Machine Learning
https://doi.org/10.1007/s10994-018-5769-2

Aggregating Algorithm for prediction of packs

Dmitry Adamskiy[1,2] · Anthony Bellotti[3] · Raisa Dzhamtyrova[4] · Yuri Kalnishkan[2,4,5]

Received: 10 December 2017 / Accepted: 25 October 2018

Abstract
This paper formulates a protocol for prediction of packs, which is a special case of on-line prediction under delayed feedback. Under the prediction of packs protocol, the learner must make a few predictions without seeing the respective outcomes and then the outcomes are revealed in one go. The paper develops the theory of prediction with expert advice for packs by generalising the concept of mixability. We propose a number of merging algorithms for prediction of packs with tight worst case loss upper bounds similar to those for Vovk's Aggregating Algorithm. Unlike existing algorithms for delayed feedback settings, our algorithms do not depend on the order of outcomes in a pack. Empirical experiments on sports and house price datasets are carried out to study the performance of the new algorithms and compare them against an existing method.

Keywords Machine learning · On-line learning · Prediction with expert advice · Sport · House prices

Editors: Jesse Davis, Elisa Fromont, Derek Greene, and Bjorn Bringmann.

✉ Yuri Kalnishkan
Yuri.Kalnishkan@rhul.ac.uk

Dmitry Adamskiy
D.Adamskiy@cs.ucl.ac.uk

Anthony Bellotti
A.Bellotti@imperial.ac.uk

Raisa Dzhamtyrova
Raisa.Dzhamtyrova.2015@live.rhul.ac.uk

1 Department of Computer Science, University College London, 66-72, Gower Street, London WC1E 6EA, UK

2 Computer Learning Research Centre, Royal Holloway, University of London, Egham, Surrey TW20 0EX, UK

3 Department of Mathematics, Imperial College London, London SW7 2AZ, UK

4 Department of Computer Science, Royal Holloway, University of London, Egham, Surrey TW20 0EX, UK

5 Laboratory of Advanced Combinatorics and Network Applications, Moscow Institute of Physics and Technology, Institutsky per., 9, Dolgoprudny, Russia 41701

Published online: 07 January 2019

1 Introduction

This paper deals with the on-line prediction protocol, where a learner needs to predict outcomes $\omega_1, \omega_2 \ldots$ occurring in succession. The learner is getting feedback along the way.

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

In this paper, we are concerned with the problem of prediction with expert advice. Suppose that the learner has access to predictions of a number of experts. Before the learner makes a prediction, it can see experts' predictions and its goal is to suffer loss close to that of the retrospectively best expert.

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

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

We develop a theory of prediction with expert advice for packs by extending Aggregating Algorithm (AA) introduced by Vovk (1990, 1998). In Sect. 2.2 and Appendix A, we survey the existing theory of the AA for predicting single outcomes (in our terminology, packs of size one). The theory is based on the concept of *mixability* of prediction environments called games. In Sect. 3, we develop the theory of mixability for games with packs of outcomes. Theorem 1 shows that a game involving packs of K outcomes has the same profile of mixability constants as the original game with single outcomes, but the learning rate divides by K. This observation allows us to handle packs of constant size. However, as discussed above, in really interesting cases the size of the pack varies with time and thus the mixability of the environment varies from step to step. This problem can be approached in different ways resulting in different algorithms with different performance bounds. In Sect. 4, we introduce three *Aggregating Algorithms for prediction of Packs*, AAP-max, AAP-incremental, and AAP-current and obtain worst-case upper bounds for their cumulative loss.

The general theory of the AA (Vovk 1998) allows us to show in Sect. 5 that in some sense our bounds are optimal. In Sect. 5.1, we provide a standalone derivation of a lower bound for the mix-loss framework of Adamskiy et al. (2016). However, the question of optimality for packs is far from being fully resolved and requires further investigation.

As mentioned before, the problem of prediction of packs can be considered as a special case of the delayed feedback problem. This problem has been studied mostly within the framework of on-line convex optimisation (Zinkevich 2003; Joulani et al. 2013; Quanrud and Khashabi 2015). The terminology and approach of on-line convex optimisation is different from ours,

which go back to Littlestone and Warmuth (1994) and were surveyed by Cesa-Bianchi and Lugosi (2006).

The problem of prediction with expert advice for delayed feedback can be solved by running parallel copies of algorithms predicting single outcomes. In Sect. 2.3, we describe the algorithm Parallel Copies, which is essentially BOLD of Joulani et al. (2013) using the Aggregating Algorithm as a base algorithm for single outcomes. The theory of the Aggregating Algorithm implies a worst case upper bound on the loss of Parallel Copies. We see that the regret term multiplies by the maximum delay or pack size as in the existing literature (Joulani et al. 2013; Weinberger and Ordentlich 2002).

The bounds we obtain for our new algorithms are the same (AAP-max and AAP-incremental) or incompatible (AAP-current) with the bound for Parallel Copies. We discuss the bounds in Sect. 5 and then in Sect. 6 carry out an empirical comparison of the performance of the algorithms.

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

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

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

2 Preliminaries and background

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

2.1 Protocols for prediction of packs

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

Outcomes $\omega_1, \omega_2, \ldots \in \Omega$ occur in succession. A *learner* or *prediction strategy* outputs predictions $\gamma_1, \gamma_2, \ldots \in \Gamma$ before seeing respective outcomes. The learner may have access to some side information; we will say that the learner sees *signals* x_t coming from a *signal space* X.

In the classical protocol, the learner makes a prediction (possibly upon using a signal) and then the outcome is immediately revealed. In this paper we consider an extension of

this protocol and allow the outcomes to come in *packs* of possibly varying size. The learner must produce a pack of predictions before seeing the true outcomes. The following protocol summarises the framework.

Protocol 1 (Prediction of packs)

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

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

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

Now let $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_N$ be learners working according to Protocol 1. We will refer to these learners as *experts*. Suppose that on each turn, their predictions are made available to a learner $\mathcal{S}$ as a special kind of side information. The learner then works according to the following protocol.

Protocol 2 (Prediction of packs with expert advice)

FOR $t = 1, 2, \ldots$
  each expert $\mathcal{E}_i$, $i = 1, 2, \ldots, N$, announces
    predictions $\gamma_{t,k}(i) \in \Gamma$, $k = 1, 2, \ldots, K_t$
  learner outputs predictions $\gamma_{t,k} \in \Gamma$, $k = 1, 2, \ldots, K_t$
  nature announces $\omega_{t,k} \in \Omega$, $k = 1, 2, \ldots, K_t$
  each expert $\mathcal{E}_i$, $i = 1, 2, \ldots, N$, suffers
    losses $\lambda(\gamma_{t,k}(i), \omega_{t,k})$, $k = 1, 2, \ldots, K_t$
  learner suffers losses $\lambda(\gamma_{t,k}, \omega_{t,k})$, $k = 1, 2, \ldots, K_t$
ENDFOR

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

The merging strategies we are interested in are computable in some natural sense; we will not make exact statements about computability though. We do not impose any restrictions on experts. In what follows, the reader may substitute the clause 'for all predictions $\gamma_{t,k}(i)$ appearing in Protocol 2' for the more intuitive clause 'for all experts'.

There can be subtle variations of this protocol. Instead of getting all K_t predictions from each expert at once, the learner may be getting predictions for each outcome one by one and making its own before seeing the next set of experts' predictions. For most of our analysis this does not matter, as we will see later. The learner may have to work on each 'pack' of experts' predictions sequentially without even knowing its size in advance. The only thing that matters is that the outcomes come in one go after the learner has finished predicting the pack.

2.2 Packs of size one

For packs of size 1 ($K_t = 1, t = 1, 2, \ldots$), the Aggregating Algorithm (AA) by Vovk (1990, 1998) solves the problem of prediction with expert advice in a very general sense.

The algorithm is based on the notion of *mixability*.

Consider a game $\mathfrak{G} = \langle \Omega, \Gamma, \lambda \rangle$. A constant $C > 0$ is *admissible* for a *learning rate* $\eta > 0$ if for every $N = 1, 2, \ldots$, every set of predictions $\gamma(1), \gamma(2), \ldots, \gamma(N)$, and every distribution $p(1), p(2), \ldots, p(N)$ (such that $p(i) \geq 0$ and $\sum_{i=1}^{N} p(i) = 1$) there is $\gamma \in \Gamma$ ensuring for all outcomes $\omega \in \Omega$ the inequality

$$\lambda(\gamma, \omega) \leq -\frac{C}{\eta} \ln \sum_{i=1}^{N} p(i) e^{-\eta \lambda(\gamma(i), \omega)} \ . \tag{1}$$

The *mixability constant* C_η is the infimum of all $C > 0$ admissible for η. This infimum is usually achieved. For example, it is achieved for all $\eta > 0$ whenever Γ is compact and $e^{-\lambda(\gamma, \omega)}$ is continuous[1] in γ.

The Aggregating Algorithm takes a learning rate $\eta > 0$, a constant C admissible for $\mathfrak{G}$ with η, and a prior distribution $p(1), p(2), \ldots, p(N)$ ($p(i) \geq 0$ and $\sum_{i=1}^{N} p(i) = 1$) on experts $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_N$.

We use the notation

$$\mathrm{Loss}_T(\mathcal{S}) = \sum_{t=1}^{T} \lambda(\gamma_t, \omega_t) \ ,$$

$$\mathrm{Loss}_T(\mathcal{E}_i) = \sum_{t=1}^{T} \lambda(\gamma_t(i), \omega_t) \ , \quad i = 1, 2, \ldots, N$$

for the cumulative loss of a learner and experts (as the pack size is always 1, we omit the second index k and write, say, ω_t instead of $\omega_{t,1}$). The following proposition provides an upper bound on the learner's loss.

Proposition 1 (Vovk 1990, 1998) *Let C be admissible for $\eta > 0$. Then for every $N = 1, 2, \ldots$, the loss of a learner $\mathcal{S}$ using the AA with η and a prior distribution $p(1), p(2), \ldots, p(N)$ satisfies*

$$\mathrm{Loss}_T(\mathcal{S}) \leq C \mathrm{Loss}_T(\mathcal{E}_i) + \frac{C}{\eta} \ln \frac{1}{p(i)} \tag{2}$$

for every expert $\mathcal{E}_i$, $i = 1, 2, \ldots, N$, all time horizons $T = 1, 2, \ldots$, and all outputs made by the nature. □

The pseudocode of the Aggregating Algorithm and the proof of the proposition are given in Appendix A.

The choice of a particular learning rate depends on the size of the mixability constants. For some games we have natural optimal choices. For example, consider the general square loss game with $\Omega = \Gamma = [A, B]$ and $\lambda(\gamma, \omega) = (\gamma - \omega)^2$ used for the experiments in this paper. It is one of the so called *mixable* games with C achieving 1. The natural choice of η is then the maximum value such that $C_\eta = 1$; it minimises the second term on the right-hand side of (2). Such η is given by $\eta_0 = 2/(B - A)^2$; one can easily adapt to the interval $[A, B]$ the derivation of Vovk (2001) for the interval $[-Y, Y]$.

[1] Or $\lambda(\gamma, \omega)$ is continuous w.r.t. the extended topology of $[0, +\infty]$.

Remark 1 For the general square loss game, if the learning rate η_0 is used, one can find a value of γ satisfying (1) for all $\omega \in [A, B]$ using an explicit *substitution function*

$$\gamma = \frac{A+B}{2} - \frac{g(B) - g(A)}{2},$$

where

$$g(\omega) = -\frac{1}{\eta_0} \ln \sum_{i=1}^{N} p(i) e^{-\eta_0 \lambda(\gamma(i), \omega)}$$

following Vovk (2001). This makes the Aggregating Algorithm for the general square loss game efficient.

For non-mixable games (such as the absolute loss game with $\lambda(\gamma, \omega) = |\gamma - \omega|$), bound (2) provides a trade-off. Optimising the bound is a more difficult task and may require assumptions on the behaviour of experts or the time horizon T.

The importance of the AA follows from the results of Vovk (1998). Under some mild regularity assumptions on the game and assuming the uniform initial distribution, it can be shown that the constants in inequality (2) are optimal. If any merging strategy achieves the guarantee (with $A > 0$)

$$\mathrm{Loss}_T(\mathcal{S}) \le C\mathrm{Loss}_T(\mathcal{E}_i) + A \ln N$$

for all experts $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_N$, $N = 1, 2, \ldots$, all time horizons T, and all outcomes, then the AA with the uniform prior distribution $p(i) = 1/N$ and some $\eta > 0$ provides the guarantee with the same or lower C and A. We discuss this result in more detail in Appendix A.

2.3 Delayed feedback approach

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

In the delayed feedback prediction with expert advice protocol, on every step the learner gets just one round of predictions from each expert and must produce its own. However, the outcome corresponding to these predictions may become available later. If it is revealed on the same trial as in Sect. 2.2, we say that the delay is one. If it is revealed on the next trial, the delay equals two, etc. Prediction of packs of size not exceeding K can be considered as prediction with delays not exceeding K.

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

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

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

$$\mathrm{Loss}_T(\mathcal{S}) \le C\mathrm{Loss}_T(\mathcal{E}_i) + \frac{CD}{\eta} \ln \frac{1}{p(i)} \tag{3}$$

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

For Protocol 2 we can define plain cumulative loss

$$\mathrm{Loss}_T(\mathcal{S}) = \sum_{t=1}^{T} \sum_{k=1}^{K_t} \lambda(\gamma_{t,k}, \omega_{t,k}) \ , \tag{4}$$

$$\mathrm{Loss}_T(\mathcal{E}_i) = \sum_{t=1}^{T} \sum_{k=1}^{K_t} \lambda(\gamma_{t,k}(i), \omega_{t,k}) \ , \quad i = 1, 2, \ldots, N. \tag{5}$$

Then (3) implies

$$\mathrm{Loss}_T(\mathcal{S}) \le C\mathrm{Loss}_T(\mathcal{E}_i) + \frac{CK}{\eta} \ln \frac{1}{p(i)} \ , \tag{6}$$

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

However, the Parallel Copies algorithm has two disadvantages. First, it requires us to maintain D arrays of experts' weights (see Protocol 7 in Appendix A). Each copy of AA needs to maintain N weights, one for each expert. If packs of size D come up, we will need D such arrays. Secondly, and more importantly, the algorithm depends on the order of predictions in the pack. It matters what copy of the AA will pick a particular round of experts' predictions and the result is not invariant w.r.t. the order within the packs.

Below we will build algorithms that are order-independent and have loss bounds both similar (Sect. 4.1) and essentially different (Sect. 4.2) from (6). Our method is based on a generalisation of the concept of mixability and a direct application of AA to packs. The resulting algorithms will maintain one array of N weights (or losses).

3 Mixability

In this section we extend the concept of mixability defined in Sect. 2.2 to packs of outcomes. This will be a key tool for the analysis of the algorithms we will construct. We need an upper bound on admissible constants in order to get upper loss bounds and lower bounds in order to establish some form of optimality. As we cannot restrict ourselves to packs of constant size, we need to consider suboptimal constants too.

For a game $\mathfrak{G} = \langle \Omega, \Gamma, \lambda \rangle$ and a positive integer K consider the game $\mathfrak{G}^K$ with the outcome and prediction space given by the Cartesian products Ω^K and Γ^K and the loss function $\lambda^{(K)}((\gamma_1, \gamma_2, \ldots, \gamma_K), (\omega_1, \omega_2, \ldots, \omega_K)) = \sum_{k=1}^{K} \lambda(\gamma_k, \omega_k)$. What are the mixability constants for this game? Let C_η be the constants for $\mathfrak{G}$ and $C_\eta^{(K)}$ be the constants for $\mathfrak{G}^K$.

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

Lemma 1 *If $C > 0$ is admissible for a game $\mathfrak{G}$ with a learning rate $\eta > 0$, then C is admissible for the game $\mathfrak{G}^K$ with the learning rate η/K.*

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

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

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

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

We will now apply the generalised Hölder inequality. On measure spaces, the inequality states that $\| \prod_{k=1}^{K} f_k \|_r \leq \prod_{k=1}^{K} \| f_k \|_{r_k}$, where $\sum_{k=1}^{K} 1/r_k = 1/r$. This follows by induction from the version of the inequality given by Loève (1977, Sect. 9.3). Interpreting a vector $x_k = (x_k(1), x_k(2), \ldots, x_k(N)) \in \mathbb{R}^N$ as a function on a discrete space $\{1, 2, \ldots, N\}$ and introducing on this space a measure $p(i)$, $i = 1, 2, \ldots, N$, we obtain

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

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

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

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

Remark 2 Note that the proof of the lemma offers a constructive way of solving (1) for $\mathfrak{G}^K$ provided we know how to solve (1) for $\mathfrak{G}$. Namely, to solve (1) for $\mathfrak{G}^K$ with the learning rate η/K, we solve K systems for $\mathfrak{G}$ with the learning rate η.

We will now show that the admissible constants given by Lemma 1 cannot be decreased for a wide class of games. In order to get a lower bound for $C_\eta^{(K)}$, we need the following concepts.

A *generalised prediction* w.r.t. a game $\mathfrak{G}$ is a function from Ω to $[0, +\infty]$. Every prediction $\gamma \in \Gamma$ specifies a generalised prediction by $\lambda(\gamma, \cdot)$, hence the name.

A *superprediction* is a generalised prediction minorised by the loss of some prediction, i.e., a superprediction is a function $f : \Omega \to [0, +\infty]$ such that for some $\gamma \in \Gamma$ we have $f(\omega) \geq \lambda(\gamma, \omega)$ for all $\omega \in \Omega$. The shape of the set of superpredictions plays a crucial role in determining C_η.

Lemma 2 *Let a game $\mathfrak{G}$ have a convex set of superpredictions. If $C > 0$ is admissible for $\mathfrak{G}^K$ with a learning rate $\eta/K > 0$, then C is admissible for $\mathfrak{G}$ with the learning rate η.*

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

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

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

for all $\omega_1, \omega_2, \ldots, \omega_K \in \Omega$.

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

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

for all $\omega \in \Omega$ (we let $\omega_1 = \omega_2 = \ldots = \omega_K = \omega$).

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

Since C_η and $C_{\eta/K}^{(K)}$ are the infimums of admissible values, Lemmas 1 and 2 can be combined into the following theorem.

Theorem 1 *For a game $\mathfrak{G}$ with a convex set of superprediction, any positive integer K and learning rate $\eta > 0$, we have $C_{\eta/K}^{(K)} = C_\eta$.*

This theorem allows us to merge experts' predictions in an optimal way for the case when all packs are of the same size. In this case, we simply apply Proposition 1 and all the existing theory of the AA to the game $\mathfrak{G}^K$.

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

Lemma 3 *For every game $\mathfrak{G}$, if $C > 0$ is admissible with a learning rate $\eta_1 > 0$, it is also admissible with every $\eta_2 \le \eta_1$. Hence the value of C_η is non-decreasing in η.*

Proof Raising the inequality

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

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

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

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

Corollary 1 *For every game $\mathfrak{G}$ and positive integers $K_1 \le K_2$, we have $C^{(K_1)}_{\eta/K_2} \le C^{(K_1)}_{\eta/K_1}$.*

Proof The proof is by applying Lemma 3 to $\mathfrak{G}^{K_1}$. □

Remark 3 The proofs of the lemma and corollary are again constructive in the following sense. If we know how to solve (1) for $\mathfrak{G}$ with a learning rate η_1 and an admissible C, we can solve (1) for $\eta_2 \le \eta_1$ and the same C.

Suppose we play the game $\mathfrak{G}^{K_1}$ but have to use the learning rate η/K_2, where $K_2 \ge K_1$, with C admissible for $\mathfrak{G}$ with η. To solve (1), we can take K_1 solutions for (1) for $\mathfrak{G}$ with the learning rate η. □

4 Algorithms for prediction of packs

In this section we apply the theory we have developed to obtain prediction algorithms. This can be done in two essentially different ways leading to different types of bounds. In Sect. 4.1 we introduce AAP-max and AAP-incremental, and in Sect. 4.2 we introduce AAP-current.

4.1 Prediction with plain bounds

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

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

Protocol 3 (AAP-max)

initialise losses $L_0(i) = 0$, $i = 1, 2, \dots, N$
this step is skipped
set weights to $w_0(i) = p(i)$, $i = 1, 2, \dots, N$
FOR $t = 1, 2, \dots$
  normalise the weights $p_{t-1}(i) = w_{t-1}(i) / \sum_{i=1}^N w_{t-1}(i)$
  FOR $k = 1, 2, \dots, K_t$
    read the experts' predictions $\gamma_{t,k}(i)$, $i = 1, 2, \dots, N$
    output $\gamma_{t,k} \in \Gamma$ satisfying for all $\omega \in \Omega$ the inequality $\lambda(\gamma_{t,k}, \omega) \le -\frac{C}{\eta} \ln \sum_{i=1}^N p_{t-1}(i) e^{-\eta\lambda(\gamma_{t,k}(i),\omega)}$
  ENDFOR
  observe the outcomes $\omega_{t,k}$, $k = 1, 2, \dots, K_t$
  update the losses $L_t(i) = L_{t-1}(i) + \sum_{k=1}^{K_t} \lambda(\gamma_{t,k}(i), \omega_{t,k})$, $i = 1, 2, \dots, N$
  let $K_t^{\max} = K$
  update the experts' weights $w_t(i) = p(i) e^{-\eta L_t(i)/K_t^{\max}}$, $i = 1, 2, \dots, N$
END FOR

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

Protocol 4 (AAP-incremental)

2 initialise $K_0^{\max} = 1$
12 update $K_t^{\max} = \max(K_{t-1}^{\max}, K_t)$

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

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

and losses do not need to be maintained explicitly.

If C is admissible for $\mathfrak{G}$ with the learning rate η and $p_{t-1}(i)$, $i = 1, 2, \ldots, N$, the step on line 8 can always be performed and the (plain) cumulative losses (4) and (5) satisfy the following inequalities.

Theorem 2 *Let C be admissible for $\mathfrak{G}$ with the learning rate η. Then*

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

$$\mathrm{Loss}_T(\mathcal{S}) \le C\mathrm{Loss}_{\mathcal{E}_i}(\mathcal{S}) + \frac{KC}{\eta} \ln \frac{1}{p(i)}$$

   for all outcomes and experts' predictions as long as the pack size does not exceed K, i.e., $K_t \le K$, $t = 1, 2, \ldots, T$.
2. *The learner following AAP-incremental suffers loss satisfying*

$$\mathrm{Loss}_T(\mathcal{S}) \le C\mathrm{Loss}_{\mathcal{E}_i}(\mathcal{S}) + \frac{KC}{\eta} \ln \frac{1}{p(i)} \ ,$$

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

Proof The proof essentially repeats that of Proposition 1. By induction one can show that

$$e^{-\eta \mathrm{Loss}_t(\mathcal{S})/(CK_{\max}^t)} \ge \sum_{i=1}^{N} p(i) e^{-\eta \mathrm{Loss}_t(\mathcal{S})/K_{\max}^t} \ . \tag{7}$$

Indeed, Lemma 1 with Remark 2, Corollary 1 with Remark 3, and the step on line 8 of Protocols 3 and 4 ensure that

$$e^{-\frac{\eta}{K_{\max}^{t+1}} \sum_{k=1}^{K_{t+1}} \lambda(\gamma_{t+1,k}, \omega_{t+1,k})/C} \ge \sum_{i=1}^{N} p_t(i) e^{-\frac{\eta}{K_{\max}^{t+1}} \sum_{k=1}^{K_{t+1}} \lambda(\gamma_{t+1,k}(i), \omega_{t+1,k})} \ . \tag{8}$$

If $K_{\max}^{t+1} = K_{\max}^t$, then we simply multiply inequality (7) by inequality (8) and substitute the expression for $p_t(i)$ to get the analogue of inequality (7) for time $t+1$. If $K_{\max}^{t+1} > K_{\max}^t$, we first raise inequality (7) to the power $K_{\max}^t / K_{\max}^{t+1} < 1$ and apply Jensen's inequality.

To complete the proof, it remains to drop all terms from the sum in inequality 7 except for one. □

4.2 Prediction with bounds on pack averages

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

Protocol 5 (AAP-current)

1 initialise weights $w_0(i) = p(i)$, $i = 1, 2, \dots, N$
2 FOR $t = 1, 2, \dots$
3 normalise the weights $p_{t-1}(i) = w_{t-1}(i) / \sum_{i=1}^{N} w_{t-1}(i)$
4 FOR $k = 1, 2, \dots, K_t$
5 read the experts' predictions $\gamma_{t,k}(i)$, $i = 1, 2, \dots, N$
6 output $\gamma_{t,k} \in \Gamma$ satisfying for all $\omega \in \Omega$ the inequality $\lambda(\gamma_{t,k}, \omega) \leq -\frac{C}{\eta} \ln \sum_{i=1}^{N} p_{t-1}(i) e^{-\eta \lambda(\gamma_{t,k}(i), \omega)}$
7 ENDFOR
8 observe the outcomes $\omega_{t,k}$, $k = 1, 2, \dots, K_t$
9 update the experts' weights $w_t(i) = w_{t-1}(i) e^{-\eta \sum_{k=1}^{K} \lambda(\gamma_{t,k}(i), \omega_{t,k}) / K_t}$, $i = 1, 2, \dots, N$
10 END FOR

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

Defining *cumulative average loss* of a strategy $\mathcal{S}$ and experts $\mathcal{E}_i$ working in the environment specified by Protocol 1 as

$$\mathrm{Loss}_T^{\mathrm{average}}(\mathcal{S}) = \sum_{t=1}^{T} \frac{\sum_{k=1}^{K_t} \lambda(\gamma_{t,k}, \omega_{t,k})}{K_t} \ ,$$

$$\mathrm{Loss}_T^{\mathrm{average}}(\mathcal{E}_i) = \sum_{t=1}^{T} \frac{\sum_{k=1}^{K_t} \lambda(\gamma_{t,k}(i), \omega_{t,k})}{K_t} \ , \quad i = 1, 2, \dots, N,$$

we get the following theorem.

Theorem 3 *If C is admissible for $\mathfrak{G}$ with the learning rate η, then the learner following AAP-current suffers loss satisfying*

$$\mathrm{Loss}_T^{\mathrm{average}}(\mathcal{S}) \leq C \mathrm{Loss}_{\mathcal{E}_i}^{\mathrm{average}}(\mathcal{S}) + \frac{C}{\eta} \ln \frac{1}{p(i)} \tag{9}$$

for all outcomes and experts' predictions. □

Proof We again prove by induction that

$$e^{-\eta \mathrm{Loss}_t^{\mathrm{average}}(\mathcal{S})/C} \geq \sum_{i=1}^{N} p(i) e^{-\eta \mathrm{Loss}_t^{\mathrm{average}}(\mathcal{S})} \ . \tag{10}$$

The step on line 6 of Protocol 5 ensures, as in the proof of Theorem 2, that

$$e^{-\frac{\eta}{K_{t+1}} \sum_{k=1}^{K_{t+1}} \lambda(\gamma_{t+1,k}, \omega_{t+1,k})/C} \geq \sum_{i=1}^{N} p_t(i) e^{-\frac{\eta}{K_{t+1}} \sum_{k=1}^{K_{t+1}} \lambda(\gamma_{t+1,k}(i), \omega_{t+1,k})} \ . \tag{11}$$

The induction step is by multiplying inequalities (10) and (11). □

5 Discussion and optimality

The loss bounds from Theorem 2 do not improve on inequality (6), which holds for Parallel Copies (see Sect. 2.3 for details). However, the performance of AAP-max and AAP-incremental does not depend on the order of outcomes in packs. In Sect. 6.2.1 we describe numerical experiments comparing AAP-max, AAP-incremental, and AAP-current against the loss of Parallel Copies averaged over permutations within packs.

If all K_t are equal, $K_1 = K_2 = \cdots = K_T = K$, the algorithms AAP-max and AAP-incremental are identical and equivalent to applying the Aggregating Algorithm with the learning rate η/K to the game $\mathfrak{G}^K$. Under the conditions of Theorem 1, the optimality property of the Aggregating Algorithm proven by Vovk (1998) apply. Thus the constants in the bounds of Theorem 2 cannot be improved without the loss of generality. However, if the pack size varies, AAP-max clearly uses a suboptimal learning rate η/K where η/K_t is needed. AAP-incremental does that if the pack size decreases. We compare AAP-incremental and AAP-max experimentally in Sect. 6.2.2.

The bound of Theorem 3 is, to our knowledge, novel and cannot be straightforwardly obtained using a parallel copies-type merging strategy. If the pack size is the same, the bound is optimal (and identical to those from Theorem 2). If the pack size varies, AAP-current always uses the optimal learning rate. However, technically it is not covered by the optimality results of Vovk (1998) as the game changes from step to step. We leave this as an open problem.

The bound of Theorem 3 involves cumulative average loss and do not imply good bounds for plain cumulative loss straightforwardly. If $K_{\min} \le K_1, K_2, \ldots, K_T \le K_{\max}$, then $\mathrm{Loss}_T^{\text{average}}(\mathcal{S}) \ge \mathrm{Loss}_T(\mathcal{S})/K_{\max}$ and $\mathrm{Loss}_{\mathcal{E}_i}^{\text{average}}(\mathcal{S}) \le \mathrm{Loss}_{\mathcal{E}_i}(\mathcal{S})/K_{\min}$. We get the bound

$$\mathrm{Loss}_T(\mathcal{S}) \le \frac{K_{\max}}{K_{\min}} C\mathrm{Loss}_{\mathcal{E}_i}(\mathcal{S}) + \frac{CK_{\max}}{\eta} \ln \frac{1}{p(i)} \tag{12}$$

for the cumulative loss of AAP-current, which appears inferior to those from Theorem 2. However, in experiments AAP-current shows good performance even in terms of the plain cumulative loss; see Sect. 6.2.3. Bound (12), loose it may be, provides an explanation to some phenomena we observe in Sect. 6.2.3.

5.1 A mix loss lower bound

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

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

Protocol 6 (Mix loss)

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

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

Lemma 4 *For any K arrays of N probabilities $p_k(1), p_k(2), \ldots, p_k(N)$, $k = 1, 2, \ldots, K$ (where $p_k(i) \in [0, 1]$ for all $i = 1, 2, \ldots, N$ and $k = 1, 2, \ldots, K$, and $\sum_{i=1}^{N} p_k(i) = 1$ for all k), there is i_0 such that*

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

Proof Assume the converse. Let $\prod_{k=1}^{K} p_k(i) > 1/N^K$ for all i. By the inequality of arithmetic and geometric means

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

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

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

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

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

The contradiction proves the lemma. □

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

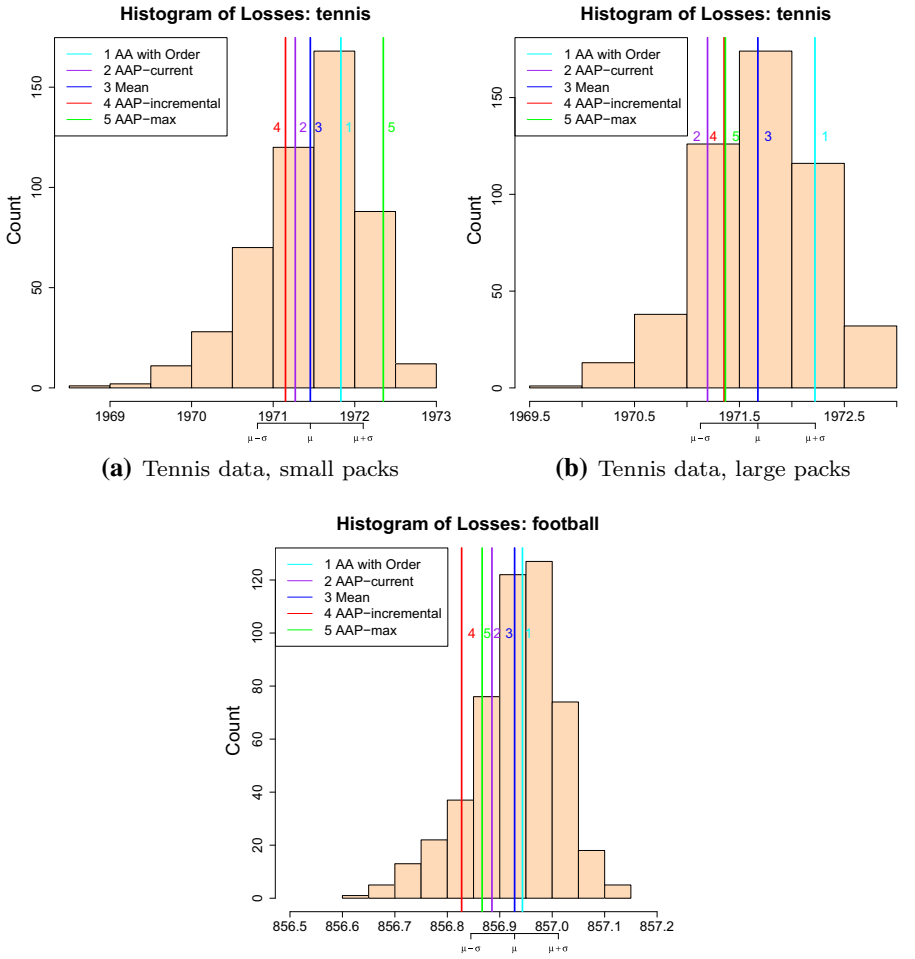
$$\ell_t = -\sum_{k=1}^{K_t} \ln p_{t,k} = -\ln \prod_{k=1}^{K_t} p_{t,k}(i_0) \geq -\ln \frac{1}{N^{K_t}} = K_t \ln N$$

while $\ell_t(i_0) = 0$. We see that over a single pack of size K we can achieve the regret of $K \ln N$. Thus every upper bound of the form $L_T \leq L_T(i) + R$ should have $R \geq K_1 \ln N$, where K_1 is the size of the first pack.

6 Experiments

In this section, we present some empirical results.[2] We want to compare the behaviour of the AAP family algorithms against each other and against the Parallel Copies algorithm of Sect. 2.3. Appendices B and C investigate related questions concerned with the power of on-line learning.

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



(a) Tennis data, small packs

(b) Tennis data, large packs

(c) Football data

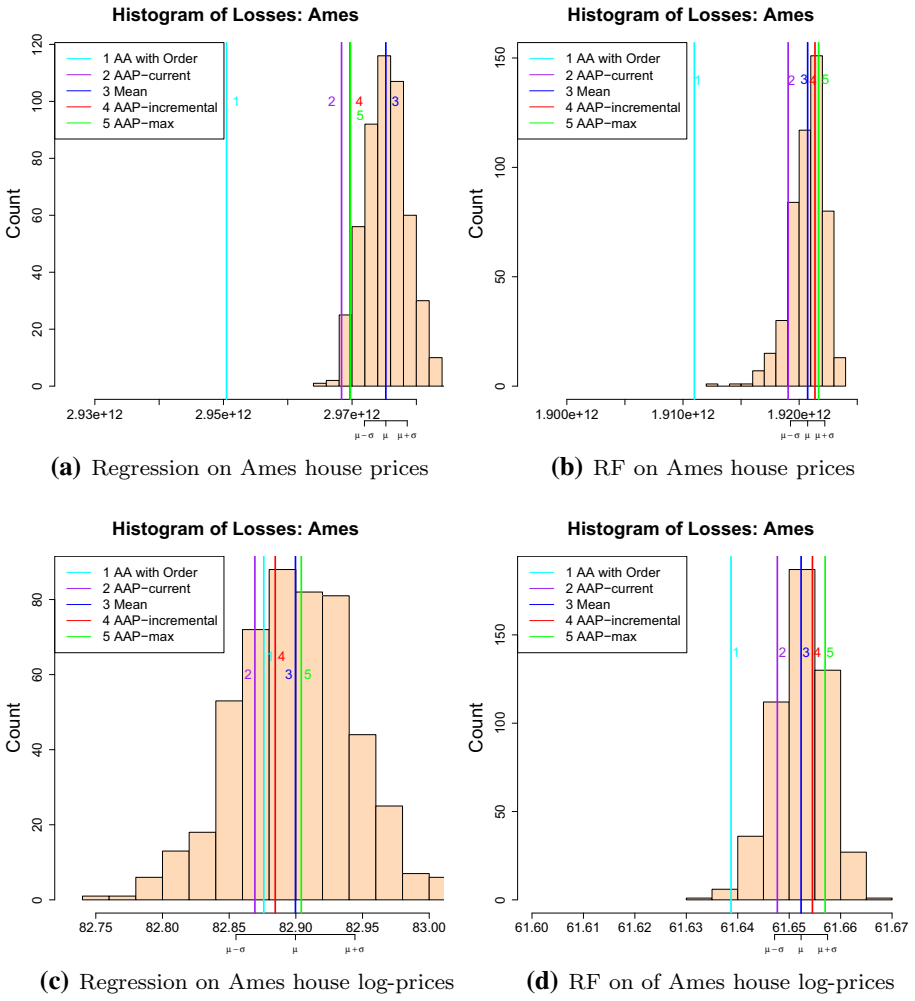
Fig. 1 Histogram of total losses of Parallel Copies with total losses of AAP algorithms on sports datasets

6.1 Datasets and experts

For our experiments, we used two sports datasets and two datasets of house prices.

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

There is a tradition of using house prices as a benchmark for machine learning algorithms going back to the Boston housing dataset. However, batch learning protocols have hitherto been used in most studies. Recently extensive datasets with timestamps have become available. They call for on-line learning protocols. Property prices are prone to strong movements



(a) Regression on Ames house prices

(b) RF on Ames house prices

(c) Regression on Ames house log-prices

(d) RF on of Ames house log-prices

Fig. 2 Histogram of total losses of Parallel Copies with total losses of AAP algorithms on house price datasets

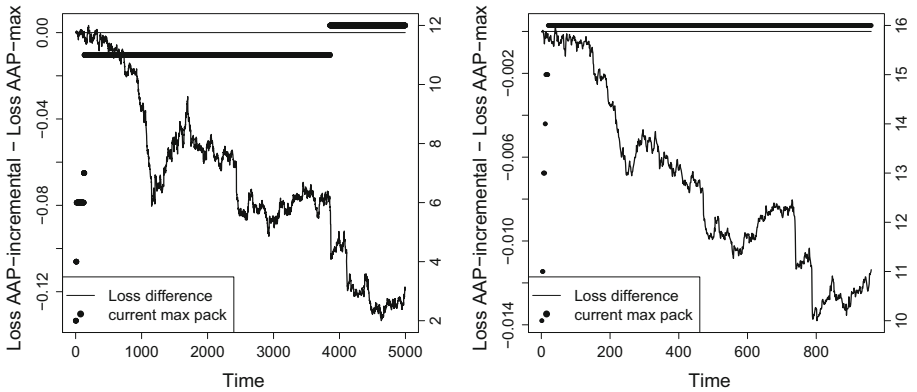
over time and the pattern of change may be complicated. On-line algorithms should capture these patterns.

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

6.1.1 Sports datasets

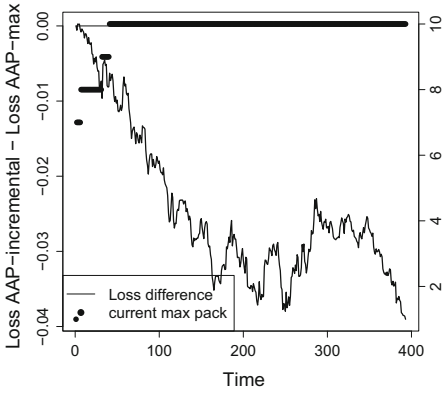
In order to establish continuity with the existing empirical work, we use the tennis dataset[3] studied by Vovk and Zhdanov (2009). It contains historical information about tennis tournaments from 2004, 2005, 2006, and 2007, including Australian Open, French Open, US Open, and Wimbledon. The outcomes in the dataset are results of tennis matches coded as 0

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



(a) Tennis data, small packs

(b) Tennis data, large packs



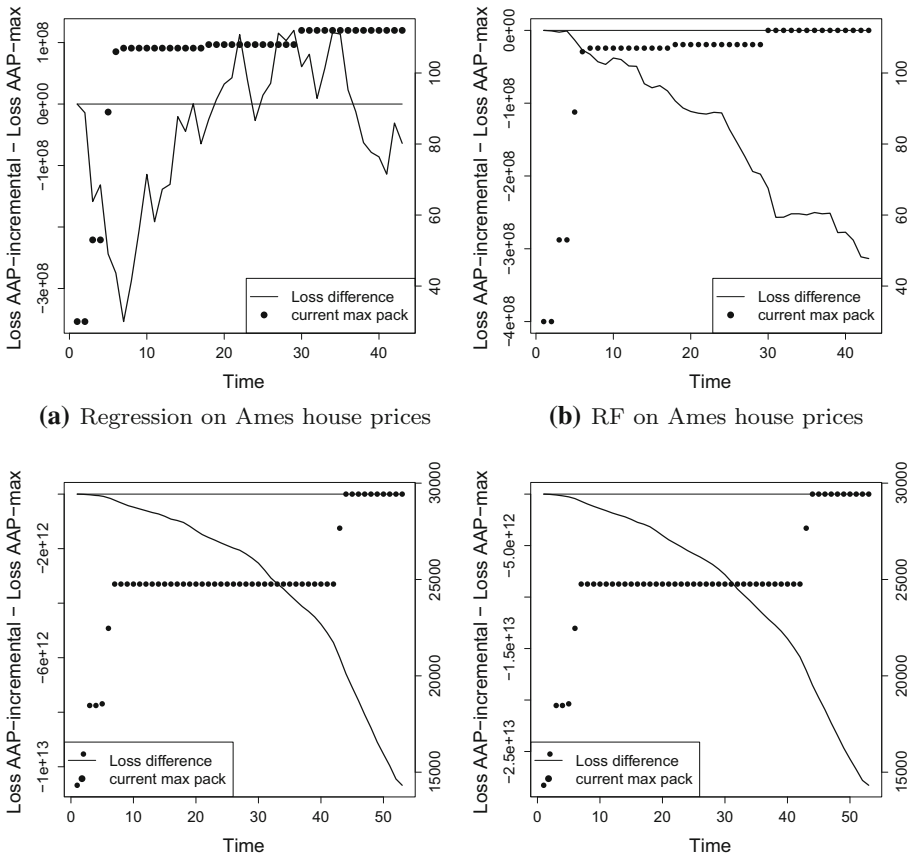
(c) Football data

Fig. 3 Difference of cumulative losses of AAP-incremental and AAP-max versus time on sports data with cumulative maximum pack sizes superimposed

or 1 according to which side wins (there can be no draws). The total number of outcomes is 10,087. A prediction is $\gamma \in [0, 1]$, which can be understood as the probability of the outcome 1. We use the quadratic loss function $\lambda(\gamma, \omega) = (\gamma - \omega)^2$. This falls under the definition of the general square loss game described in Sect. 2.2 (see Remark 1 for the discussion of the substitution function). Note that the loss function used in this paper equals one half of the one used by Vovk and Zhdanov (2009); we make this choice for consistency with regression experiments.

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

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



(a) Regression on Ames house prices

(b) RF on Ames house prices

(c) Regression on London house prices

(d) RF on London house prices

Fig. 4 Difference of cumulative losses of AAP-incremental and AAP-max versus time on house price datasets with cumulative maximum pack sizes superimposed

The second sports dataset was compiled by us from historical information[4] on football matches and bookmakers' odds. The dataset covers four seasons, 2013/2014, 2014/2015, 2015/2016, and 2016/2017 of the English Premier League and totals 1520 matches. Each match can have three outcomes, 'home win', 'draw', or 'away win', interpreted as three unit vectors (1, 0, 0), (0, 1, 0), and (0, 0, 1). A prediction is a vector $\gamma = (p_1, p_2, p_3)$ from the simplex, i.e., $p_i \geq 0$ and $p_1 + p_2 + p_3 = 1$ and the loss is the quadratic norm of the difference, $\lambda(\gamma, \omega) = \|\gamma - \omega\|_2^2$. This is a case of the multidimensional Brier game (Vovk and Zhdanov 2009). The game is mixable and the maximum learning rate such that $C_\eta = 1$ is $\eta_0 = 1$; the substitution function is provided by Vovk and Zhdanov (2009, Proposition 2).

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

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

[4] Available at http://Football-Data.co.uk.

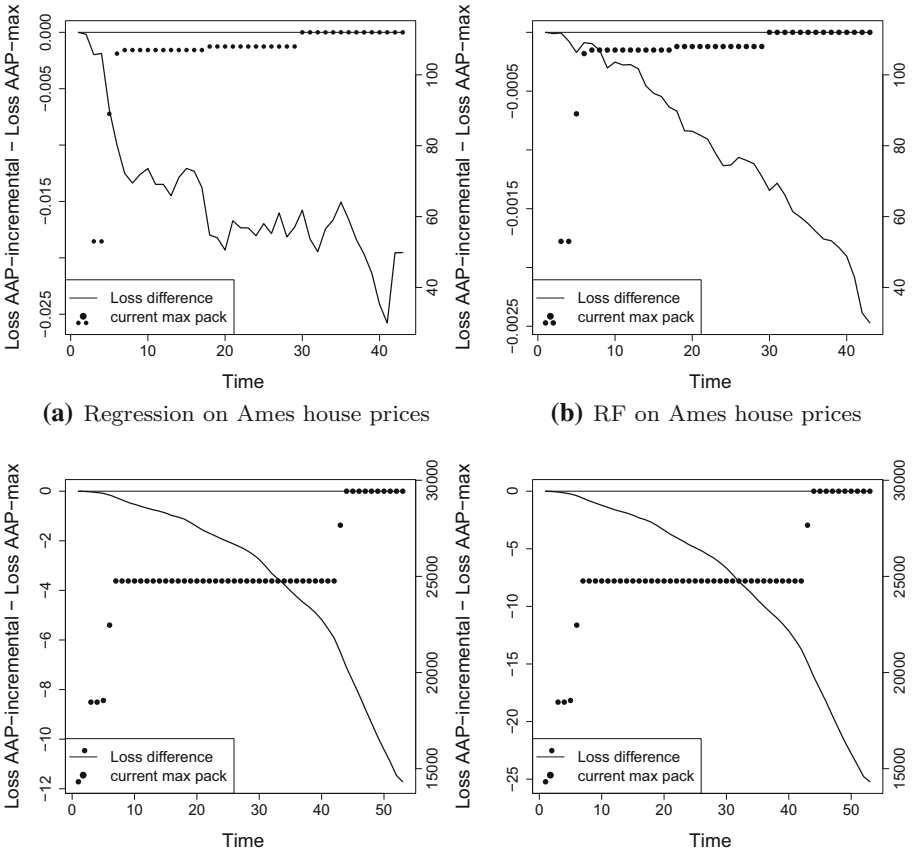


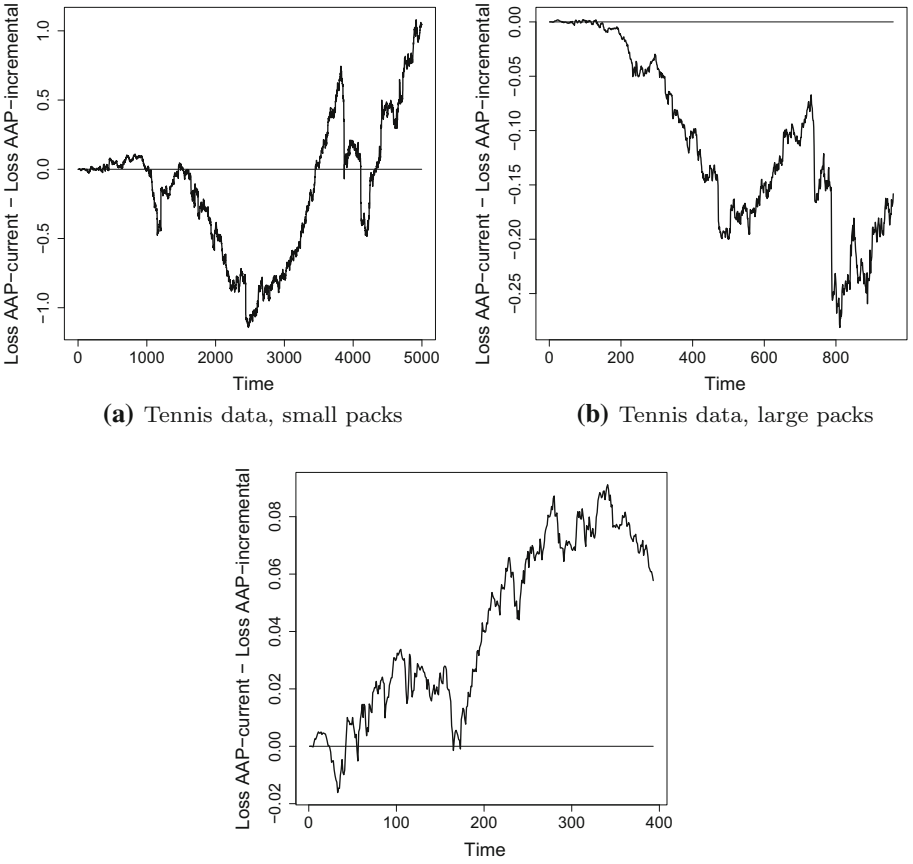
Fig. 5 Difference of cumulative losses of AAP-incremental and AAP-max versus time on logarithms of house prices with cumulative maximum pack sizes superimposed

6.1.2 Ames house prices

The Ames dataset describes the property sales that occurred in Ames, Iowa between 2006 and 2010. The dataset contains records of 2930 house sales transactions with 80 attributes, which are a mixture of nominal, ordinal, continuous, and discrete parameters (including physical property measurements) affecting the property value. The dataset was compiled by De Cock (2011) for use in statistics education as a modern substitute for the Boston Housing dataset. For the outcome we take the raw sales prices or their logarithms; these make two sets of experiment. We try to predict the outcomes measuring the deviation by the squared difference. This again falls under the definition of the general square loss game of Sect. 2.2. The bounds A and B are taken from the first year of data, which is used for training.

There are timestamps in the dataset, but they contain only the month and the year of the purchase. The date is not available. We treat one month of transactions as a pack and interpret the problem as an on-line one falling under Protocol 1.

We create two pools of experts for experiments. In the first pool, our experts are linear regression models based on only two attributes: the neighbourhood and the total square



(a) Tennis data, small packs

(b) Tennis data, large packs

(c) Football data

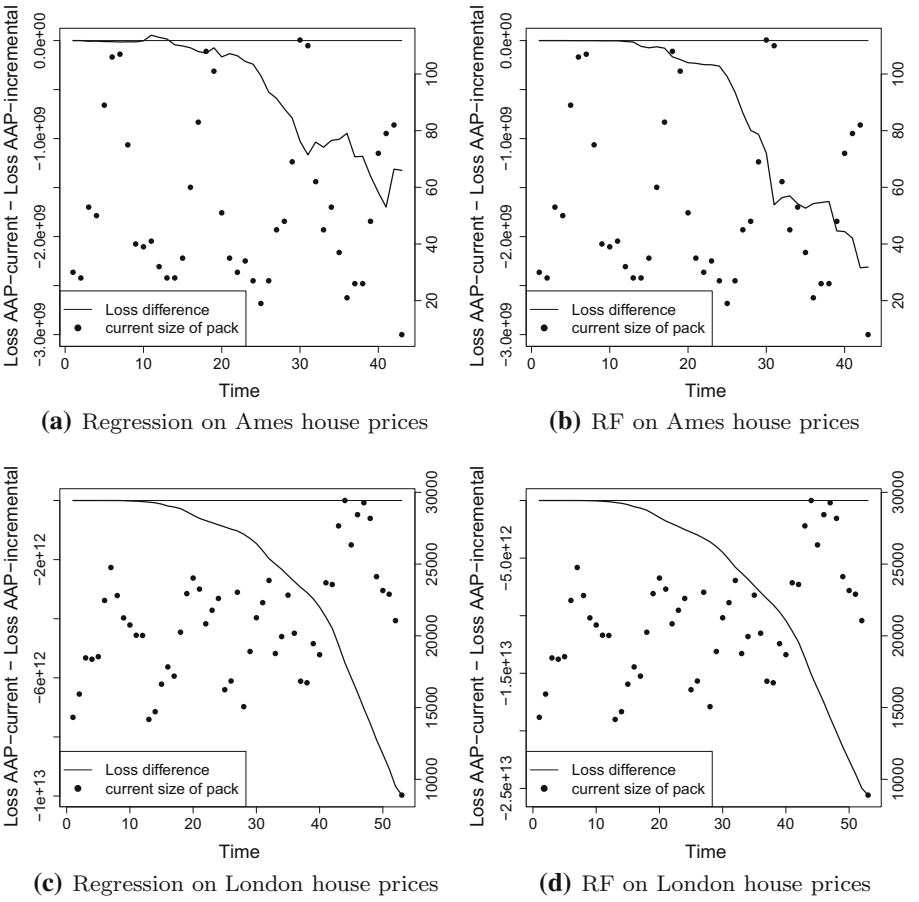
Fig. 6 Difference of cumulative losses of AAP-current and AAP-incremental versus time on sports data

footage of the dwelling. These simple models explain around 80% of the variation in sales prices and they are very easy to train. Each expert has been trained on one month from the first year of the data. Hence there are 12 'monthly' experts. For the second pool we use random forests (RF) models after Bellotti http://wwwf.imperial.ac.uk/~abellott/publications.htm. A model was built for each quarter of the first year. Hence there are four 'quarterly' experts. They take longer to train but produce better results. Note that 'monthly' RF experts were not practical; training a tree requires a lot of data and 'monthly' experts returned very poor results. We apply the experts to predict the prices starting from year two.

6.1.3 London house prices

Another dataset we used contains house prices in and around London over the period 2009 to 2014. This dataset was made publicly available by the Land Registry[5] in the UK and was originally sourced as part of a Kaggle competition. The Property Price data consists of details

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



(a) Regression on Ames house prices

(b) RF on Ames house prices

(c) Regression on London house prices

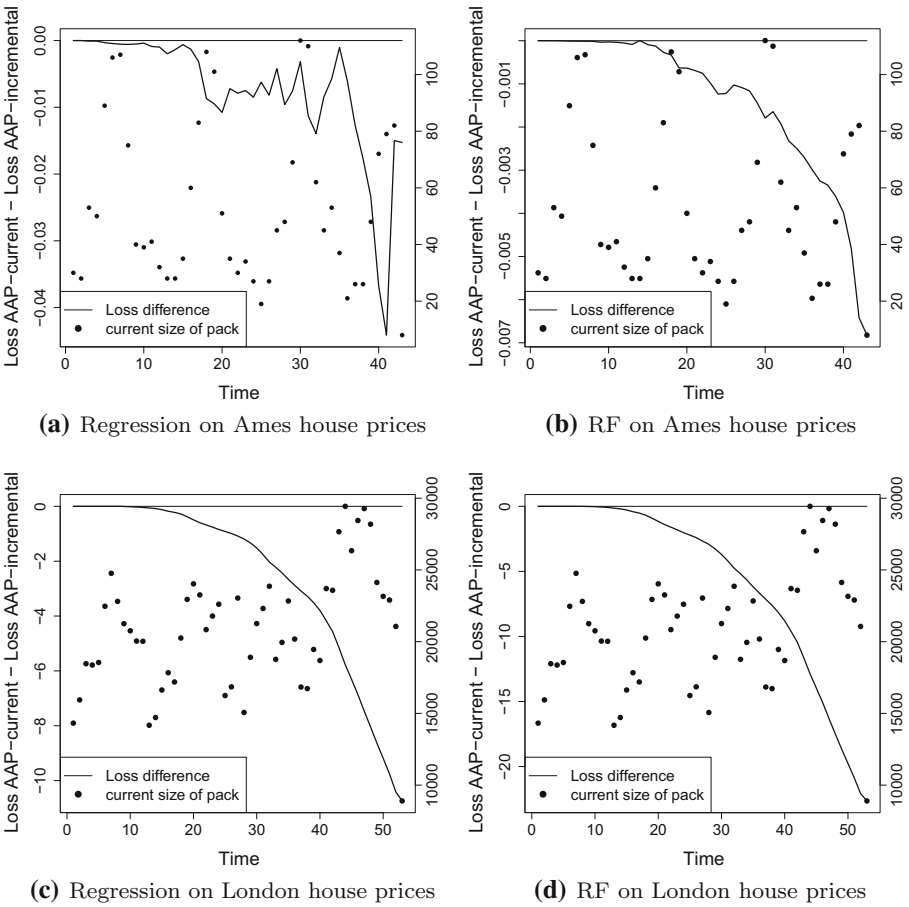
(d) RF on London house prices

Fig. 7 Difference of cumulative losses of AAP-current and AAP-incremental versus time on house price data with current pack sizes superimposed

for property sales and contains around 1.38 million observations. This dataset was studied by Bellotti (2017) to provide reliable region predictions for Automated Valuation Models of house prices. Again, we try to predict sales prices and their logarithms.

As with the Ames dataset, we use linear regression models that were built for each month of the first year of the data as experts of AAP. The features that were used in regression models contain information about the property: property type, whether new build, whether free- or leasehold. Along with the information about the proximity to tube and railway stations, our models use the English indices of deprivation from 2010,[6] which measures relative levels of deprivation. The following deprivation scores were used in models: income, employment, health and disability, education for children and skills for adults, barriers to housing and services with sub-domains wider barriers and geographical barriers, crime, living environment score with sub-domains for indoor and outdoor living (i.e., quality of housing and external environment, respectively). In addition to the general income score, separate scores for income deprivation affecting children and the older population were used.

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



(a) Regression on Ames house prices **(b)** RF on Ames house prices

(c) Regression on London house prices **(d)** RF on London house prices

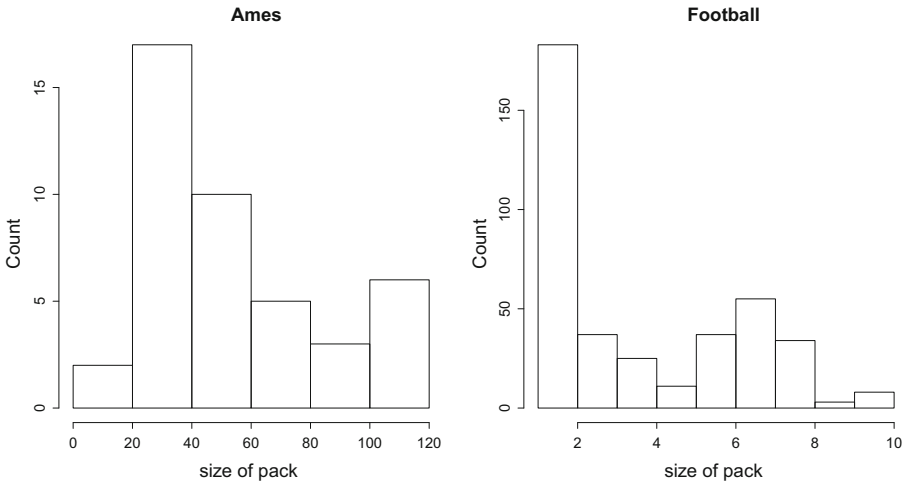
Fig. 8 Difference of cumulative losses of AAP-current and AAP-incremental versus time on logarithms of house price datasets with current pack sizes superimposed

In the second set of experiments on London house price dataset, we use RF models built for each month of the first year as experts. Unlike the Ames dataset, London dataset contains enough observations to train RF models on one month of data. Hence we get 12 'monthly' experts.

6.2 Comparison of merging algorithms

6.2.1 Comparison of AAP with parallel copies of AA

We start by comparing the family of AAP merging algorithms against parallel copies of AA. While for AAP algorithms the order of examples in the pack makes no difference, for Parallel Copies it is important. To analyse the dependency on the order we ran Parallel Copies 500 times randomly shuffling each pack each time. The experiments were only carried out on sports and Ames data, as on London data they would take too long to complete. Figures 1 and 2



(a) Ames pack sizes **(b)** Football pack sizes

Fig. 9 Histograms of pack sizes

show histograms of total losses of Parallel Copies, total losses of AAP family algorithms, and the total loss of Parallel Copies with one particular order, as in the database.

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

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

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

6.2.2 Comparison of AAP-incremental and AAP-max

As one can see from Figs. 1 and 2, AAP-max is usually the worst among the AAP bunch. In this section we check this by comparing AAP-max against AAP-incremental. Here AAP-max receives the maximum pack size calculated retrospectively from the start and AAP-incremental uses the current maximum.

For a detailed comparison of two on-line learning algorithms, $\mathcal{S}_1$ and $\mathcal{S}_2$, it is not enough to consider the two values of their total losses $\text{Loss}_{\mathcal{S}_1}(T)$ and $\text{Loss}_{\mathcal{S}_2}(T)$. We need to see how these losses were accumulated. So, following Vovk and Zhdanov (2009) and Kalnishkan et al. (2015), we plot the difference of their cumulative losses, $\text{Loss}_{\mathcal{S}_1}(t) - \text{Loss}_{\mathcal{S}_2}(t)$ versus time t. If the difference steadily decreases, then $\mathcal{S}_1$ consistently outperforms $\mathcal{S}_2$.

Figures 3, 4, and 5 plot the differences in total losses of AAP-incremental and AAP-max on sports datasets, house prices, and logarithms of house prices, respectively. Over the graphs of the difference of losses, the values of $K^t_{\max} = \max_{s=1,2,\dots,t}$, the current maximum pack size, are superimposed.

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

These observations are consistent with the discussion in Sect. 5: AAP-max uses a suboptimal learning rate before the maximum pack size is achieved.

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

6.2.3 Comparison of AAP-current and AAP-incremental

The comparison of AAP-current and AAP-incremental provides a more challenging problem: sometimes one performs better and sometimes the other. Recall that we assess AAP-current by the plain cumulative loss (4) for comparison purposes.

Figures 6, 7, and 8 show the difference in plain cumulative losses of AAP-current and AAP-incremental for sports dataset, house prices and logarithms of house prices, respectively.

We see that AAP-current outperforms AAP-incremental on house prices and tennis data with large packs. The performance of AAP-current is remarkable because by design it is not optimised to minimise the total loss; see the discussion in Sect. 5. In a way, here we assess AAP-current with a measure it is not good at. Still optimal decisions of AAP-current produce superior performance.

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

For the football and housing datasets we do not control the ratio of the maximum and minimum pack sizes. For the football dataset, where AAP-current performs poorly, the ratio is $10/1 = 10$ and for the London house prices, where it performs well, the ratio is much less and equals $29{,}431/8900 = 3.3$.

The Ames dataset apparently does not fit the pattern with a large ratio of $112/8 = 14$. However, one can see from the histogram shown on Fig. 9 that packs of small size are relatively rare; if we ignore them, the ratio immediately goes down. The same argument does not apply to the football dataset with plenty of small packs.

6.3 Conclusions

This section summarises the conclusions from empirical experiments.

We have found that the average performance of Parallel Copies of AA is close to the performance of the AAP family. Some members of the family (especially AAP-incremental

and AAP-current) often perform better than the average. However, Parallel Copies may be able to benefit from extra information contained in the order.

We have also found that AAP-incremental typically outperforms AAP-max, especially before the pack size has reached the maximum. Therefore, we do not need to know the maximum size of the pack in advance.

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

Appendix A: Aggregating Algorithm for making individual predictions

In this appendix we formulate the Aggregating Algorithm for making individual predictions following Vovk (1990, 1998), prove Proposition 1, and discuss optimality of its bound.

Consider Protocol 2, where $K_t = 1$, $t = 1, 2, \ldots$; we drop the second lower index k for brevity.

As discussed in Sect. 2.2, the AA takes as parameters a learning rate $\eta > 0$, a constant C admissible for $\mathfrak{G}$ with η, and a prior distribution $p(1), p(2), \ldots, p(N)$ ($p(i) \geq 0$ and $\sum_{i=1}^{N} p(i) = 1$) on experts $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_N$. The algorithm works according to the following protocol.

Protocol 7 (AA)

1 initialise weights $w_0(i) = p(i)$, $i = 1, 2, \ldots, N$
2 FOR $t = 1, 2, \ldots$
3 read the experts' predictions $\gamma_t(i)$, $i = 1, 2, \ldots, N$
4 normalise the weights $p_{t-1}(i) = w_{t-1}(i) / \sum_{i=1}^{N} w_{t-1}(i)$
5 output $\gamma_t \in \Gamma$ satisfying for all $\omega \in \Omega$ the inequality $\lambda(\gamma_t, \omega) \leq -\frac{C}{\eta} \ln \sum_{i=1}^{N} p_{t-1}(i) e^{-\eta\lambda(\gamma_t(i),\omega)}$
6 observe the outcome ω_t
7 update the experts' weights $w_t(i) = w_{t-1}(i) e^{-\eta\lambda(\gamma_t(i),\omega_t)}$, $i = 1, 2, \ldots, N$
8 END FOR

Since C is admissible, a suitable γ_t can always be found on line 5. For a particular game $\mathfrak{G}$, a simple method can usually be used, such as the substitution function from Remark 1.

We can now sketch the proof of Proposition 1.

Proof Inequality (1) can be rewritten as

$$e^{-\eta\lambda(\gamma,\omega)/C} \geq \sum_{i=1}^{N} p(i) e^{-\eta\lambda(\gamma(i),\omega)} \quad .$$

One can check by induction that the equality

$$e^{-\eta \mathrm{Loss}_t(\mathcal{S})/C} \geq \sum_{i=1}^{N} p(i) e^{-\eta \mathrm{Loss}_t(\mathcal{E}_i)}$$

holds for all $t = 1, 2, \ldots$. Dropping all terms but one on the right-hand side we get the desired inequality. □

Let us discuss the optimality of AA following Vovk (1998). Consider the set $\mathcal{L} \subseteq [0, +\infty)^2$ of points (c, a) such that there is a strategy for the learner $\mathcal{S}$ making sure for any finite set of experts $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_N$ that

$$\mathrm{Loss}_t(\mathcal{S}) \le c\mathrm{Loss}_t(\mathcal{E}_i) + a \ln N \tag{13}$$

for all $t = 1, 2, \ldots$ and $i = 1, 2, \ldots, N$. If $(c_1, a_1) \in \mathcal{L}$ and c_2 and a_2 are such that $c_2 \ge c_1$ and $a_2 \ge a_1$, then clearly $(c_2, a_2) \in \mathcal{L}$.

Let $\mathcal{C} = \{(c(\eta), a(\eta)) \mid \eta \in [0, +\infty]\} \subseteq [0, +\infty]^2$, where

$$c(\eta) = C_\eta \ ,$$
$$a(\eta) = \frac{C_\eta}{\eta}$$

for $\eta \in (0, +\infty)$ and $c(0)$, $a(0)$, $c(+\infty)$, and $a(+\infty)$ are the limits of $c(\eta)$ and $a(\eta)$ as $\eta \to 0$ or $\eta \to +\infty$, respectively (the limits always exist). Vovk (1998, Theorem 1,) shows that under the following assumptions

1. The prediction space Γ is a compact topological space.
2. The function $\lambda(\cdot, \omega)$ is continuous in the first argument for every $\omega \in \Omega$.
4. There is $\gamma_0 \in \Gamma$ such that $\lambda(\gamma_0, \omega) < +\infty$ for every $\omega \in \Omega$.
5. There is no $\gamma \in \Gamma$ such that $\lambda(\gamma, \omega) = 0$ for every $\omega \in \Omega$.

the boundary of $\mathcal{L}$ relative to $[0, +\infty)^2$ coincides with $\mathcal{C} \cap \mathbb{R}^2$, the finite part of $\mathcal{C}$.

Under the assumptions either the set $\mathcal{C}$ collapses to one point $(+\infty, +\infty)$ or the part corresponding to $\eta \in (0, +\infty)$ is a continuous curve (Vovk 1998, Lemma 10) going 'northwest to southeast', i.e., $c(\eta)$ is non-decreasing and $a(\eta)$ is non-increasing (Vovk 1998, Lemma 9).

Since $c(\eta) \ge 1$ (Vovk 1998, Lemma 8), we get $a(\eta) \ge 1/\eta$ and $a(0) = +\infty$. Thus the curve $\mathcal{C}$ starts 'at the top' of the quadrant $[0, +\infty)^2$. Since $a(\eta)/c(\eta) = 1/\eta \to 0$ as $\eta \to +\infty$, we get either $a(+\infty) = 0$ or $c(+\infty) = +\infty$, i.e., the curve $\mathcal{C}$ finishes 'at the bottom' or 'on the right' of $[0, +\infty)^2$ or both (Vovk 1998, Lemma 11).

It is easy to see that every point from $\mathcal{L}$ is lower bounded by some point from $\mathcal{C}$, i.e., for every $(c, a) \in \mathcal{L}$ there is $\eta \in [0, +\infty]$ such that $c(\eta) \le c$ and $a(\eta) \le a$. If $a > 0$, then we can choose $\eta \in (0, +\infty)$ and the Aggregating Algorithm with the learning rate η, admissible C_η, and uniform prior distribution achieves loss satisfying

$$\mathrm{Loss}_t(\mathcal{S}) \le C_\eta \mathrm{Loss}_t(\mathcal{E}_i) + \frac{C_\eta}{\eta} \ln N \ ,$$

where $C_\eta \le c$ and $C_\eta/\eta \le a$. Thus if any algorithm assures bound (13) with $a > 0$, then the Aggregating Algorithm can do the same or better.

Remark 4 It is possible to define a special 'limit case' of the AA for $\eta = +\infty$ to drop the $a > 0$ clause (Vovk 1998, Sect. 4). It is easy to see that for any distribution $p(1), p(2), \ldots, p(N)$ we have

$$-\frac{1}{\eta} \sum_{i=1}^{N} p(i) e^{-\eta \ell_i} \to \min_{i=1,\ldots,N \,:\, p(i)>0} \ell_i$$

as $\eta \to +\infty$. We can thus call C admissible for $\eta = +\infty$ if for every positive integer N and every set of predictions $\gamma(1), \gamma(2), \ldots, \gamma(N) \in \Gamma$ there is $\gamma \in \Gamma$ such that for all outcomes $\omega \in \Omega$ the inequality

$$\lambda(\gamma, \omega) \leq C \min_{i=1,\ldots,N} \lambda(\gamma(i), \omega) \tag{14}$$

holds. If $c(+\infty) < +\infty$, then $C = c(+\infty)$ is admissible for $\eta = +\infty$ by continuity and we can formulate the Aggregating Algorithm as follows. It is independent of prior probabilities $p(i)$, maintains no weights $w_t(i)$, and on line 5 outputs a prediction γ_t such that $\lambda(\gamma_t, \omega) \leq C\lambda(\gamma_t(i), \omega)$ for all $i = 1, 2, \ldots, N$ and $\omega \in \Omega$. The learner $\mathcal{S}$ using this algorithm achieves the bound

$$\mathrm{Loss}_t(\mathcal{S}) \leq C\mathrm{Loss}_t(\mathcal{E}_i) \ . \tag{15}$$

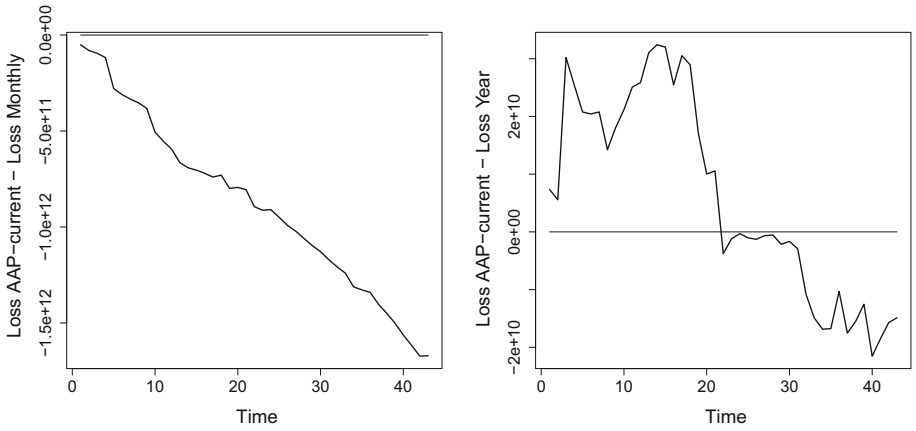
The special case appears to be of little practical use.

Appendix B: Comparison of AAP with batch models

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

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

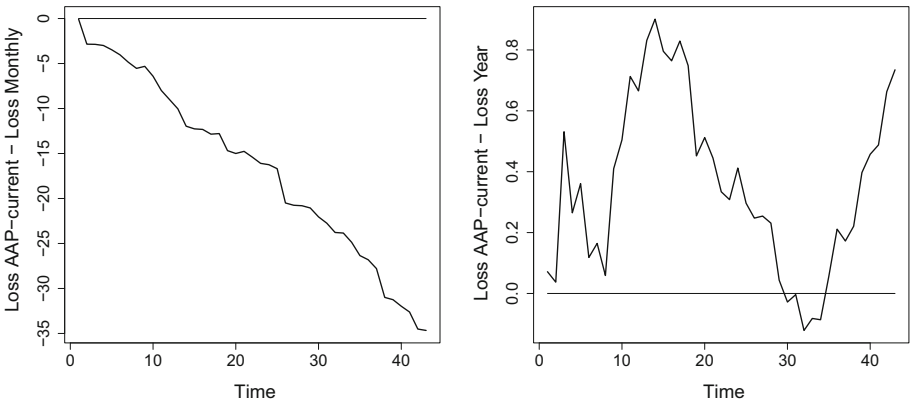
The first batch model we compare our on-line algorithm to is the seasonal model that predicts January with the linear regression model trained on January of the first year, February with the linear model trained on February of the first year, etc.



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

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

Fig. 10 Difference of cumulative losses of AAP and batch models versus time on house price data



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

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

Fig. 11 Difference of cumulative losses of AAP and batch models versus time on log prices

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

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

Figures 10 and 11 show the comparison of total losses of AAP-current and batch linear regression models for Ames house dataset for prices and logarithmic prices respectively. AAP-current consistently performs better than the seasonal batch model. Thus the straightforward utilisation of seasonality does not help.

When compared to the linear regression model of the first year, AAP-current initially has higher losses but it becomes better towards the end. It could be explained as follows. AAP-current needs time to train until it becomes good in prediction. These results show that we can make a better use of the past data with prediction with expert advice than with models trained in the batch mode. However, these results do not hold for logarithmic prices where the linear regression model of the first year outperforms AAP-current almost on the whole period of the dataset.

Appendix C: Improving predictions with inflation data

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

We used Acadata House Price Index (HPI) data[7] to improve the quality of our prediction. Every expert was adjusted on the basis of inflation data. For every month passed since the expert had been trained, we added to the log price it predicted the value of $\ln(1 + r)$, where

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

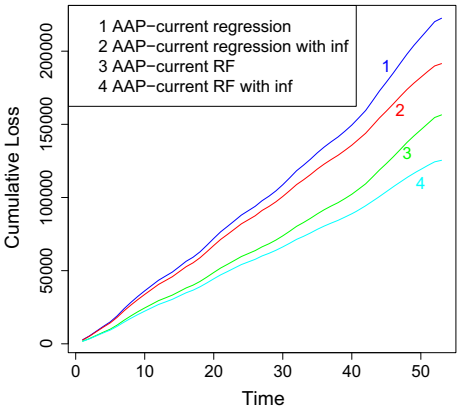
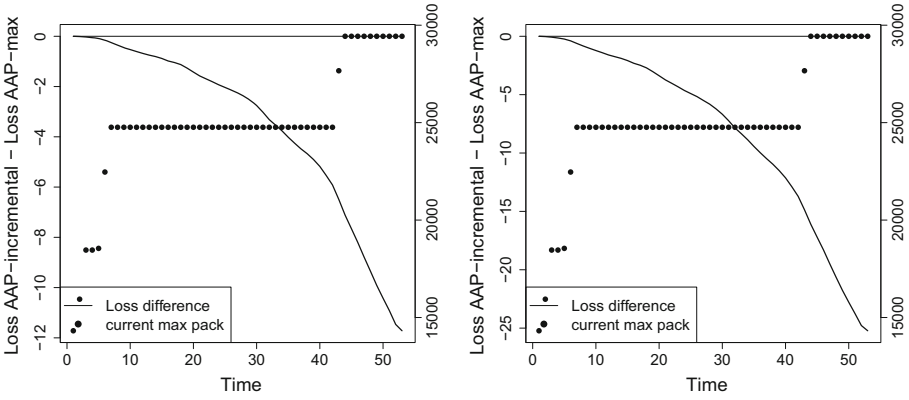
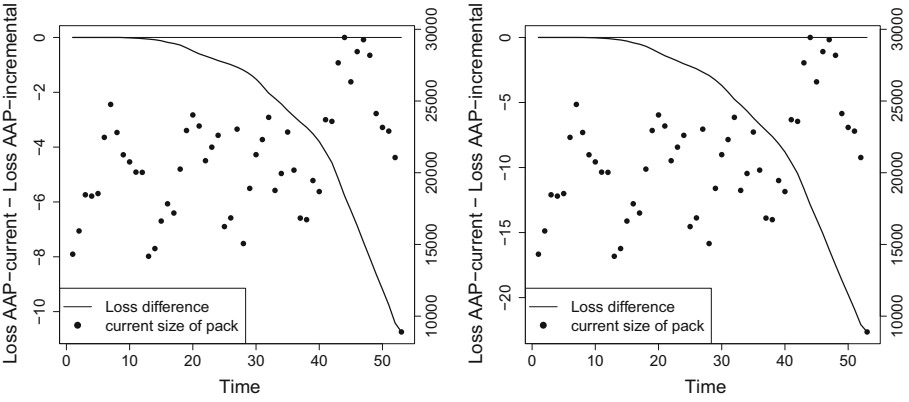


Fig. 12 Cumulative losses of AAP-current with experts adjusted and not adjusted for inflation



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

Fig. 13 Difference of cumulative losses of AAP-incremental and AAP-max versus time on log prices with inflation



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

Fig. 14 Difference of cumulative losses of AAP-current and AAP-incremental versus time on log prices with inflation

r is the monthly index calculated by Acadata. (The index for the month when transactions occurred was not used; we assumed this information is only available afterwards.)

Figure 12 shows the comparison of cumulative losses of AAP-current with and without inflation. It is clear from the graph that taking inflation into account improves both linear regression and random forests experts. As original experts were built on the first year of the dataset, they consistently under-estimate house prices for more recent data.

Figure 13 illustrates the comparison of total losses of AAP-incremental and AAP-max on log prices with experts adjusted for inflation. Figure 14 illustrates the comparison of total losses of AAP-current and AAP-incremental. The patterns are similar to what we previously observed: AAP-current consistently outperforms AAP-incremental, whereas AAP-incremental is better than AAP-max on the whole period of data.

References

Adamskiy, D., Koolen, W. M., Chernov, A., & Vovk, V. (2016). A closer look at adaptive regret. *The Journal of Machine Learning Research*, *17*(23), 1–21.

Bellotti, A. *Reliable region predictions for automated valuation models: supplementary material for Ames housing data*. Retrieved 20 December 2018, from http://wwwf.imperial.ac.uk/~abellott/publications.htm.

Bellotti, A. (2017). Reliable region predictions for automated valuation models. *Annals of Mathematics and Artificial Intelligence*, *81*, 71–74.

Cesa-Bianchi, N., & Lugosi, G. (2006). *Prediction, learning, and games*. Cambridge: Cambridge University Press.

De Cock, D. (2011). Ames, Iowa: Alternative to the Boston housing data as an end of semester regression project. *Journal of Statistics Education*, *19*(3). https://doi.org/10.1080/10691898.2011.11889627.

Joulani, P., Gyorgy, A., & Szepesvári, C.: Online learning under delayed feedback. In *Proceedings of the 30th international conference on machine learning (ICML-13)*, (pp. 1453–1461).

Kalnishkan, Y., Vovk, V., & Vyugin, M. V. (2004). Loss functions, complexities, and the Legendre transformation. *Theoretical Computer Science*, *313*(2), 195–207.

Kalnishkan, Y., Adamskiy, D., Chernov, A., & Scarfe, T.: Specialist experts for prediction with side information. In *2015 IEEE international conference on data mining workshop (ICDMW)*, (pp. 1470–1477). IEEE, (2015).

Littlestone, N., & Warmuth, M. K. (1994). The weighted majority algorithm. *Information and Computation*, *108*, 212–261.

Loève, M. (1977). *Probability theory I* (4th ed.). New York: Springer.

Quanrud, K., Khashabi, D. (2015). Online learning with adversarial delays. In *Advances in neural information processing systems*, (pp. 1270–1278).

Vovk, V. (1990). Aggregating strategies. In *Proceedings of the 3rd annual workshop on computational learning theory*, (pp. 371–383), San Mateo, CA: Morgan Kaufmann.

Vovk, V. (1998). A game of prediction with expert advice. *Journal of Computer and System Sciences*, *56*, 153–173.

Vovk, V. (2001). Competitive on-line statistics. *International Statistical Review*, *69*(2), 213–248.

Vovk, V., & Zhdanov, F. (2009). Prediction with expert advice for the Brier game. *Journal of Machine Learning Research*, *10*, 2445–2471.

Weinberger, M. J., & Ordentlich, E. (2002). On delayed prediction of individual sequences. *IEEE Transactions on Information Theory*, *48*(7), 1959–1976.

Zinkevich, M. (2003). Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th international conference on machine learning (ICML-03)*, (pp. 928–936).