le p \le 2$, and $1/p + 1/q = 1$. Then*

$$\|a\|_2 \le \|a\|_p,$$
$$\|a\|_2 \le n^{1/2-1/q}\|a\|_q.$$

PROOF The first inequality follows from the fact that the function $f(p) = \|a\|_p$ is decreasing in p. Indeed,

$$f_p' = \|a\|_p \left(-\frac{\ln \|a\|_p^p}{p^2} + \frac{\sum_{i=1}^n |a_i|^p \ln |a_i|}{p\|a\|_p^p} \right).$$

We need to prove that

$$\sum_{i=1}^n |a_i|^p \ln |a_i| \le \|a\|_p^p \ln \|a\|_p.$$

Since norm is homogenous, it is enough to prove the inequality for any a such that $\|a\|_p = 1$. This follows from the fact that $\ln |a_i| \le 0$ for $|a_i| \le 1$ for all i.

To prove the second inequality we consider the Hölder inequality for $x, y \in \mathbb{R}^n$ and $b \ge 1$ (see Beckenbach and Bellman, 1961, p.21):

$$\sum_{i=1}^n |x_i y_i| \le \left(\sum_{i=1}^n |x_i|^b \right)^{1/b} \left(\sum_{i=1}^n |y_i|^c \right)^{1/c}$$

for $1/b + 1/c = 1$. This implies

$$\|a\|_2^2 = \sum_{i=1}^n |a_i|^2 \le \left(\sum_{i=1}^n (|a_i|^2)^{q/2} \right)^{2/q} \left(\sum_{i=1}^n |1|^{\frac{q}{q-2}} \right)^{\frac{q-2}{q}},$$

for $b = q/2 \ge 1$ and $c = \frac{q}{q-2}$. Therefore $\|a\|_2 \le n^{1/2-1/q}\|a\|_q$. ∎

Lemma 5.2 *For each positive integer T there is a constant $a > 0$ such that for any sequence $(x_1, y_1), \ldots, (x_T, y_T)$ such that $||x_t||_q \le X$ for all $t = 1, \ldots, T$,*

the loss L_T of the AAR with the parameter a satisfies

$$L_T \le L_T^\theta + (Y^2X^2 + \|\theta\|_p^2)T^{1/2}n^{1/2-1/\max(q,p)} \tag{5.1}$$

for any $\theta \in \ell_p^n$.

PROOF Let by X denote the design matrix of the rows of the input vectors $x_1', \ldots, x_T'$. By Lemma A.5 we have

$$\det\left(I + \frac{1}{a}\sum_{t=1}^{T} x_t x_t'\right) = \det\left(I + \frac{1}{a}XX'\right)$$

in (3.8). Thus, bounding the determinant of the matrix by the product of its diagonal elements (see Beckenbach and Bellman, 1961, Chapter 2, Theorem 7) and using $\ln(1+z) \le z$ for $z \ge 0$, we obtain

$$L_T \le L_T^\theta + a\|\theta\|_2^2 + Y^2T\frac{\max_{t=1,\ldots,T}\|x_t\|_2^2}{a}.$$

If $q \ge 2$, then by Lemma 5.1 $\|x_t\|_2^2 \le n^{1-2/q}\|x_t\|_q^2$ for all t and $\|\theta\|_2^2 \le \|\theta\|_p^2$. This leads to $a\|\theta\|_p^2 + \frac{Y^2Tn^{1-2/q}X^2}{a}$ in the bound. By choosing $a = \sqrt{Tn^{1-2/q}}$ we obtain $(Y^2X^2 + \|\theta\|_p^2)T^{1/2}n^{1/2-1/q}$.

If $1 \le q \le 2$, then by Lemma 5.1 $\|x_t\|_2^2 \le \|x_t\|_q^2$ for all t and $\|\theta\|_2^2 \le n^{1-2/p}\|\theta\|_p^2$. This leads to $an^{1-2/p}\|\theta\|_p^2 + \frac{Y^2TX^2}{a}$ in the bound. For $a = \sqrt{Tn^{1-2/q}}$ and using $1/p + 1/q = 1$ we obtain $(Y^2X^2 + \|\theta\|_p^2)T^{1/2}n^{1/2-1/p}$. ∎

Remark 5.1 We can deduce another bound from (3.9). We have $\|x\|_\infty \le \|x\|_q$ for any $q \ge 1$. If $q \ge 2$, we have $\|\theta\|_2 \le \|\theta\|_p$, and the upper bound becomes

$$L_T \le L_T^\theta + a\|\theta\|_p^2 + nY^2\ln\left(\frac{TX^2}{a} + 1\right).$$

If $1 \le q < 2$, we have $\|\theta\|_2 \le n^{1/2-1/p}\|\theta\|_p$, and the upper bound becomes

$$L_T \le L_T^\theta + an^{1/2-1/p}\|\theta\|_p^2 + nY^2\ln\left(\frac{TX^2}{a} + 1\right).$$

The last bounds are better in T but worse in n than (5.1). In our main theorem

below we consider spaces of infinite dimension. The role of n is played there by the dimension of the span of the input vectors so far, which is generally T (because input vectors also come from an infinite-dimensional space), and only bounds similar to (5.1) remain nontrivial.

5.1.2 Banach lattices in semi-online setting: framework, algorithm, and performance guarantee

In this section we need to consider a different protocol than Protocol 3. The learner plays the game following Protocol 5. S is an abstract normed linear space. The prediction set Γ is the real line.

Protocol 5 Semi-online abstract regression

Reality announces number of steps T and input vectors $x_1, \dots, x_T \in S$.
for $t = 1, 2, \dots, T$ **do**
  Experts announce $\xi_t^\theta \in \Gamma$.
  Learner announces $\gamma_t \in \Gamma$.
  Reality announces $y_t \in \Omega$.
end for

The learner competes with the experts predicting according to the linear predictors from the dual space S^*. We derive the algorithm BLAAR (Banach Lattices-competing AAR) working in $L_p(\mu), p \geq 1$. Recall that $L_p(\mu)$ is the space of all μ-equivalent classes of p-integrable μ-measurable functions on $\mathbb{R}^n$:

$$\|f\|_{L_p(\mu)} = \left(\int_{\mathbb{R}^n} |f|^p d\mu \right)^{1/p} < \infty.$$

We use the notation $L_p := L_p(\mu)$. The BLAAR uses the Kernelized Aggregating Algorithm for Regression (see Section 4.2) to give its predictions. It is described as Algorithm 12 and the derivation follows in Section 5.1.3.

We prove the following upper bound for the cumulative loss of the BLAAR. It competes with the experts following linear functions from $(L_p)^*$. Given an input vector x, the expert f predicts $f(x)$. Its cumulative square loss is denoted by L_T^f.

Theorem 5.1 *Suppose we are given $p > 1$ and $x_1, \dots, x_T \in L_p$ for any positive integer T. Assume also that $\|x_t\|_{L_p} \le X$ for all $t = 1, \dots, T$. Then there exists $a > 0$ such that for all $f \in (L_p)^*$ and any sequence $y_1, \dots, y_T$, the cumulative loss L_T of the BLAAR with the parameter a satisfies*

$$L_T \le L_T^f + (Y^2X^2 + \|f\|^2_{(L_p)^*})T^{1/2+|1/2-1/p|}. \tag{5.2}$$

The proof of this theorem is given in Section 5.1.3. Note that if in Lemma 5.2 we take $n = T$, then the AAR gives the regret term of the same order $T^{1-1/p}$ for $\ell_p^n, p \ge 2$.

Algorithm 12 BLAAR for L_p.

Require: Number of steps T and input vectors $x_1, \dots, x_T \in L_p$.

Step 1. Find a linearly independent subset of $x_1, \dots, x_T$ with the maximum number of vectors: $x_{r_1}, \dots, x_{r_n}$.

Step 2. Solve the following optimization problem. Maximize the absolute value of the determinant of the $n \times n$ matrix $C = \{c_{ij}\}_{ij}$ ($|\det C| \to \max$) of the expansion coefficients of $\gamma_i = \sum_{j=1}^n c_{ij} x_{r_j}$ subject to the constraints

$$\left\| \sqrt{\sum_{i=1}^n |\gamma_i|^2} \right\|_{L_p} \le 1.$$

Step 3. Take $a = \sqrt{T} n^{-|1/2-1/p|}$. Use it as a parameter for the KAAR.

Step 4. Find the coefficients β_{si}, $s = 1, \dots, T$, $i = 1, \dots, n$, of the expansions $x_s = \sum_{i=1}^n \beta_{si}\gamma_i$.

Step 5. Define the kernel matrix

$$k(x_s, x_t) = \frac{1}{n} \sum_{i=1}^n \beta_{si}\beta_{ti}, \quad s, t = 1, \dots, T. \tag{5.3}$$

for $t = 1, 2, \dots, T$ **do**

Predict $\gamma_t \in \Gamma$ with KAAR according to the formula (4.3) with $c = 0$, $Y_2 = Y$, and $Y_1 = -Y$.

Read y_t.

end for

It is possible to generalize the result on Banach lattices of more general types (see the definition in Lindenstrauss and Tzafriri, 1979 or Tomczak-

Jaegermann, 1989). The algorithm becomes rather tricky and the derivation of it is too technical, thus we do not discuss the general case here. The proof of an analogue of Theorem 5.1 for general lattices is given in Zhdanov et al. (2010). The lattices are a well-studied wide class of Banach spaces. We use the fact that any $L_p(\mu)$ is a lattice (consequently, ℓ_p is a lattice). Other examples of Banach lattices include Orlicz spaces.

5.1.3 The proof of the main result and the derivation of the algorithm

In this section we prove Theorem 5.1 and derive Algorithm 12.

Proof of the main result

The proof is based on the possibility to construct an isomorphism between a finite-dimensional subspace of L_p and a Hilbert space such that the norms of the vectors do not increase too much. Precisely (see Lewis, 1978, Corollary 5),

Theorem 5.2 (Distance) *If $\mathcal{X}$ is an n-dimensional subspace of L_p, $1 < p < \infty$, then there exists an isomorphic operator $U : \mathcal{X} \to \mathbb{R}^n = \ell_2^n$ such that*

$$\|U\|\|U^{-1}\| \leq n^{|1/2-1/p|}. \tag{5.4}$$

Here $\|U\| = \sup_{x\in\mathcal{X}} \frac{\|Ux\|}{\|x\|}$ and $\|U^{-1}\| = \sup_{r\in\mathbb{R}^n} \frac{\|U^{-1}r\|}{\|r\|}$.

The expression $\inf_V \|V\|\|V^{-1}\|$ over all isomorphisms $V : \mathcal{X} \to \mathbb{R}^n$ defines a *Banach-Mazur distance* $d(\mathcal{X}, \mathbb{R}^n)$ between $\mathcal{X}$ and $\mathbb{R}^n$. For the case of a general Banach space the John theorem (John, 1948) states that $d(\mathcal{X}, \mathbb{R}^n) \leq \sqrt{n}$ for any n-dimensional subspace $\mathcal{X} \in B$, where B is any Banach space. Thus our theorem can be applied for the cases $p = 1$ and $p = \infty$ though the regret term becomes trivial (of order T).

Proof (of Theorem 5.1) Let $\mathcal{X} \subset L_p$ be the linear span of the input vectors $x_1, \ldots, x_T$. Let the dimension of $\mathcal{X}$ be n. By the Distance theorem there

exists an isomorphism $U : \mathcal{X} \to \mathbb{R}^n$ such that

$$\|U\|\|U^{-1}\| \le n^{|1/2-1/p|} = C.$$

Let us assume (without loss of generality) that the norms of the operator U and of the inverse operator U^{-1} are as follows:

$$\|U\| = \sup_{x\neq 0, x\in\mathcal{X}} \frac{\|U(x)\|}{\|x\|} = 1, \quad \|U^{-1}\| = \sup_{r\neq 0, r\in\mathbb{R}^n} \frac{\|U^{-1}(r)\|}{\|r\|} \le C. \tag{5.5}$$

If the norm of the direct operator does not equal one, we can always replace it by the operator $V = U/||U||$ with unitary norm. The norm of the inverse operator then increases by $||U||$.

By $r_i = U(x_i)$, $i = 1, \ldots, T$, we denote the images of the input vectors x_i applying operator U. We apply the KAAR with the scalar product kernel to these images r_i consequently. From Theorem 4.2 we obtain that for any $g \in (\mathbb{R}^n)^*$ and $a > 0$ the loss L_T of the learner at the step T satisfies

$$L_T \le L_T^g + a\|g\|_2^2 + Y^2 T \frac{\max_{i=1,\ldots,T} \|r_i\|_2^2}{a}, \tag{5.6}$$

where the determinant of the positive definite matrix $I + \frac{1}{a}\mathbf{K}_T$ ($\mathbf{K}_T$ is the matrix of scalar products (r_i, r_j)) is bounded by the product of its diagonal elements (Beckenbach and Bellman, 1961, Chapter 2, Theorem 7) and the logarithm $\ln(1 + x)$ is bounded by x for $x \ge 0$.

For any $f \in (L_p)^*$ we fix $\tilde{g} : \mathbb{R}^n \to \mathbb{R}$ such that $\tilde{g}(r) := f(U^{-1}(r))$ for any $r \in \mathbb{R}^n$. Since U is an isomorphism, $U^{-1}(r) \in L_p$. The linearity of $\tilde{g}$ follows from the linearity of U^{-1}, so $\tilde{g} \in (\mathbb{R}^n)^*$. This means that $L_T^{\tilde{g}} = L_T^f$ because the values of f and $\tilde{g}$ are equal on the corresponding vectors from $\mathcal{X}$ and $\mathbb{R}^n$.

Let us consider any linear functional $h : \mathcal{X} \to \mathbb{R}$ such that $h(x) = f(x)$ for all $x \in \mathcal{X}$. Clearly,

$$\|f\|_{(L_p)^*} = \sup_{\|x\|=1, x\in L_p} |f(x)| \ge \sup_{\|x\|=1, x\in\mathcal{X}} |f(x)| = \sup_{\|x\|=1, x\in\mathcal{X}} |h(x)| = \|h\|_{\mathcal{X}}.$$

The norm of h can be lower estimated using (5.5):

$$\|h\|_{\mathcal{X}} = \sup_{x\neq 0, x\in\mathcal{X}} \frac{|h(x)|}{\|x\|} = \sup_{r\neq 0, r\in\mathbb{R}^n} \frac{|\tilde{g}(r)|}{\|U^{-1}(r)\|} \geq \frac{1}{C} \sup_{r\neq 0, r\in\mathbb{R}^n} \frac{|\tilde{g}(r)|}{\|r\|} = \frac{1}{C}\|\tilde{g}\|_{(\mathbb{R}^n)^*}.$$

On the other hand, we have $\|r\| \leq \|x\|$ for all $x \in \mathcal{X}$, $r = U(x)$. Thus $\|r_i\|^2 \leq \|x_i\|^2$, $i = 1, \ldots, T$, and $\|\tilde{g}\|^2_{(\mathbb{R}^n)^*} \leq C^2\|h\|^2_{\mathcal{X}} \leq C^2\|f\|^2_{(L_p)^*}$, so the upper bound (5.6) transforms to

$$L_T(\text{BLAAR}) \leq L_T(f) + aC^2\|f\|^2_{(L_p)^*} + \frac{Y^2TX^2}{a}.$$

We can choose $a = \sqrt{T}/C$ and recall that n in C is the number of linearly independent input vectors among $x_1, \ldots, x_T$. Thus $n \leq T$, and we get the bound (5.2). ∎

Derivation of Algorithm 12

The derivation of our algorithm is based on the proof of the Distance theorem given in Lewis (1978) and on the proof of Theorem 5.1.

Step 2: The optimization task. Let $\mathcal{X} := \text{span}\{x_{r_1}, \ldots, x_{r_n}\}$ and thus $\dim \mathcal{X} = n$. We construct the isomorphism U (to be more precise, U^{-1}) for the Distance theorem in the following way.

We take some basis $\phi_1, \ldots, \phi_n \in \mathcal{X}^*$. For any isomorphism $u : \mathbb{R}^n \to \mathcal{X} \subset L_p$ we define its determinant by $\det u = \det\{\phi_i(\gamma_j)\}_{ij}$, where $\gamma_j = u(e_j)$ for the unit vector basis $e_1, \ldots, e_n \in \mathbb{R}^n$.

Then we find u_0 such that $|\det u_0| \to \max$ and $\|\gamma_{\mathcal{X}}\|_{L_p} \leq 1$, where $\gamma_{\mathcal{X}} = \sqrt{\sum_{i=1}^n |\gamma_i|^2}$. The resulting isomorphism u_0 turns out to be the one which is used in the Distance theorem, $u_0 = U^{-1}$, for the proof see Lewis (1978) or Tomczak-Jaegermann (1989). The absolute value of the maximum of the determinant is unique up to a constant depending on the choice of $\{\phi_i\}_1^n$. It is convenient to choose ϕ_i such that $\phi_i(x) = a_i$ for any $x = \sum_{i=1}^n a_i x_{r_i} \in \mathcal{X}$ (so $\{\phi_i\}_1^n$ is a biorthogonal system to $\{x_{r_i}\}_1^n$). Then $|\det u| = |\det\{c_{ij}\}_{ij}|$, where $\{c_{ij}\}_{ij}$ is the matrix of the coefficients of the expansions of $\gamma_1, \ldots, \gamma_n$ in the basis $x_{r_1}, \ldots, x_{r_n}$: $\gamma_i = \sum_{j=1}^n c_{ij} x_{r_j}$. We say that the coefficients $\{c_{ij}\}_{ij}$ make

up the $n \times n$ matrix C.

Steps 4 and 5: The scalar product. The scalar product of $x_1, \ldots, x_n$ (the kernel matrix) can be defined through the scalar product of the images of these vectors under the isomorphism u_0^{-1}, i.e., $\langle x_s, x_t \rangle := \langle u_0^{-1}(x_s), u_0^{-1}(x_t) \rangle_{\mathbb{R}^n}$. Each input vector can be expanded in the basis $\gamma_1, \ldots, \gamma_n$:

$$x_s = \sum_{i=1}^{n} \beta_{si} \gamma_i$$

with some coefficients β_{si}.

It is known (Lewis, 1978) that $\langle \gamma_i, \gamma_j \rangle = \delta_{ij}/n$, where δ_{ij} is the Kronecker delta, $i, j = 1, \ldots, n$. Thus we have

$$\langle x_s, x_t \rangle = \frac{1}{n} \sum_{i=1}^{n} \beta_{si} \beta_{ti}.$$

5.1.4 Applications of the algorithm

In this section we consider possible applications of our main theorem.

Algorithm competing with functional Banach spaces

A different protocol than Protocol 5 is usually considered in the online regression literature (Protocol 3): in this protocol input vectors are elements of some domain $\mathbf{X} \subseteq \mathbb{R}^n$. Our goal is to find an algorithm competing with all the functions from a functional Banach space $\mathcal{B}$ on this domain $\mathbf{X}$. Many algorithms are capable to compete with the functions from Reproducing Kernel Hilbert Spaces, for example the algorithms which we described in Chapter 4. The generalization of the notion of these spaces for the Banach case is called a Proper Banach Functional Space by Vovk (2007).

Definition 5.1 (Proper Banach Functional Space) A functional Banach space $\mathcal{B}$ on a set $\mathbf{X}$ is called a *Proper Banach Functional Space* (PBFS) if the evaluation functional $\varphi : f \in \mathcal{B} \mapsto f(x)$ is continuous for each $x \in \mathbf{X}$.

We will use the notation $c_{\mathcal{B}}(x)$ for the norm of this functional: $c_{\mathcal{B}}(x) := \sup_{f:\|f\|_{\mathcal{B}}\leq 1} |f(x)|$ and the embedding constant

$$c_{\mathcal{B}} := \sup_{x\in\mathbf{X}} c_{\mathcal{B}}(x)$$

is assumed to be finite.

We cannot compete with L_p spaces directly since they are not proper. Despite of that, we will show a way to compete with very important classes of Banach spaces: Besov and Triebel-Lizorkin spaces with appropriate parameters (Triebel, 1978). We start our description with the discussion of the algorithm competing with fractional Sobolev spaces. Sobolev spaces W_p^s have two parameters p, s. Intuitively, the parameter p is responsible for the rotundity of the unit ball (the unit ball is perfectly round when $p = 2$, and the space becomes a Hilbert space), and the parameter s is responsible for the smoothness of the functions in the space. If s is a positive integer, it represents the number of times which the functions from W_p^s are differentiable. Spaces with fractional s are generalizations of the spaces with integer s (see, for example, Adams and Fournier, 2003).

Competing with Sobolev spaces The main trick used in order to compete with Sobolev spaces is to impose an L_p structure on these spaces. We will further restrict ourselves to the spaces defined on the whole $\mathbb{R}^n$: in this case this isomorphism can be easily found. It can also be found if $\mathbf{X}$ is an open, non-empty subset of $\mathbb{R}^n$ such that there exists a linear extension operator (see definition on p.1372 of Pełczyński and Wojciechowski, 2003) from the Sobolev space $W_p^s(\mathbf{X})$ into $W_p^s(\mathbb{R}^n)$, for example for Lipschitz domains (see, e.g., Rogers, 2006).

Let us take a function $u(x) : \mathbb{R}^n \to \mathbb{R}$ and by

$$\widehat{u}(y) = \frac{1}{(2\pi)^{n/2}} \int_{\mathbb{R}^n} u(x) e^{-ixy} dx$$

denote the Fourier transform of $u(x)$. By $f^\vee$ we denote the inverse Fourier

transform

$$f^{\vee}(x) = \frac{1}{(2\pi)^{n/2}} \int_{\mathbb{R}^n} f(y) e^{ixy} dy$$

of a function f. Then the isomorphism between a Sobolev space and a subspace of L_p is described by the following theorem. It is constructed using Bessel potentials.

Theorem 5.3 (Isomorphism of W_p^s and L_p) *Let $1 < p < \infty, s > 0$ such that $sp > n$. Then W_p^s may be described as*

$$W_p^s = \{f \in S'(\mathbb{R}^n) : \left((1 + \|y\|_2^2)^{s/2} \widehat{f}(y)\right)^{\vee} \in L_p(\mathbb{R}^n)\}, \tag{5.7}$$

where $S'(\mathbb{R}^n)$ is a collection of all tempered distributions on $\mathbb{R}^n$.

The proof is provided in Theorem 1.3.2 of Triebel (1992). In other words, the Isomorphism theorem states that each element of W_p^s can be linearly (because of the linearity of Fourier transform) identified with an element from L_p, and vice versa, such that under this identification the norms in W_p^s and in L_p are equivalent.

The mapping in (5.7) means the convolution of the given function f and the function with the polynomial Fourier transform $(1 + \|y\|_2^2)^{s/2}$ (the latter called Bessel potential). Explicit expressions of these functions can be found in Aronszajn et al. (1963).

Using this theorem, we obtain the following upper bound on the loss of an algorithm competing with Sobolev spaces. By L_T^f we denote the cumulative loss of the predictor $f \in W_p^s$.

Theorem 5.4 *Assume $\mathbf{X} \subseteq \mathbb{R}^n$ and $W_p^s(\mathbb{R}^n)$ is a fractional Sobolev space of functions on $\mathbb{R}^n$, $s > 0, p > 1$. Suppose we are given a positive integer T and $x_1, \ldots, x_T \in \mathbf{X}$. Then there exists an algorithm such that for all $f \in W_p^s(\mathbb{R}^n)$ and any sequence $y_1, \ldots, y_T$, its loss L_T satisfies*

$$L_T \leq L_T^f + (Y^2 c_{W_p^s}^2 + \|f\|^2) K T^{1/2 + |1/2 - 1/p|}. \tag{5.8}$$

Here K is defined by the isomorphism (5.7) between W_p^s and L_p.

PROOF Sobolev spaces are proper Banach spaces (see Triebel, 2005, Proposition 7(ii)) and have $c_{W_p^s} < \infty$. From the other hand, we impose a lattice structure on the Sobolev space using isomorphism (5.7).

We represent $x_1, \ldots, x_T$ as elements α_i of the dual space $(W_p^s)^*$. For all $f \in W_p^s$ we take $\alpha_i(f) = \alpha_{x_i}(f) := f(x_i)$, $i = 1, \ldots, T$. We can now consider the abstract setting from Subsection 5.1.2. In this setting we compete with the elements of $(W_p^s)^{**}$: for each $f \in W_p^s$ we take $g_f \in (W_p^s)^{**}$ such that by definition $g_f(\alpha) := \alpha(f)$ for any $\alpha \in (W_p^s)^*$. This change of variables does not change the prediction error because $f(x_i) = \alpha_i(f) = g_f(\alpha_i)$.

The Isomorphism theorem states that there exists a linear isomorphism $U : L_p \to W_p^s$ between L_p and W_p^s such that $\|U\|\|U^{-1}\| < K$ for some constant K. We also denote

$$C_U = ||U|| = \sup_{\eta \in L_p} \frac{||U\eta||}{||\eta||}, \quad C_{U^{-1}} = ||U^{-1}|| = \sup_{f \in W_p^s} \frac{||U^{-1}f||}{||f||}.$$

This isomorphism defines the dual isomorphism $U^* : (W_p^s)^* \to (L_p)^*$ by

$$(U^*\alpha)(\eta) = \alpha(U\eta)$$

with any $\eta \in L_p$ and $\alpha \in (W_p^s)^*$. Clearly, $(U^*\alpha)(U^{-1}f) = \alpha(f)$ for all $f \in W_p^s$. We denote $\beta = U^*\alpha \in (L_p)^*$. Similarly, we have the correspondence $U^{**} : (W_p^s)^{**} \to (L_p)^{**}$ defined by

$$(U^{**}g)(\beta) = g((U^*)^{-1}\beta),$$

which gives us functions $h = U^{**}g \in (L_p)^{**}$ to compete with. After these replacements we have the same prediction error since for any $g \in (W_p^s)^{**}$ we get $h(\beta_i) = (U^{**}g)(\beta_i) = g((U^*)^{-1}\beta_i) = g(\alpha_i)$, $i = 1, \ldots, T$. The norm of the function can not increase by more than $C_{U^{-1}}$:

$$||h|| = \sup_{\beta} \frac{|h(\beta)|}{||\beta||} = \sup_{\alpha} \frac{|g(\alpha)|}{||U^*\alpha||} \leq \frac{|g(\alpha)|}{|\alpha(U\eta)|}||\eta|| = ||\eta|| \leq C_{U^{-1}}\|f\|,$$

where $\eta = U^{-1}f$ and $g(\alpha) = g_{U\eta}(\alpha)$. The first inequality follows from the fact

that

$$\|U^*\alpha\| = \sup_{\eta} \frac{|(U^*\alpha)(\eta)|}{\|\eta\|} \geq \frac{|(U^*\alpha)(\eta)|}{\|\eta\|} = \frac{|\alpha(U\eta)|}{\|\eta\|}$$

for all $\eta \in L_p$. To apply Theorem 5.1, we have to ensure that all $||\beta_i||$ are bounded. It holds since

$$||\beta_i|| = \sup_{\eta} \frac{|(U^*\alpha_i)(\eta)|}{||\eta||} = \sup_{f} \frac{|f(x)|}{||U^{-1}f||} \leq \sup_{f} \frac{|f(x_i)|}{||f||} C_U \leq c_{W_p^s} C_U$$

for $i = 1, \ldots, T$. Applying Theorem 5.1 concludes the proof. ∎

The algorithm used in the proof of Theorem 5.4 first identifies all the input vectors $x_1, \ldots, x_T$ with the functionals from $(W_p^s)^*$. It then finds $\beta_1, \ldots, \beta_T \in (L_p)^*$ defined using the isomorphism obtained from (5.7) between the duals $(W_p^s)^*$ and $(L_p)^*$. Then it applies the BLAAR to $\beta_1, \ldots, \beta_T$.

The order of the regret term in (5.8) reaches its minimum in p when $p = 2$. In this case W_p^s is a Hilbert space. The PBFS property implies that W_p^s is a Reproducing Kernel Hilbert Space. The order $O(\sqrt{T})$ of the regret term corresponds to the order in the bounds for the algorithms competing with RKHS described in the previous chapter.

Competing with Besov spaces Lately Besov $B_{p,q}^s$ and Triebel-Lizorkin $F_{p,q}^s$ function spaces begin to interest researchers due to their connections with wavelets theory. They have the PBFS property (see Triebel, 2005, Proposition 7(ii)), and $c_{B_{p,q}^s}, c_{F_{p,q}^s} < \infty$. By the embedding theorem (Triebel, 1978, Theorem 2.3), we have $F_{p,q}^s \to B_{p,q}^s \to F_{p,2}^{s'} = W_p^{s'}$, $1 < p \leq q < \infty, s > s'$. Here embedding $A \to B$ implies that there exists a constant C and linear operator $T : A \to B$ such that for any $f \in A$ we have $Tf \in B$ and $\|Tf\|_B \leq C\|f\|_A$. For Slobodetsky spaces $B_p^s = B_{p,p}^s$, we can use another result from the same theorem: $B_p^s \to W_p^s$, $2 \leq p < \infty, s > 0$. It helps to keep the same parameter s and thus does not increase the multiplicative constants in the regret term. Using Theorem 5.4, we can get the upper bound

$$L_T \leq L_T^f + (Y^2 c_{\{B,F\}_{p,q}^s}^2 + \|f\|_{\{B,F\}_{p,q}^s}^2) C T^{1/2+|1/2-1/p|}$$

for all $f \in \{B, F\}^s_{p,q}, p > 1$ and some $C > 0$ defined as the multiplication of the embedding constants. An interesting class of spaces is the class of Hölder-Zygmund spaces $\mathcal{C}^s = B^s_\infty$. They are embedded to $B^{s'}_p = B^{s'}_{p,p}$ whenever $s' < s$. It is known that fractional Brownian motion $B^{(h)}$ almost surely belongs to $\mathcal{C}^s$, $s < h$.

Application of the abstract framework

In this section we describe an example of how our algorithm can be used in signal processing. An input vector can often be interpreted as a function on some domain, e.g., a picture can be thought of as a mapping from points to colors. A musical fragment can be thought of as a mapping from a point in time into sound frequencies. We may be given weak regularity restrictions on the class these functions form, e.g., it can be a Sobolev or Besov space. The family of Hilbert spaces is reasonably wide, but if lacks many classes of functions of irregular behavior.

Imagine we are given a film consisting frames of resolution 1024×768 and we want to predict some score calculated from each image. The correct linear score for each image is given to us only after we make a prediction about the score of this image.

Applying the algorithm from Lemma 5.2 for prediction we can get the following upper bound for the square loss of our predictions

$$L_T \leq (Y^2 X^2 + \|\theta\|_p^2) T^{1/2} n^{1/2-1/p}$$

where $p \geq 2$, $n = 1024 \times 768 = 786432$, T is the length of the film in frames, X is the maximal q-norm of the images ($1/q + 1/p = 1$), and Y is the upper bound on the absolute value of the score. The upper bound from the remark after Lemma 5.2 is worse in n, and in our example n is the dominating constant for reasonable films length. On the other hand, the algorithm from Theorem 5.1 has the upper bound

$$L_T \leq (Y^2 X^2 + \|\theta\|_p^2) T^{1-1/p}.$$

If we wish to predict 24 frames per second (say, to detect defective frames), the upper bound on the loss of the second algorithm will be better if we work with films of duration less than 32768 (around 9 hours). The higher the resolution of the images is the more advantage the second algorithm has. This improvement is due to the fact that it finds linearly independent vectors and significantly depends only on them. Note that the example above works well in the semi-online setting.

Learning a classifier

Online regression algorithms are often applied in the batch setting, when one has a training set with input vectors and their labels and a test set containing just input vectors. In this case the semi-online setting does not appear as a drawback.

Online regression methods can be used to learn a linear classifier, for example Perceptron. Cesa-Bianchi et al. (2005) use the AAR to make an algorithm to train a Perceptron and to derive upper bounds on the number of mistakes during the training process. They consider both linear classification and classification in RKHS. We show that the combination of our preprocessing steps and their algorithm allows us to learn a classifier in PBFS.

Let $(x_1, y_1), \ldots, (x_T, y_T)$ be a set of training examples, where $x_i \in \mathbb{R}^n$ is an input vector and $y_i = \{-1, 1\}$ is its label, $i = 1, \ldots, T$. The label corresponds to the class of the input vector. For any fixed margin $\gamma > 0$ and pair (x, y) define the hinge loss

$$D_\gamma(f, (x, y)) := \max\{0, \gamma - yf(x)\}$$

of any function f from a Sobolev space $W_p^s, s > 0, p > 1$. If we perform the preprocessing steps described in the proofs of Theorems 5.1 and 5.4, we will obtain the vectors $r_1, \ldots, r_T \in \mathbb{R}^n$ corresponding to our input vectors, for some $n \leq T$.

Let $a > 0$ be the parameter of the second-order Perceptron algorithm (see

Cesa-Bianchi et al., 2005, Figure 3.1). At step t it predicts

$$\widehat{y}_t = \operatorname{sign}\left[\left(\sum_{i\in\mathcal{M}_t} y_i r_i\right)' \left(aI_n + \sum_{i\in\mathcal{M}_t} r_i r_i'\right)^{-1} r_t\right],$$

where $\mathcal{M}_t \subseteq \{1,2,\ldots\}$ is the set of indices of mistaken steps ($y_i \neq \widehat{y}_i, i \in \mathcal{M}_t$) before the step t.

We prove the following upper bound on the number of mistakes.

Theorem 5.5 *It is possible to run the second-order Perceptron Algorithm on any finite sequence $(x_1, y_1), \ldots, (x_T, y_T)$ of examples such that the number k of mistakes satisfies*

$$k \le \inf_{\gamma>0} \min_{f\in W_p^s} \left(\frac{R^2(f,a,T)}{2\gamma^2} + \frac{D_\gamma^T(f)}{\gamma} + \frac{R(f,a,T)}{\gamma}\sqrt{\frac{D_\gamma^T(f)}{\gamma} + \frac{R^2(f,a,T)}{4\gamma^2}}\right),$$

where $R^2(f,a,T) = c_{W_p^s}^2 \left(T^{|1/2-1/p|} K \|f\|_{W_p^s}^2 + \frac{1}{a}\sum_{i\in\mathcal{M}_T} f^2(x_i)\right)$, $D_\gamma^T(f)$ is the cumulative hinge loss $\sum_{i=1}^T D_\gamma(f,(x_i,y_i))$ of f, and $K = \|U\|\|U^{-1}\|$ for the isomorphism U defined by (5.7).

Proof After the preprocessing steps we obtain the images $r_1, \ldots, r_T \in \mathbb{R}^n$ of the input vectors. Let by X_k denote the matrix consisting of the columns of r_i with $i \in \mathcal{M}_T$. The second-order Perceptron algorithm run on the sequence $(r_1, y_1), \ldots, (r_T, y_T)$ achieves (see Theorem 1 in Cesa-Bianchi et al., 2005) for any $g \in \mathbb{R}^n$

$$k \le \inf_{\gamma>0} \min_{g\in\mathbb{R}^n} \left(\frac{\tilde{D}_\gamma^T(g)}{\gamma} + \frac{1}{\gamma}\sqrt{(a\|g\| + g' X_k X_k' g)\sum_{i=1}^n \ln(1+\lambda_i/a)}\right),$$

where λ_i are the eigenvalues of the matrix $X_k X_k'$ and $\tilde{D}_\gamma^T(g)$ is the cumulative hinge loss of the linear function g on this sequence.

Clearly, since g is linear, we have $g' X_k X_k' g = \sum_{i\in\mathcal{M}_T} g^2(r_i)$. The sum of the eigenvalues of a square matrix is equal to the trace of it. At the same time, the nonzero eigenvalues of $X_k X_k'$ coincide with the nonzero eigenvalues of $X_k' X_k$. We first use $\ln(1+x) \le x$ for $x \ge 0$ and bound the trace $\sum_{t\in\mathcal{M}} \|r_t\|^2$

of the matrix $X_k' X_k$ by $k \max_{t \in \mathcal{M}} \|r_t\|^2$. Then we obtain

$$k \leq \inf_{\gamma > 0} \min_{g \in \mathbb{R}^n} \left(\frac{\tilde{D}_\gamma^T(g)}{\gamma} + \frac{1}{\gamma} \sqrt{\left(\|g\| + \frac{1}{a} \sum_{i \in \mathcal{M}_T} g^2(r_i) \right) k \max_{t \in \mathcal{M}} \|r_t\|^2} \right).$$

Denote $\tilde{R}^2(g, a, T) := \left(\|g\| + \frac{1}{a} \sum_{i \in \mathcal{M}_T} g^2(r_i) \right) \max_{t \in \mathcal{M}} \|r_t\|^2$. Solving the inequality in k, we obtain

$$k \leq \inf_{\gamma > 0} \min_{g \in \mathbb{R}^n} \left(\frac{\tilde{R}^2(g, a, T)}{2\gamma^2} + \frac{\tilde{D}_\gamma^T(g)}{\gamma} + \frac{\tilde{R}(g, a, T)}{\gamma} \sqrt{\frac{\tilde{D}_\gamma^T(g)}{\gamma} + \frac{\tilde{R}^2(g, a, T)}{4\gamma^2}} \right).$$

Recall that due to the fact that r_i are the images of the input vectors, we can say that $\|r_i\| \leq c_{W_p^s}$. Following the same arguments as in the proofs of Theorems 5.1 and 5.4, we can ensure that for any $f \in W_p^s$ there exists $g \in \mathbb{R}^n$ such that $f(x_i) = g(r_i)$ for all $i = 1, \ldots, T$, and $\|g\| \leq K T^{|1/2 - 1/p|} \|f\|_{W_p^s}$ for the constant K defined from the isomorphism (5.7). This concludes the proof.■

Note that if the algorithm is run on the same sequence of input vectors several times, then T does not increase with every new run.

5.1.5 Discussion

Our results have more theoretical value rather than of real practical use: we are not aware of any ways to perform several steps of our algorithms in practice (among them the main optimization step, the search for the linearly independent subsets, the identifications of the input vectors). The semi-online setting requires the knowledge of the input vectors and the horizon of the prediction in advance. Both these restrictions weaken the result if considered within the online prediction concept. We believe that there exists a better way to deal with the problem of competing with Banach spaces, but our research may give some insights of how to approach it.

The idea of competing with infinite dimensional Banach spaces was approached by Vovk (2006, 2007) using two different ways. The first technique is based on the game-theoretic probability theory (Shafer and Vovk, 2001) and the algorithm is called BBK29. The second technique is based on the metric

entropy of the space with which the learner wishes to compete. The Aggregating Algorithm is used for prediction. Suppose that input vectors are taken from a domain $\mathbf{X} \subseteq \mathbb{R}^n$. The main difference in the upper bounds for two algorithms can be described on the example of Slobodetsky spaces $B_p^s(\mathbf{X}) = B_{p,p}^s(\mathbf{X})$. We always assume that $sp > n$: this condition ensures that the elements of B_p^s are continuous functions on $\mathbf{X}$ (see, e.g., Triebel, 1978). Assuming $p \geq 2$, the known upper bound on the regret term is of order $O(T^{1-1/p})$ when the learner uses either the BBK29 or our algorithm from Theorem 5.4 to predict the outcomes. This order does not depend on s. The order $O(T^{n/(n+s)})$ is provided by the Metric Entropy technique. This order does not depend on p and so this algorithm can be applied to compete with spaces with $p = 1$.

The question asked by Vovk (2006) is whether it is possible to create an algorithm which will have an upper bound such that the order of its regret term involves both parameters p and s and is better than the regret terms for the existing algorithms. Our paper gives another way to apply the Aggregating Algorithm, and the order of the regret term corresponds to the order given by the BBK29, $O(T^{1-1/p})$. Thus we reduced (in a sense) the problem to the analysis of two different ways of using the Aggregating Algorithm to compete with functions from Banach spaces.

Chapter 6

Conclusion and directions of future research

In this thesis we suggested a methodology to approach competitive online prediction problems. Two algorithms are described: the Aggregating Algorithm and the Defensive Forecasting algorithm. The way which is used by the AA to give predictions can be easily represented geometrically. This makes the AA more intuitive than the DF. On the other hand, the DF can be naturally applied to a wider class of problems. That is why it may often be handy to derive an algorithm using Defensive Forecasting, but then rephrase it in terms of the Aggregating Algorithm (in this thesis this can be done with the PPIR).

We have shown that in simple cases with square and log loss functions, the AA and the DF have the same guarantees on their performance and even their predictions are the same if a particular substitution function is used in the AA.

The natural idea is to consider second-guessing experts. This setting allows the experts to give the actual prediction after the prediction of the algorithm is made. We have shown that in this setting, the DF and the AA achieve the same performance guarantees on their loss.

The outcomes of the predicted sequence can be assumed to belong to a probability simplex. The important problem of classification under the Brier loss can be solved under this setting. By considering prediction the results of sports matches we have shown that the performance guarantees can be

tight in practice. The algorithms which do not have guarantees can fail to produce reliable performance. Moreover, if the performance of an algorithm is measured by a different loss function than the one it was trained with, it can fail to satisfy strict performance requirements. In this case the DF can be useful, because it is able to compete under several loss functions at the same time.

We have shown that online prediction algorithms (in particular, the AA for the Brier game) can be beneficial to use for predicting ovarian cancer. By combining different prediction rules, it achieves comparable or better performance than the best rule chosen in hindsight.

In Chapter 3 we described a different class of problems. These are online regression problems, where each expert follows a determined prediction strategy, which depends on an input vector at each step. We proved a general lemma, which helps to provide guarantees for different algorithms for online regression. Using this lemma, we proved the guarantees for the Aggregating Algorithm for Regression. It competes with all linear functions of input vectors under the assumption that the outcomes lie in an interval.

An assumption which is often made in statistics is that the outcomes are corrupted by Gaussian noise. We considerd the problem of competing with Gaussian linear experts under the logarithmic loss function. We have shown that applying the Aggregating Algorithm leads to the derivation of Bayesian Ridge Regression and proved guarantees on its cumulative logarithmic loss. These guarantees are easily transformed to an equality on the square loss of Ridge Regression. We have shown that the equality leads to an upper bound on the cumulative square loss of Ridge Regression competing with linear experts (under the assumption that the outcomes are bounded).

If the outcomes are not bounded uniformly, in other words, the bounds for the outcomes become known only in the beginning of each prediction step, it is possible to apply the DF and prove guarantees on the square loss of the derived algorithm. We investigated the performance of the algorithm on artificial and real world data sets and found that the algorithm can be used in conjunction with an interval predictor to point prediction intervals. We have shown that this problem closely relates to the problem of competing under the discounted

square loss. We solved both problems using the same approach.

Linear experts are not very efficient when it is known that the outcomes lie in an interval. We generalized the class of the experts and considered generalized linear experts. Generalized linear models are often used for classification. Using the AA, we derived an algorithm competing with the experts under the square loss. We investigated the performance of the algorithm in practice and showed that it can outperform known benchmark batch and online algorithms.

If the outcomes lie in a multi-dimensional probability simplex (for example, in the case of multi-class classification), it is possible to consider the multi-dimensional regression problems. We derived two algorithms competing with linear experts under the Brier loss, proved performance guarantees for them, and investigated their performance in practice. The experiments show that the algorithms can be as precise as benchmark online methods, and at the same time require less training time.

For all the algorithms from Chapter 3, we derive their kernelized versions. They are capable of competing with functions from Reproducing Kernel Hilbert Spaces. Many of them are based on the kernel trick. We showed that if the kernel trick cannot be directly applied, as for the problem with generalized linear models, it is possible to derive an algorithm competing with functions from RKHS using the Bayesian non-parametric approach with Gaussian processes.

We approached the problem of competing with functional Banach spaces through the intermediate abstract linear regression framework. Facts from functional analysis allowed us to find a correspondence between linear regression in abstract Banach spaces and linear regression in the Euclidean spaces in the semi-online setting.

Future research

We see the following interesting future directions of the research.

- The algorithm competing under the discounted loss has the property of tracking the best expert (or the best linear predictor). It is very interesting to investigate the Tracking the Best Linear Predictor framework

(Herbster and Warmuth, 2001) directly using the methodology suggested in this thesis.

- It may be possible to use methods developed by van der Vaart and van Zanten (2008) or by Zhang et al. (2009) to compete with infinite-dimensional functional Banach spaces in a different way than the one which is suggested in this thesis. The first step may be to investigate the extent to which the smoothness of the functions in an RKHS influences the performance guarantees for the algorithms competing with the RKHS.

- Many of the algorithms suggested in this thesis can be applied in practice. The investigation of their empirical properties is an important possible direction of the future research.

- In this thesis we paid special attention to the square (and somewhere logarithmic) loss function. Another loss function which is often used in practice is the absolute loss. It may be possible to develop methods based on the Weak Aggregating Algorithm (Kalnishkan and Vyugin, 2005) suggested for the finite and countable number of experts to compete with larger classes of experts.

- The algorithm which competes with generalized linear models under the square loss requires the application of the MCMC technique to numerical integration. It may be useful to find approximations of the predictions similar to the ones which are used in standard generalized linear models (such as Iteratively Reweighted Least Squares) to develop more efficient algorithms.

Appendix A

Lemmas from linear algebra

Lemma A.1 *Let $Q(\theta) = \theta' A\theta + b'\theta + c$, where $\theta, b \in \mathbb{R}^n$, c is a scalar, and 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 proof of this lemma can be found in Harville (1997, Theorem 15.12.1).

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 This lemma is proven by taking the derivative of the quadratic forms in F in θ and calculating the minimum: $\min_{\theta \in \mathbb{R}^n} (\theta' A\theta + c'\theta) = -\frac{(A^{-1}c)'}{4} c$ for any $c \in \mathbb{R}^n$ (see Harville, 1997, Theorem 19.1.1). ■

Lemma A.3 *Let $Q_1(\theta) = \theta' A\theta + b_1'\theta + c_1$, $Q_2(\theta) = \theta' A\theta + b_2'\theta + c_2$, where $\theta, b_1, b_2 \in \mathbb{R}^n$, c_1, c_2 are scalars, and A is a symmetric positive definite $n \times n$ matrix. Then*

$$\frac{\int_{\mathbb{R}^n} e^{-Q_1(\theta)} d\theta}{\int_{\mathbb{R}^n} e^{-Q_2(\theta)} d\theta} = e^{c_2 - c_1 - \frac{1}{4}(b_2 + b_1)' A^{-1} (b_2 - b_1)}$$

PROOF After evaluating each of the integrals using Lemma A.1 the ratio is represented as follows:

$$\frac{\int_{\mathbb{R}^n} e^{-Q_1(\theta)} d\theta}{\int_{\mathbb{R}^n} e^{-Q_2(\theta)} d\theta} = e^{\min_{\theta\in\mathbb{R}^n} Q_2(\theta) - \min_{\theta\in\mathbb{R}^n} Q_1(\theta)} \ .$$

The difference of minimums can be calculated using Lemma A.2 with $b = \frac{b_2+b_1}{2}$ and $z = \frac{b_2-b_1}{2}$:

$$\min_{\theta\in\mathbb{R}^n} Q_2(\theta) - \min_{\theta\in\mathbb{R}^n} Q_1(\theta) = c_2 - c_1 - \frac{1}{4}(b_2+b_1)'A^{-1}(b_2-b_1) \ . \qquad \blacksquare$$

Lemma A.4 *For any $n \times m$ matrix B, any $m \times n$ matrix C, and any number a such that $aI + CB$ and $aI + BC$ are nonsingular, we have*

$$B(aI_n + CB)^{-1} = (aI_m + BC)^{-1}B, \tag{A.1}$$

where I_n, I_m are unit matrices $n \times n$ and $m \times m$, respectively.

PROOF This is equivalent to $(aI_n + BC)B = B(aI_m + CB)$. That is true because of distributivity of matrix multiplication. $\blacksquare$

Lemma A.5 (Matrix Determinant Identity) *For any $n \times m$ matrix B, any $m \times n$ matrix C, and any number a*

$$\det(aI_n + BC) = \det(aI_m + CB), \tag{A.2}$$

where I_n, I_m are unit matrices $n \times n$ and $m \times m$, respectively.

PROOF It follows from matrix multiplication rules that

$$\begin{pmatrix} I_n & B \\ 0 & I_m \end{pmatrix} \begin{pmatrix} aI_n + BC & 0 \\ -C & aI_m \end{pmatrix} = \begin{pmatrix} aI_n & aB \\ -C & aI_m \end{pmatrix}$$
$$= \begin{pmatrix} aI_n & 0 \\ -C & aI_m + CB \end{pmatrix} \begin{pmatrix} I_n & B \\ 0 & I_m \end{pmatrix}$$

Taking the determinant of both sides and using rules of rules of taking the determinant of block matrices we get the statement of the lemma. $\blacksquare$

Lemma A.6 *Let $x_t \in \mathbb{R}^n$ be a set of column vectors, $t = 1, \ldots, T$. Then for any $a > 0$,*

$$\det\left(I + \frac{1}{a}\sum_{t=1}^{T} x_t x_t'\right) = \prod_{t=1}^{T}(1 + x_t' A_{t-1}^{-1} x_t)$$

for $A_t = aI + \sum_{i=1}^{t} x_i x_i'$.

PROOF This fact can be proven by induction in T: for $T = 0$ it is obvious $(1 = 1)$ and for $T \geq 1$ we have

$$\begin{aligned}\det\left(I + \frac{1}{a}\sum_{t=1}^{T} x_t x_t'\right) &= a^{-n} \det A_T = a^{-n} \det\left(A_{T-1} + x_T x_T'\right) \\ &= a^{-n}(1 + x_T' A_{T-1}^{-1} x_T) \det A_{T-1} \\ &= \det\left(I + \frac{1}{a}\sum_{t=1}^{T-1} x_t x_t'\right)(1 + x_T' A_{T-1}^{-1} x_T) = \prod_{t=1}^{T}(1 + x_t' A_{t-1}^{-1} x_t).\end{aligned}$$

The third equality follows from the Matrix Determinant Identity. The last equality follows from the inductive assumption. ■

Appendix B

Additional online resources

Online prediction web-site It is very convenient to use a web-site which gathers information about online prediction and related areas. We set up the web-site `http://onlineprediction.net/`. This is a wiki-type web-site accessible for reading and editing to everyone. The web-site has two major purposes:

- Provide overviews of various research topics, for students or people who are new to the area.
- Provide descriptions of open questions, for people who wish to further develop the area.

The LaTeX syntax can be used for formulas. We encourage everyone to participate in editing and improving the content of the web-site.

There are both articles about theory and articles about experimental results (we try to separate theory and experiments, covering them in different, often cross-referenced, articles).

Prediction with expert advice package We implemented the most interesting algorithms used in this thesis and made them accessible for the public use. The Matlab implementation of the algorithms used in Sections 2.6 and 2.7 is accessible[1]. The following algorithms are implemented: the Aggregating Al-

[1] `http://www.mathworks.com/matlabcentral/fileexchange/28131-prediction-with-expert-advice`

gorithm (and Algorithm 4), the Defensive Forecasting algorithm, the Weighted Average Algorithm, the Weak Aggregating Algorithm, and the Follow the Best Expert Algorithm. File examplepredict.m contains an example of use, predicting results of tennis matches using bookmakers' predictions.

Online/Batch generalized linear models under square loss The Matlab implementation of the new algorithms used in Section 3.5 (AAGLM) is accessible[2]. The algorithms have guarantees on the cumulative square loss for the worst case when applied in online fashion. The variable regressed should lie in $[0, 1]$, thus the program is a tool for two-class classification or for bounded regression. Four possibilities are provided: linear regression, logistic regression, probit regression, comlog regression. Other functions can be easily added and used. File examplepredict.m contains an example of use.

[2] http://www.mathworks.com/matlabcentral/fileexchange/28251-onlinebatch-generalized-linear-models-under-square-loss

Bibliography

J. Aczél. *Lectures on Functional Equations and Their Applications*, volume 19 of *Mathematics in Science and Engineering*. New York, 1966.

Robert A. Adams and John J. F. Fournier. *Sobolev Spaces*, volume 140 of *Pure and Applied Mathematics*. Academic Press (Elsevier), Amsterdam, second edition, 2003.

R. Agarwal, M. Meehan, and D. O'Regan. *Fixed Point Theory and Applications*, volume 141 of *Cambridge Tracts in Mathematics*. Cambridge University Press, Cambridge, England, 2001.

A. Aizerman, E. M. Braverman, and L. I. Rozoner. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

Christophe Andrieu, Nando de Freitas, Arnaud Doucet, and Michael Jordan. An introduction to MCMC for machine learning. *Machine Learning*, 50: 5–43, 2003.

Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68, 1950.

Nachman Aronszajn, Fuad Mulla, and Pawel Szeptycki. On spaces of potentials connected with L^p classes. *Annales de l'Institut Fourier (Grenoble)*, 13:211–306, 1963.

Katy S. Azoury and Manfred K. Warmuth. Relative loss bounds for on-line density estimation with the exponential family of distributions. *Machine Learning*, 43:211–246, 2001.

Arindam Banerjee. On Bayesian bounds. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 81–88, 2006.

Arindam Banerjee. An analysis of logistic models: exponential family connections and online performance. In *Proceedings of the 2007 SIAM International Conference on Data Mining*, pages 204–215, 2007.

G. A. Barnard. New methods of quality control. *Journal of the Royal Statistical Society. Series A (General)*, 126:255–258, 1963.

Edwin F. Beckenbach and Richard Bellman. *Inequalities*. Springer, Berlin, 1961.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, 2006.

Avrim Blum and Yishay Mansour. From external to internal regret. *Journal of Machine Learning Research*, 8:1307–1324, 2007.

Glenn W. Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78:1–3, 1950.

Robert G. Brown. *Smoothing, Forecasting and Prediction of discrete time series*. Prentice-Hall, Englewood Cliffs, NJ, 1963.

Steven Busuttil. *The Aggregating Algorithm and Regression*. PhD thesis, Department of Computer Science, Royal Holloway University of London, UK, 2008.

Gavin C. Cawley, Gareth J. Janacek, and Nicola L. C. Talbot. Generalised kernel machines. In *Proceedings of the International Joint Conference on Neural Networks*, pages 1720–1725, 2007.

Nicolò Cesa-Bianchi, Alex Conconi, and Claudio Gentile. A second-order Perceptron algorithm. *SIAM Journal on Computing*, 34:640–668, 2005.

Nicolò Cesa-Bianchi, Philip M. Long, and Manfred K. Warmuth. Worst-case quadratic loss bounds for on-line prediction of linear functions by gradient descent. *IEEE Transactions on Neural Networks*, 7:604–619, 1996.

Nicolò Cesa-Bianchi and Gábor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, Cambridge, UK, 2006.

Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001.

Alexey Chernov, Yuri Kalnishkan, Fedor Zhdanov, and Vladimir Vovk. Supermartingales in prediction with expert advice. In *Proceedings of the 19th International Conference on Algorithmic Learning Theory*, pages 199–213. Springer, Berlin, 2008.

Alexey Chernov, Yuri Kalnishkan, Fedor Zhdanov, and Vladimir Vovk. Supermartingales in prediction with expert advice. *Theoretical Computer Science*, 411:2647–2669, 2010.

Alexey Chernov and Vladimir Vovk. Prediction with expert evaluators' advice. In *Proceedings of the 20th International Conference on Algorithmic Learning Theory*, pages 8–22. Springer, Berlin, 2009.

Alexey Chernov and Fedor Zhdanov. Prediction with expert advice under discounted loss. In *Proceedings of the 21st International Conference on Algorithmic Learning Theory*, pages 255–269, 2010.

Thomas M. Cover. Universal portfolios. *Mathematical Finance*, 1:1–29, 1991.

Arnak Dalalyan and Alexandre B. Tsybakov. Sparse regression learning by aggregation and Langevin Monte-Carlo. In *Proceedings of the 22nd Annual Conference on Computational Learning Theory*, 2009.

A. Philip Dawid. Probability forecasting. In Samuel Kotz, Norman Johnson, and Campbell Read, editors, *Encyclopedia of Statistical Sciences*, volume 7, pages 210–218. Wiley, New York, 1986.

Alfredo DeSantis, George Markowsky, and Mark N. Wegman. Learning probabilistic prediction functions. In *Proceedings of the 29th Annual IEEE Symposium on Foundations of Computer Science*, pages 110–119, Los Alamitos, CA, USA, 1988. IEEE Computer Society.

Dmitry Devetyarov, Ilia Nouretdinov, Brian Burford, Zhiyuan Luo, Alexey Chervonenkis, Vladimir Vovk, Mike Waterfield, Ali Tiss, Celia Smith, Rainer Cramer, Alex Gentry-Maharaj, Rachel Hallett, Stephane Camuzeaux, Jeremy Ford, John Timms, Usha Menon, Ian Jacobs, and Alex Gammerman. Analysis of serial UKCTOCS-OC data: discriminating abilities of proteomics peaks. *Technical report, http://clrc.rhul.ac.uk/projects/proteomic3.htm*, 2009.

Dean P. Foster. Prediction in the worst case. *The Annals of Statistics*, 19: 1084–1090, 1991.

Dean P. Foster and Rakesh Vohra. Regret in the on-line decision problem. *Games and Economic Behavior*, 29:7–35, 1999.

Yoav Freund and Daniel Hsu. A new hedging algorithm and its application to inferring latent random variables. *Technical report, arXiv:0806.4802 [cs.GT], arXiv.org e-Print archive*, 2008.

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55:119–139, 1997.

Yoav Freund, Robert E. Schapire, Yoram Singer, and Manfred K. Warmuth. Using and combining predictors that specialize. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, pages 334–343, 1997.

Alex Gammerman, Vladimir Vovk, Brian Burford, Ilia Nouretdinov, Zhiyuan Luo, Alexey Chervonenkis, Mike Waterfield, Rainer Cramer, Paul Tempst, Josep Villanueva, Musarat Kabir, Stephane Camuzeaux, John Timms, Usha Menon, and Ian Jacobs. Serum Proteomic Abnormality Predating Screen Detection of Ovarian Cancer. *The Computer Journal*, 2008. bxn021.

Alexander Gammerman, Yuri Kalnishkan, and Vladimir Vovk. On-line prediction with kernels and the complexity approximation principle. In *Proceedings of the 20th Conference in Uncertainty in Artificial Intelligence*, pages 170–176, 2004.

Everette S. Gardner. Exponential smoothing: The state of the art – part II. *International Journal of Forecasting*, 22:637–666, 2006.

Carl F. Gauss. *Theoria Motus Corporum Coelestium in Sectionibus Conicis Solem Ambientum*. 1809.

Claudio Gentile and Nick Littlestone. The robustness of the p-norm algorithms. In *Proceedings of the 12th Annual Conference on Computational Learning Theory*, pages 1–11, New York, NY, USA, 1999. ACM.

R. M. Griffith. Odds adjustments by american horse-race bettors. *The American Journal of Psychology*, 62:290–294, 1949.

Adam J. Grove, Nick Littlestone, and Dale Schuurmans. General convergence results for linear discriminant updates. In *Proceedings of the 10th Annual Conference on Computational Learning Theory*, pages 171–183, New York, NY, USA, 1997. ACM.

James D. Hamilton. *Time Series Analysis*. Princeton University Press, Princeton, NJ, 1994.

David A. Harville. *Matrix Algebra From a Statistician's Perspective*. Springer, New York, 1997.

David Haussler, Jyrki Kivinen, and Manfred K. Warmuth. Sequential prediction of individual sequences under general loss functions. *IEEE Transactions on Information Theory*, 44:1906–1925, 1998.

Elad Hazan, Amit Agarwal, and Satyen Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69:169–192, 2007.

Mark Herbster and Manfred K. Warmuth. Tracking the best expert. *Machine Learning*, 32:151–178, 1998.

Mark Herbster and Manfred K. Warmuth. Tracking the best linear predictor. *Journal of Machine Learning Research*, 1:281–309, 2001.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58:13–30, 1963.

Arthur E. Hoerl. Application of ridge analysis to regression problems. *Chemical Engineering Progress*, 58:54–59, 1962.

Arthur E. Hoerl and Robert W. Kennard. Ridge Regression: biased estimation for nonorthogonal problems. *Technometrics*, 42:80–86, 2000.

Fritz John. Extremum problems with inequalities as subsidiary conditions. In *Courant Anniversary Volume*, pages 187–204. Interscience Publishers, Inc., New York, N. Y., 1948.

Sham M. Kakade and Andrew Y. Ng. Online bounds for Bayesian algorithms. In *Proceedings of the 18th Annual Conference on Neural Information Processing Systems*, 2004.

Sham M. Kakade, Matthias W. Seeger, and Dean P. Foster. Worst-case bounds for Gaussian process models. In *Proceedings of the 19th Annual Conference on Neural Information Processing Systems*, 2005.

Yuri Kalnishkan and Michael V. Vyugin. The weak aggregating algorithm and weak mixability. In *Proceedings of the 18th Annual Conference on Computational Learning Theory*, pages 188–203. Springer, Berlin, 2005.

Yuri Kalnishkan and Michael V. Vyugin. The weak aggregating algorithm and weak mixability. *Journal of Computer and System Sciences*, 74:1228–1244, 2008.

Victor Khutsishvili. Personal communication, E-mail exchanges (from 27 November 2008), 2009.

Jyrki Kivinen and Manfred K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132:1–63, 1997.

Jyrki Kivinen and Manfred K. Warmuth. Averaging expert predictions. In *Proceedings of the 4th European Conference on Computational Learning Theory*, pages 153–167. Springer, Berlin, 1999.

Jyrki Kivinen and Manfred K. Warmuth. Relative loss bounds for multidimensional regression problems. *Machine Learning*, 45:301–329, 2001.

Jyrki Kivinen and Manfred K. Warmuth. Online learning with kernels. *IEEE Transactions on Signal Processing*, 52:2165–2176, 2004.

Masayuki Kumon, Akimichi Takemura, and Kei Takeuchi. Sequential optimizing strategy in multi-dimensional bounded forecasting games. *Technical report, arXiv:0911.3933 [math.PR], arXiv.org e-Print archive*, 2009.

Adrien M. Legendre. *Nouvelles méthodes pour la détermination des orbites des comètes*. Courcier, Paris, 1805. Appendix on least squares.

Leonid Levin. Uniform tests of randomness. *Soviet Mathematics Doklady*, 17: 337–340, 1976.

D. R. Lewis. Finite dimensional subspaces of L_p. *Studia Mathematica*, 63: 207–212, 1978.

Joram Lindenstrauss and Lior Tzafriri. *Classical Banach spaces. II*, volume 97 of *Ergebnisse der Mathematik und ihrer Grenzgebiete [Results in Mathematics and Related Areas]*. Springer, Berlin, 1979.

Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108:212–261, 1994.

Peter McCullagh and John Nelder. *Generalized Linear Models*. Chapman & Hall/CRC, second edition, 1989.

Usha Menon, Steven J. Skates, Sara Lewis, Adam N. Rosenthal, Barnaby Rufford, Karen Sibley, Nicola MacDonald, Anne Dawnay, Arjun Jeyarajah, Jr Bast, Robert C., David Oram, and Ian J. Jacobs. Prospective study using the risk of ovarian cancer algorithm to screen for ovarian cancer. *Journal of Clinical Oncology*, 23:7919–7926, 2005.

Christian Michelot. A finite algorithm for finding the projection of a point onto the canonical simplex of $\mathbb{R}^n$. *Journal of Optimization Theory and Applications*, 50:195–200, 1986.

Tom M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

Peter A. Morris. Decision analysis expert use. *Management Science*, 20:1233–1241, 1974.

John F. Muth. Optimal properties of exponentially weighted forecasts. *Journal of the American Statistical Association*, 55:299–306, 1960.

Radford M. Neal. Regression and classification using Gaussian process priors. In J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, editors, *Bayesian Statistics 6*, pages 475–501. Oxford University Press, 1999.

Radford M. Neal. Slice sampling. *The Annals of Statistics*, 31:705–741, 2003.

Aleksander Pełczyński and Michał Wojciechowski. Sobolev spaces. In *Handbook of the Geometry of Banach spaces, Vol. 2*, pages 1361–1423. North-Holland, Amsterdam, 2003.

David M. Pennock, Steve Lawrence, Finn Arup Nielsen, and C. Lee Giles. Extracting collective probabilistic forecasts from web games. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 174–183, 2001.

William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical recipes in C*. Cambridge University Press, Cambridge, second edition, 1992.

Cazhaow S. Qazaz, Christopher K. I. Williams, and Christopher M. Bishop. An upper bound on the Bayesian error bars for generalized linear regression. In *Proceedings of the First International Conference on Mathematics of Neural Networks: Models, Algorithms and Applications*, pages 295–299, 1997.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, Cambridge, MA, 2006.

Luke G. Rogers. Degree-independent Sobolev extension on locally uniform domains. *Journal of Functional Analysis*, 235:619–665, 2006.

Raymond D. Sauer. The state of research on markets for sports betting and suggested future directions. *Journal of Economics and Finance*, 29:416–426, 2005.

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2002.

Matthias W. Seeger, Sham M. Kakade, and Dean P. Foster. Information consistency of nonparametric Gaussian process methods. *IEEE Transactions on Information Theory*, 54:2376–2382, 2008.

Glenn Shafer and Vladimir Vovk. *Probability and Finance: It's Only a Game!* Wiley-Interscience, New York, 2001.

Steven J. Skates, Usha Menon, Nicola MacDonald, Adam N. Rosenthal, David H. Oram, Robert C. Knapp, and Ian J. Jacobs. Calculation of the Risk of Ovarian Cancer from serial CA-125 values for preclinical detection in postmenopausal women. *Journal of Clinical Oncology*, 21:206–210, 2003.

Erik Snowberg and Justin Wolfers. Explaining the favorite-longshot bias: Is it risk-love or misperceptions? *Available on-line at http://bpp.wharton.upenn.edu/jwolfers/ (accessed on 4 October 2009)*, 2007.

Nicole Tomczak-Jaegermann. *Banach-Mazur Distances and Finite-dimensional Operator Ideals*, volume 38 of *Pitman Monographs and Surveys in Pure and Applied Mathematics*. Longman Scientific & Technical, Harlow, 1989.

H. Triebel. Sampling numbers and embedding constants. *Trudy Matematicheskogo Instituta im. V.A. Steklova RAN*, 248:275–284, 2005.

Hans Triebel. *Interpolation Theory, Function Spaces, Differential Operators*. North-Holland Pub. Co., Amsterdam, 1978.

Hans Triebel. *Theory of Function Spaces. II*, volume 84 of *Monographs in Mathematics*. Birkhäuser Verlag, Basel, 1992.

Hans Triebel. *Theory of Function Spaces. III*, volume 100 of *Monographs in Mathematics*. Birkhäuser Verlag, Basel, 2006.

A. W. van der Vaart and J. H. van Zanten. Rates of contraction of posterior distributions based on Gaussian process priors. *Annals of Statistics*, 36: 1435–1463, 2008.

Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, 1995.

Vladimir Vovk. Aggregating strategies. In *Proceedings of the Third Annual Workshop on Computational Learning Theory*, pages 371–383, San Mateo, CA, 1990. Morgan Kaufmann.

Vladimir Vovk. A game of prediction with expert advice. *Journal of Computer and System Sciences*, 56:153–173, 1998.

Vladimir Vovk. Derandomizing stochastic prediction strategies. *Machine Learning*, 35:247–282, 1999.

Vladimir Vovk. Competitive on-line statistics. *International Statistical Review*, 69:213–248, 2001.

Vladimir Vovk. On-line regression competitive with reproducing kernel Hilbert spaces. *Technical report, arXiv:cs/0511058 [cs.LG], arXiv.org e-Print archive*, 2005.

Vladimir Vovk. Metric entropy in competitive on-line prediction. *Technical report, arXiv:cs/0609045 [cs.LG], arXiv.org e-Print archive*, 2006.

Vladimir Vovk. Competing with wild prediction rules. *Machine Learning*, 69: 193–212, 2007.

Vladimir Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer, New York, 2005.

Vladimir Vovk and Fedor Zhdanov. Prediction with expert advice for the Brier game. *Technical report, arXiv:0710.0485 [cs.LG], arXiv.org e-Print archive*, 2007.

Vladimir Vovk and Fedor Zhdanov. Prediction with expert advice for the Brier game. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1104–1111, 2008.

Vladimir Vovk and Fedor Zhdanov. Prediction with expert advice for the Brier game. *Journal of Machine Learning Research*, 10:2413–2440, 2009.

Haizhang Zhang, Yuesheng Xu, and Jun Zhang. Reproducing Kernel Banach Spaces for machine learning. *Journal of Machine Learning Research*, 10: 2741–2775, 2009.

Kun Zhang and Wei Fan. Forecasting skewed biased stochastic ozone days: analyses, solutions and beyond. *Knowledge and Information Systems*, 14: 299–326, 2008.

Fedor Zhdanov, Alexey Chernov, and Yuri Kalnishkan. Aggregating Algorithm competing with Banach lattices. *Technical report, arXiv:1002.0709 [cs.LG], arXiv.org e-Print archive*, 2010.

Fedor Zhdanov and Yuri Kalnishkan. Linear probability forecasting. *Technical report, arXiv:1001.0879 [cs.LG], arXiv.org e-Print archive*, 2009.

Fedor Zhdanov and Yuri Kalnishkan. An identity for Kernel Ridge Regression. In *Proceedings of the 21st International Conference on Algorithmic Learning Theory*, pages 405–419, 2010a.

Fedor Zhdanov and Yuri Kalnishkan. Linear probability forecasting. In *Proceedings of the 6th IFIP Conference on Artificial Intelligence Applications and Innovations*, pages 4–11, 2010b.

Fedor Zhdanov and Vladimir Vovk. Competing with Gaussian linear experts. *Technical report, arXiv:0910.4683 [cs.LG], arXiv.org e-Print archive*, 2009.

Fedor Zhdanov and Vladimir Vovk. Competitive online generalized linear regression under square loss. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases 2010*, pages 531–546, 2010.

Fedor Zhdanov, Vladimir Vovk, Brian Burford, Dmitry Devetyarov, Ilia Nouretdinov, and Alex Gammerman. Online prediction of ovarian cancer. In *Proceedings of the 12th Conference on Artificial Intelligence in Medicine*, pages 375–379, 2009a.

Fedor Zhdanov, Vladimir Vovk, Brian Burford, Dmitry Devetyarov, Ilia Nouretdinov, and Alex Gammerman. Online prediction of ovarian cancer. *Technical report, arXiv:0904.1579 [cs.AI], arXiv.org e-Print archive*, 2009b.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning*, pages 928–936, 2003.