Averaging Expert Predictions

Jyrki Kivinen[1] and Manfred K. Warmuth*[2]

[1] Department of Computer Science, P.O. Box 26 (Teollisuuskatu 23), FIN-00014 University of Helsinki, Finland; e-mail Jyrki.Kivinen@cs.Helsinki.FI
[2] Department of Computer Science, University of California, Santa Cruz, CA 95064, USA; e-mail manfred@cse.ucsc.edu

Abstract. We consider algorithms for combining advice from a set of experts. In each trial, the algorithm receives the predictions of the experts and produces its own prediction. A loss function is applied to measure the discrepancy between the predictions and actual observations. The algorithm keeps a weight for each expert. At each trial the weights are first used to help produce the prediction and then updated according to the observed outcome. Our starting point is Vovk's Aggregating Algorithm, in which the weights have a simple form: the weight of an expert decreases exponentially as a function of the loss incurred by the expert. The prediction of the Aggregating Algorithm is typically a nonlinear function of the weights and the experts' predictions. We analyze here a simplified algorithm in which the weights are as in the original Aggregating Algorithm, but the prediction is simply the weighted average of the experts' predictions. We show that for a large class of loss functions, even with the simplified prediction rule the additional loss of the algorithm over the loss of the best expert is at most $c \ln n$, where n is the number of experts and c a constant that depends on the loss function. Thus, the bound is of the same form as the known bounds for the Aggregating Algorithm, although the constants here are not quite as good. We use relative entropy to rewrite the bounds in a stronger form and to motivate the update.

1 Introduction

The focus of this paper is a certain class of on-line learning algorithms. In on-line learning the algorithm receives one by one a sequence of inputs x_t and makes after each x_t a *prediction* $\widehat{y}_t$. For each input x_t there is also a corresponding *outcome* (or desired output) y_t which is revealed to the learner after it has made its prediction $\widehat{y}_t$.

To define our on-line learning problem more closely, we need to specify which sequences $((x_1, y_1), \ldots, (x_\ell, y_\ell))$ are allowed as inputs, and what is the criterion for judging the quality of the predictions $\widehat{y}_t$. Regarding the input sequences, we take a worst-case view that given some domain X for the inputs and Y for the outcomes, for each t the pair (x_t, y_t) can be any element of $X \times Y$. In particular,

* Supported by NSF grant CCR 9700201

the pairs need not come from any probability distribution, and we make no assumptions about possible dependence between y_t and x_t. In this paper we consider mainly the case $X = [0,1]^n$ for some n and $Y = [0,1]$. Many of the results have obvious extensions to larger ranges of real inputs and outputs. We sometimes also consider the special case $Y = \{0,1\}$ where the outputs (but not the inputs) are required to be discrete.

To judge the quality of the predictions, we first introduce a *loss function* L that gives a (nonnegative) quantity $L(y_t, \widehat{y}_t)$ as a measure of discrepancy between the prediction and actual outcome. The square loss given by $L(y, \widehat{y}) = (y - \widehat{y})^2$ is a good example of a loss function suitable for our setting.

In addition to the loss function, it is essential to give a *comparison class* $\mathcal{F}$ of predictors as a reference point. The predictors are mappings from the set of possible inputs X to the set of possible predictions. We then define the *total loss* for an algorithm A that gives the predictions $\widehat{y}_t$ on a sequence $S = ((\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_\ell, y_\ell))$ as $\mathrm{Loss}_A(S) = \sum_{t=1}^{\ell} L(y_t, \widehat{y}_t)$, and similarly for a predictor $f \in \mathcal{F}$ as $\mathrm{Loss}_f(S) = \sum_{t=1}^{\ell} L(y_t, f(\boldsymbol{x}_t))$. We can measure the performance of our prediction algorithm by considering the additional loss $\mathrm{Loss}_A(S) - \inf_{f \in \mathcal{F}} \mathrm{Loss}_f(S)$ it incurs compared to the best fixed predictor from the comparison class. We call such performance bounds *relative loss bounds*.

In the extreme case that the outcomes y_t are completely random, the algorithm obviously cannot perform better than random guessing, but then neither can the predictors from the comparison class, so the additional loss can still be made small. In the more interesting extreme case that one predictor $f \in \mathcal{F}$ is perfect and we have $L(y_t, f(\boldsymbol{x}_t)) = 0$ for all t, the algorithm can still be allowed some initial interval of bad predictions, but to achieve a small additional loss it needs to quickly learn to make good predictions. Usually we are somewhere between these to extremes. Some predictors from the comparison class predict better than others, and the algorithm is required to perform roughly as well as the better ones.

In this paper the comparison classes we use come from the framework of predicting with expert advice [Vov90,CBFH+97]. We assume there are n experts, and the prediction of the ith expert for the tth outcome is given by $x_{t,i} \in [0,1]$. The vector $\boldsymbol{x}_t$ of all the experts' predictions at trial t is then the tth input vector to our algorithm. Hence, if we define $\mathcal{E}_i(\boldsymbol{x}) = x_i$, then $\mathrm{Loss}_{\mathcal{E}_i}(S)$ denotes the loss that the expert $\mathcal{E}_i$ would incur on the sequence S. The obvious thing to do now is to take as comparison class the set $\{\mathcal{E}_1, \ldots, \mathcal{E}_n\}$ of expert predictors and thus compare the loss of the algorithm to the loss $\min_i \mathrm{Loss}_{\mathcal{E}_i}(S)$ of the best single expert.

Earlier work on the expert framework by Vovk [Vov90] has shown that for a very general class of loss functions his *Aggregating Algorithm* (AA) achieves the bound

$$\mathrm{Loss}_{\mathrm{AA}}(S) \le \mathrm{Loss}_{\mathcal{E}_i}(S) + c_L \ln n \qquad \text{for all } i \tag{1}$$

where the constant c_L depends on the loss function. For example, with the square loss we have $c_L = 1/2$. This bound has also been shown to be essentially optimal [HKW98]. (Notice that for the important special case of absolute

loss $L(y, \widehat{y}) = |y - \widehat{y}|$, only bounds of a somewhat weaker form are possible [LW94,Vov90,CBFH$^+$97].) Vovk's Aggregating Algorithm is based on maintaining for each expert a weight that is decreased exponentially as the expert incurs loss. The predictions of the algorithm are of course affected more by the experts with large weights than by those with small weights, but the actual method of obtaining the prediction is somewhat more complicated than just taking a weighted average of the experts' predictions.

The main technical novelty in this paper is considering what happens if we keep using Vovk's algorithm for maintaining the weights but replace the prediction simply by the weighted average of the experts. Considering the optimality of Vovk's algorithm, we cannot hope to outperform it, but it turns out that for the simplified *Weighted Average Algorithm* (WAA) we can still prove the bound

$$\mathrm{Loss}_{\mathrm{WAA}}(S) \le \mathrm{Loss}_{\mathcal{E}_i}(S) + \widetilde{c}_L \ln n \qquad \text{for all } i \tag{2}$$

where $\widetilde{c}_L$ is a constant somewhat greater than c_L in (1). For example, with the square loss we have $\widetilde{c}_L = 2$ and $c_L = 1/2$.

The main reason why we want to consider the simplified prediction at the cost of slightly larger additional loss is that the simplified algorithm leads to simplified proofs of the relative loss bounds. Another intuitively appealing aspect of the weighted average as prediction is its probabilistic interpretation. If the negated loss $-L(y_t, x_{t,i})$ can be interpreted as the log likelihood of y_t given model $\mathcal{E}_i$, then the weight of the expert $\mathcal{E}_i$ after the trials can be interpreted as the posterior probability assigned to that expert. The prior probabilities here are the initial weights of the experts. In this setting, the prediction by weighted average correponds to the mean posterior prediction. The log loss, for which the log likelihood interpretation is most obvious, has been analyzed in this context before [Vov90,CBFH$^+$97,FSSW97]. It turns out that in the special case of log loss, the prediction of the Aggregating Algorithm also is the weighted average, so the Weighted Average Algorithm coincides with the original Aggregating Algorithm.

In reducing the algorithm's dependence on the particular loss function, the next step would be Freund and Schapire's Hedge Algorithm [FS97] that needs to assume only that the loss function has a bounded range. They can still prove loss bounds of the same flavor as the bounds here, but in the slightly weaker form of

$$\mathrm{Loss}_{\mathrm{Hedge}}(S) \le \mathrm{Loss}_{\mathcal{E}_i}(S) + a\sqrt{\mathrm{Loss}_{\mathcal{E}_i}(S))\ln n} + b \ln n \qquad \text{for all } i$$

for certain $a, b > 0$. Hence, there is a progression of algorithms where Vovk's original Aggregating Algorithm has a weight update that is uniform for all kinds of loss functions, but the prediction method is dependent on L. For the Weighted Average Algorithm, the prediction is made by the weighted average regardless of the loss function, but this happens at the cost of slightly worse constants in the loss bounds. Finally, the Hedge Algorithm is even more uniform in its treatment of loss functions, but the loss bounds get worse by more than just a constant. (Also notice that the bound for the Hedge Algorithm does not work with the unbounded log loss.)

After the technical remarks, consider now relating these results to a larger body of work where the *relative entropy* is the fundamental concept for motivating and analyzing learning algorithms [KW97]. Let $\boldsymbol{u} \in \mathbf{R}^n$ and $\boldsymbol{v} \in \mathbf{R}^n$ be *probability vectors*; i.e., $\sum_i u_i = \sum_i v_i = 1$ and $u_i, v_i \geq 0$ for all i. The relative entropy between $\boldsymbol{u}$ and $\boldsymbol{v}$ is then $d_{\mathrm{re}}(\boldsymbol{u}, \boldsymbol{v}) = \sum_{i=1}^n u_i \ln(u_i/v_i)$. To introduce relative entropy methods into the present problem, it is useful to start by considering a slightly extended comparison class. We define $\mathrm{Loss}_{\boldsymbol{u}}^{\mathrm{avg}}(S) = \sum_{t=1}^{\ell} \sum_{i=1}^n u_i L(y_t, x_{t,i})$ to be the expected loss if we predict by a random expert chosen according to $\boldsymbol{u}$. We first rewrite Vovk's original proof in order to bring out how the additional loss incurred by the algorithm relates to a relative entropy. The resulting bound is

$$\mathrm{Loss}_{\mathrm{WAA}}(S) \leq \mathrm{Loss}_{\boldsymbol{u}}^{\mathrm{avg}}(S) + \widetilde{c}_L \, d_{\mathrm{re}}(\boldsymbol{u}, \boldsymbol{v}_1) \tag{3}$$

where $\boldsymbol{v}_1$ is the algorithm's initial weigth vector. With $\boldsymbol{v}_1 = (1/n, \ldots, 1/n)$, and $u_i = 1$ and $u_j = 0$ for $j \neq i$, this simplifies to bound (2) where comparison is against the single best expert $\mathcal{E}_i$. Note that since always $\mathrm{Loss}_{\boldsymbol{u}}^{\mathrm{avg}}(S) \geq \min_i \mathrm{Loss}_{\mathcal{E}_i}(S)$, going from (2) to (3) does not bring any improvement in the first term of the bound. However, improvement in the second term are possible. If there are several expert with nearly optimal performance, then substituting into (3) a comparison vector $\boldsymbol{u}$ that distributes the weight nearly evenly among the good experts gives a significantly sharper bound than (2). As a simple example, assume that k experts all have some small loss Q. Then (2) gives the loss bound $Q + \widetilde{c}_L \ln n$ while the bound (3) goes down to $Q + \widetilde{c}_L \ln(n/k)$. The new method brings out in a more explicit form the feature implicit in earlier proofs (see, e.g., [LW94,Vov90]) that having more than one good expert results in a smaller additional loss. For log loss this feature, with bounds of the form (3) and proofs analogous to ours, was already pointed out in [FSSW97].

Our second use for relative entropy is as a regularizing term in setting up a minimization problem that gives Vovk's rule for updating the weights. The basic idea in such a derivation (see [KW97,HKW95] for other examples) is to see the update as an act of balancing the need to maintain old information by staying close to the old weight vector and the need to learn by moving the weights in the direction of small loss on the last example.

In Sect. 2 we review the basic expert framework and Vovk's algorithm. Sect. 3 gives the new upper bound for the additional loss achieved by the modified algorithm that predicts with the weighted combination of experts. A straightforward proof is given in Sect. 4. In Sect. 5 we restate the bound and proof using a relative entropy, and give a motivation for the algorithm in terms of a relative entropy minimization problem. Finally, in Sect. 6 we generalize the relative loss bounds for the new algorithm to multi-dimensional predictions and outcomes.

2 The Setting and the Algorithm

We consider a simple on-line prediction setting, where learning takes place during a sequence of *trials*. At trial t, the learner tries to predict a real-valued *outcome*

y_t. The learner's prediction is denoted by $\widehat{y}_t$, and the performance of the learner is measured by using a *loss function* L. Loss functions will be discussed in more detail in Sect. 3, but for understanding the algorithm it is sufficient to think of, say, the square loss given by $L(y, \widehat{y}) = (y - \widehat{y})^2$. The learner bases its prediction $\widehat{y}_t$ on an *instance* $\boldsymbol{x}_t$. In the expert-based framework we use here, we imagine there is a set of *experts* $\mathcal{E}_i$, $i = 1, \ldots, n$, and the instance $\boldsymbol{x}_t$ is an n-dimensional vector where the ith component $x_{t,i}$ of the tth instance can be interpreted as the prediction given by expert $\mathcal{E}_i$ for the outcome y_t.

We consider here a specific kind of algorithm based on maintaining a weight on each expert. The weight vector $\boldsymbol{v}_t$ is normalized to be a *probability vector* (i.e., $\sum_i v_i = 1$, $v_i \geq 0$), and $v_{t,i}$ can be interpreted as the algorithm's belief in the expert $\mathcal{E}_i$ having the best prediction at the trial t. The prediction of the algorithm at trial t is given by the weighted average $\widehat{y}_t = \boldsymbol{v}_t \cdot \boldsymbol{x}_t$. After seeing the outcome y_t, the algorithm updates its weights. The update method and all other details of the *Weighted Average Algorithm* (WAA) we consider here are given in Figure 1.

Sometimes it is more convenient to express the update in terms of the unnormalized weights

$$w_{t,i} = w_{1,i} \exp\left(-\frac{1}{c} \sum_{j=1}^{t-1} L(y_j, x_{j,i})\right) \tag{4}$$

where $w_{1,i} = v_{1,i}$. Now $v_{t,i} = w_{t,i}/W_t$ where $W_t = \sum_{i=1}^{n} w_{t,i}$ is the normalization factor. Thus, ignoring the normalization factor, the logarithm of the weight of an expert is proportional to the expert's accumulated loss from preceding trials. We call this the *loss update* to emphasize that only the values of the loss function (and not its gradient etc.) are used.

The loss update of the Weighted Average Algorithm was introduced by Vovk [Vov90] in his Aggregating Algorithm (AA) that generalized the Weighted Majority algorithm [LW94]. However, the prediction of the Aggregating Algorithm is usually given by a function that is non-linear in $\boldsymbol{v}_t$ and depends on the loss function. In contrast, we use the fixed prediction function $\widehat{y}_t = \boldsymbol{v}_t \cdot \boldsymbol{x}_t$ for all loss functions. (A notable special case is the log loss, for which the Aggregating Algorithm also predicts with $\widehat{y}_t = \boldsymbol{v}_t \cdot \boldsymbol{x}_t$.)

3 Basic Loss Bounds

We begin with a short discussion of some basic properties of loss functions. The definitions of the loss functions most interesting to us are given in Table 1. For a loss function L, we define $L_y(\widehat{y}) = L(y, \widehat{y})$ for convenience in writing derivatives with respect to $\widehat{y}$. Note that with the exception of the absolute loss, all the loss functions given in Table 1 are *convex*, i.e., $L''_y(x) > 0$ for all x and y, and also satisfy $L'_y(y) = 0$ for $0 < y < 1$. This implies *monotonicity*, i.e., $L'_y(x) < 0$ for $x < y$ and $L'_y(x) > 0$ for $x > y$. We generalize the derivative notation also for the end points by defining $L'_0(0) = L'_1(1) = 0$. The absolute loss $L(y, \widehat{y}) = |y - \widehat{y}|$

Initialize the weights to some probability vector $v_{1,i}$;
set the parameter c to some positive value.
Repeat for $t = 1, \ldots, \ell$:

1. Receive the instance $\boldsymbol{x}_t$.
2. Output the prediction $\widehat{y_t} = \boldsymbol{v}_t \cdot \boldsymbol{x}_t$.
3. Receive the outcome y_t.
4. Update the weights by the *loss update* defined as follows:

$$v_{t+1,i} = v_{t,i} \exp(-L(y_t, x_{t,i})/c)/\mathrm{norm}_t$$

where

$$\mathrm{norm}_t = \sum_{i=1}^{n} v_{t,i} \exp(-L(y_t, x_{t,i})/c) \ .$$

Fig. 1. The Weighted Average Algorithm (WAA) for combining expert predictions

(and other loss functions that are not continuously differentiable) is not covered by the bounds given in this paper.

Given some fixed loss function L, consider now the total loss

$$\mathrm{Loss}_A(S) = \sum_{t=1}^{\ell} L(y_t, \widehat{y_t})$$

suffered by some algorithm A on the trial sequence with the instance-outcome pairs $S = ((\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_\ell, y_\ell))$. We wish to prove upper bounds for this total loss without making statistical or other assumptions about how the instances and outcomes are generated. When no such assumptions are made, one suitable way of measuring the quality of the learner's predictions is to compare it against the losses incurred by the individual experts on the same sequence. Thus, we also define $\mathrm{Loss}_{\mathcal{E}_i}(S) = \sum_{t=1}^{\ell} L(y_t, x_{t,i})$.

Consider first the known bounds for the Aggregating Algorithm, which uses the same weights $\boldsymbol{v}_t$ as the algorithm of Figure 1 but a different prediction $\widehat{y_t}$. To state the optimal constants in the bounds, and the learning rates that lead to them, define for $z, p, q \in [0, 1]$ (where z should be interpreted as a "prediction" and p and q as two possible "outcomes") the ratio

$$R(z, p, q) = \frac{L'_p(z)L'_q(z)^2 - L'_q(z)L'_p(z)^2}{L'_p(z)L''_q(z) - L'_q(z)L''_p(z)} \ ;$$

we define $R(z, p, q) = 0$ in the special case $p = q$. Let further

$$c_L = \sup_{0 \le z,p,q \le 1} R(z, p, q) \ .$$

The bound for the Aggregating Algorithm originally given by Vovk [Vov90] can now be stated as follows.

Table 1. Some common loss functions for the domain $[0,1] \times [0,1]$

loss function L	value for $L(y, \widehat{y})$
square loss	$(y - \widehat{y})^2$
relative entropy loss	$(1-y)\ln((1-y)/(1-\widehat{y})) + y\ln(y/\widehat{y})$
Hellinger loss	$\frac{1}{2}\left(\left(\sqrt{1-y} - \sqrt{1-\widehat{y}}\right)^2 + \left(\sqrt{y} - \sqrt{\widehat{y}}\right)^2\right)$
absolute loss	$\lvert y - \widehat{y}\rvert$

Theorem 1. *Let L be a convex monotone twice differentiable loss function and* AA *be the Aggregating Algorithm with $c \geq c_L$ and initial weights $w_{1,i} = 1$. Then for any sequence $S = ((\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_\ell, y_\ell))$ we have*

$$\mathrm{Loss}_{\mathrm{AA}}(S) \leq \left(\min_i \mathrm{Loss}_{\mathcal{E}_i}(S)\right) + c \ln n \ . \tag{5}$$

The Aggregating Algorithm was also considered by Haussler et al. [HKW98], who showed the bound (5) optimal in the sense that under some reasonable regularity conditions, for *any* on-line algorithm A there are sequences S such that

$$\mathrm{Loss}_A(S) \geq \left(\min_i \mathrm{Loss}_{\mathcal{E}_i}(S)\right) + c_L \ln n - o(1) \ ,$$

where $o(1)$ approaches 0 as n and ℓ approach ∞ in a suitable manner.

Vovk and Haussler et al. were mainly interested in the binary case $y_t \in \{0, 1\}$ and actually state (5) only for that case in the form

$$\mathrm{Loss}_{\mathrm{AA}}(S) \leq \left(\min_i \mathrm{Loss}_{\mathcal{E}_i}(S)\right) + c_{L,\mathrm{bin}} \ln n \tag{6}$$

where $c_{L,\mathrm{bin}} = \sup_z R(z, 0, 1)$. The actual proof of Theorem 1 is a simple generalization of the earlier proofs [Vov90,HKW98] for (6); we omit it here. Haussler et al. also use some special techniques to show that for certain loss functions such as the square loss and the relative entropy loss the bound (6) holds even when y_t is allowed to range over the whole interval $[0, 1]$. (The value of the constant for Hellinger loss for continuous-valued outcomes was left open in [HKW98].) The new formulation of Theorem 1 gives a unified method of obtaining bounds in the continuous-valued case. For square, relative entropy, and Hellinger loss a straightforward proof (omitted) shows that we actually have $c_L = c_{L,\mathrm{bin}}$, so the bound is the same for continuous-valued and binary outcomes.

The main content of the bound (5) is that even for a large number of experts, the loss of the algorithm exceeds the loss of the best expert only by a small additive constant, regardless of the number of trials. Thus, the algorithm is good at weeding out the bad experts and then following the good ones. We can prove a similar bound for the Weighted Average Algorithm that predicts with $\widehat{y}_t = \boldsymbol{v}_t \cdot \boldsymbol{x}_t$. Define

$$\widetilde{R}(z, p) = \frac{L_p'(z)^2}{L_p''(z)} \tag{7}$$

Table 2. Comparison of the constants in bounds (5) and (9) for various loss functions.

loss function L	c_L	$\widetilde{c}_L$
relative entropy	1	1
square	1/2	2
Hellinger	$2^{-1/2} \approx 0.71$	1

and

$$\widetilde{c}_L = \sup_{0<z,p<1} \widetilde{R}(z,p) \ . \tag{8}$$

We can now state the bound for WAA as follows.

Theorem 2. *Let L be a monotone convex twice differentiable loss function and* WAA *be the Weighted Average Algorithm of Figure 1 with uniform initial weights $w_{1,i} = 1$ and with $c \geq \widetilde{c}_L$. Then for any sequence $S = ((\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_\ell, y_\ell))$ we have*

$$\mathrm{Loss}_{\mathrm{WAA}}(S) \leq \left(\min_i \mathrm{Loss}_{\mathcal{E}_i}(S)\right) + c \ln n \ . \tag{9}$$

A generalization for multi-dimensional predictions and outcomes is given in Sect. 6.

To compare the constants c_L and $\widetilde{c}_L$ in (5) and (9), respectively, recall that $(a+b)/(a'+b') \leq \max\{\, a/a', b/b' \,\}$ for any $a, a', b, b' > 0$. From this it is immediate that $c_L \leq \widetilde{c}_L$. For the most usual cases (9) is strictly worse than (5), as can be seen from the comparison in Table 2. For the relative entropy loss the bouds are actually equal, which is no surprise since then also the algorithms are the same (i.e., the Aggregating Algorithm also predicts with $\widehat{y}_t = \boldsymbol{v}_t \cdot \boldsymbol{x}_t$).

4 The Basic Upper Bound Proof

We apply to our situation the potential function method commonly used in computer science to analyze on-line algorithms. Thus, we introduce a *potential* P, with the value P_t describing the algorithm's state just prior to trial t. Then $P_t - P_{t+1}$ is the decrease in the potential due to trial t. The key in proving the loss bound for an algorithm A is to show for each trial t that the prediction $\widehat{y}_t$ of A satisfies

$$L(y_t, \widehat{y}_t) \leq P_t - P_{t+1} \ , \tag{10}$$

from which summing over $t = 1, \ldots, \ell$ yields $\mathrm{Loss}_A(S) \leq P_1 - P_{\ell+1}$. That is, the total loss of the algorithm is bounded by the total decrease in potential. The basic question now is, how to choose the potential P such that the equation (10) can be satisfied by a suitable choice of the prediction $\widehat{y}_t$, and the total increase of the potential gives interesting loss bounds. This question was originally answered for general loss functions by Vovk [Vov90] who generalized the potential used in [LW94] for the absolute loss. We shall next review Vovk's method for obtaining total loss bounds from (10) using our notation and then show how (10) can be

achieved by the prediction $\widehat{y}_t = \boldsymbol{v}_t \cdot \boldsymbol{x}_t$ with slightly worse constants than with Vovk's original prediction.

First, recall from Sect. 2 that our algorithm has at trial t an n-dimensional weight vector $\boldsymbol{w}_t$ defined in (4), and we write $W_t = \sum_{i=1}^{n} w_{t,i}$. As our potential we now choose

$$P_t = c \ln W_t \tag{11}$$

where $c > 0$ is the same constant that is used in the updates. As it turns out, multiplying the weights by a constant affects neither the algorithm nor our analysis of it. Regarding the potentials in particular, multiplying the weights by a positive constant a translates into adding the constant $c \ln a$ to the potential, which leaves potential differences unaffected. Thus, without loss of generality we can scale the initial weights so that $W_1 = 1$ holds, and $P_1 = 0$.

Elaborating further on our loss bound we get

$$\begin{aligned}
\mathrm{Loss}_A(S) &\le P_1 - P_{\ell+1} \\
&= -c \ln \sum_{i=1}^{n} w_{1,i} \exp(-\mathrm{Loss}_{\mathcal{E}_i}(S)/c) \\
&\le -c \ln w_{1,i} \exp(-\mathrm{Loss}_{\mathcal{E}_i}(S)/c) \\
&= \mathrm{Loss}_{\mathcal{E}_i}(S) - c \ln w_{1,i}
\end{aligned}$$

for any given expert i. In particular, in the absence of any other preference it seems natural to set all the initial weights equal, which gives $w_{1,i} = 1/n$ for all i and thus results in the final bound

$$\mathrm{Loss}_A(S) \le \Big(\min_i \mathrm{Loss}_{\mathcal{E}_i}(S)\Big) + c \ln n \ . \tag{12}$$

To prove Theorem 2, it thus remains to show that (10) is satisfied for the Weighted Average Algorithm. This turns out to be true for all y_t and $\boldsymbol{x}_t$ exactly when the constant c satisfies the condition of the theorem.

To prove (10), first write the potential difference in the form

$$P_t - P_{t+1} = -c \ln \frac{W_{t+1}}{W_t} = -c \ln \sum_{i=1}^{n} v_{t,i} \exp(-L(y_t, x_{t,i})/c)$$

where $v_{t,i} = w_{t,i}/W_t$ is the normalized ith weight. We use the normalized weight vector in the prediction by choosing $\widehat{y}_t = \boldsymbol{v}_t \cdot \boldsymbol{x}_t$. Then (10) becomes

$$L(y_t, \boldsymbol{v}_t \cdot \boldsymbol{x}_t) \le -c \ln \sum_{i=1}^{n} v_{t,i} \exp(-L(y_t, x_{t,i})/c) \ ,$$

or equivalently

$$\exp(-L(y_t, \boldsymbol{v}_t \cdot \boldsymbol{x}_t)/c) \ge \sum_{i=1}^{n} v_{t,i} \exp(-L(y_t, x_{t,i})/c) \ .$$

If we define $f_y(x) = \exp(-L(y,x)/c)$, (10) therefore is equivalent with

$$f_{y_t}\left(\sum_{i=1}^{n} v_{t,i} x_{t,i}\right) \geq \sum_{i=1}^{n} v_{t,i} f_{y_t}(x_{t,i}) \ .$$

Since $\boldsymbol{v}_t$ is a probability vector, this holds by Jensen's inequality if f_{y_t} is concave.

Using the notation $L_y(x) = L(y,x)$, we have

$$f'_y(x) = (-L'_y(x)/c)\exp(-L_y(x)/c)$$

and

$$f''_y(x) = ((L'_y(x)/c)^2 - L''_y(x)/c)\exp(-L_y(x)/c) \ .$$

Hence, since we assume $L''_y(x)$ to be positive, $f''_y(x) \leq 0$ holds if and only if $c \geq L'_y(x)^2/L''_y(x)$. Therefore, (10) holds for the prediction $\widehat{y}_t = \boldsymbol{v}_t \cdot \boldsymbol{x}_t$ if the constant c satisfies

$$c \geq \frac{L'_{y_t}(x_{t,i})^2}{L''_{y_t}(x_{t,i})} \qquad \text{for } i = 1, \ldots, n \ .$$

This concludes the proof of Theorem 2.

The result can be generalized to multi-dimensional predictions, as we see in Sect. 6.

5 Bounds Based on the Relative Entropy

We now wish to consider bounds in which the loss of the algorithm is compared not to the loss of the best single expert, but the loss of the best *probabilistic combination* of the experts. In particular, assume that at trial t we predict according to the prediction of an expert chosen at random, with expert $\mathcal{E}_i$ having probability u_i of being chosen. For such probabilistic predictions, the expected loss over the whole sequence is given by

$$\mathrm{Loss}^{\mathrm{avg}}_{\boldsymbol{u}}(S) = \sum_{i=1}^{n} u_i \mathrm{Loss}_{\mathcal{E}_i}(S) = \sum_{t=1}^{\ell} \boldsymbol{u} \cdot \boldsymbol{L}_t \ , \quad ,$$

where $\boldsymbol{L}_t$ denotes the vector of losses of the experts at trial t, i.e., $L_{t,i} = L(y_t, x_{t,i})$.

As discussed in the introduction, we wish to bound the loss of the algorithm in terms of the average loss $\mathrm{Loss}^{\mathrm{avg}}_{\boldsymbol{u}}(S)$ and the distance $d(\boldsymbol{u}, \boldsymbol{v}_1)$ between $\boldsymbol{u}$ and the algorithm's initial weight vector $\boldsymbol{v}_1$ for some natural distance function d. For both the Aggregating Algorithm and the Weighted Average Algorithm, the most suitable distance is the *relative entropy* given by $d_{\mathrm{re}}(\boldsymbol{u}, \boldsymbol{v}) = \sum_{i=1}^{n} u_i \ln(u_i/v_i)$. Our bound is then as follows.

Theorem 3. *Let L be a monotone convex twice differentiable loss function, and let the Weighted Average Algorithm* WAA *use arbitrary initial weights $\boldsymbol{v}_1$ and parameter $c = \widetilde{c}_L$, where $\widetilde{c}_L$ is as in (8). Then for any sequence $S = ((\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_\ell, y_\ell))$ and for all probability vectors $\boldsymbol{u}$ we have*

$$\mathrm{Loss}_{\mathrm{WAA}}(S) \le \mathrm{Loss}_{\boldsymbol{u}}^{\mathrm{avg}}(S) + \widetilde{c}_L \, d_{\mathrm{re}}(\boldsymbol{u}, \boldsymbol{v}_1) \ . \tag{13}$$

It is easy to see that also in Vovk's original analysis one can use the distance $d_{\mathrm{re}}(\boldsymbol{u}, \boldsymbol{v}_t)$ as done in the above bound. As a result one gets for the Aggregating Algorithm a bound like (13) with c_L instead of $\widetilde{c}_L$.

Proof of Theorem 3: We express the progress towards the reference vector $\boldsymbol{u}$ as follows:

$$\begin{aligned} d_{\mathrm{re}}(\boldsymbol{u}, \boldsymbol{v}_t) - d_{\mathrm{re}}(\boldsymbol{u}, \boldsymbol{v}_{t+1}) &= \sum_{i=1}^{n} u_i \ln \frac{v_{t+1,i}}{v_{t,i}} \\ &= \sum_{i=1}^{n} u_i \ln \frac{w_{t+1,i} W_t}{w_{t,i} W_{t+1}} \\ &= -\boldsymbol{u} \cdot \boldsymbol{L}_t / c + \sum_{i=1}^{n} u_i \ln \frac{W_t}{W_{t+1}} \\ &= -\boldsymbol{u} \cdot \boldsymbol{L}_t / c + (P_t - P_{t+1}) / c \ . \end{aligned} \tag{14}$$

Applying (10) now yields

$$L(y_t, \widehat{y}_t) \le P_t - P_{t+1} = \boldsymbol{u} \cdot \boldsymbol{L}_t + c \, (d_{\mathrm{re}}(\boldsymbol{u}, \boldsymbol{v}_t) - d_{\mathrm{re}}(\boldsymbol{u}, \boldsymbol{v}_{t+1})) \ .$$

Summing over all the trials we obtain

$$\mathrm{Loss}_{\mathrm{WAA}}(S) \le P_1 - P_{\ell+1} = \mathrm{Loss}_{\boldsymbol{u}}^{\mathrm{avg}}(S) + c \, (d_{\mathrm{re}}(\boldsymbol{u}, \boldsymbol{v}_1) - d_{\mathrm{re}}(\boldsymbol{u}, \boldsymbol{v}_{\ell+1})) \ . \tag{15}$$

Omitting the non-negative distance $d_{\mathrm{re}}(\boldsymbol{u}, \boldsymbol{v}_{\ell+1})$ gives the bound (13) of the theorem. □

To see some interesting details of the proof, notice that in (14), the probability vector $\boldsymbol{u}$ is arbitrary. So in particular we can choose $\boldsymbol{u} = \boldsymbol{v}_t$ and thus obtain

$$-d_{\mathrm{re}}(\boldsymbol{v}_t, \boldsymbol{v}_{t+1}) = -\boldsymbol{v}_t \cdot \boldsymbol{L}_t / c + (P_t - P_{t+1}) / c \ . \tag{16}$$

Combining (14) and (16) gives us the following fundamental connection between distances and average losses:

$$\boldsymbol{v}_t \cdot \boldsymbol{L}_t = \boldsymbol{u} \cdot \boldsymbol{L}_t + c \, \{d_{\mathrm{re}}(\boldsymbol{u}, \boldsymbol{v}_t) - d_{\mathrm{re}}(\boldsymbol{u}, \boldsymbol{v}_{t+1}) + d_{\mathrm{re}}(\boldsymbol{v}_t, \boldsymbol{v}_{t+1})\} \ .$$

We conclude this section by pointing out a strong relationship between the update of the algorithm and the bound (13). One can show that the probability vector $\boldsymbol{u}$ that minimizes the right-hand side of the bound (13) is $\boldsymbol{v}_{\ell+1}$. With

this minimizer $\boldsymbol{u} = \boldsymbol{v}_{\ell+1}$ the value of the bound equals $P_1 - P_{\ell+1}$ (which is the constant value of the right-hand side of (15)). Thus, the weight vector $\boldsymbol{v}_{t+1}$ produced by the loss update at the end of trial t is the minimizer of the bound (13) with respect to the first t examples, and with this minimizer the bound on the first t examples becomes $P_1 - P_{t+1}$.

Alternatively, the update of the algorithm can be derived in an on-line fashion as $\boldsymbol{v}_{t+1} = \mathrm{argmin}_{\boldsymbol{v}} U_t(\boldsymbol{v})$ where

$$U_t(\boldsymbol{v}) = c\, d_{\mathrm{re}}(\boldsymbol{v}, \boldsymbol{v}_t) + \boldsymbol{v} \cdot \boldsymbol{L}_t$$

and $\boldsymbol{v}$ is constrained to be a probability vector. Again, substituting the minimizing argument into U_t gives a potential difference, namely

$$P_t - P_{t+1} = U_t(\boldsymbol{v}_{t+1}) \leq U_t(\boldsymbol{v}_t) = \boldsymbol{v}_t \cdot \boldsymbol{L}_t \ .$$

Note that the above upper bound for $P_t - P_{t+1}$ is complemented by the lower bound (11) that is central to the relative loss bounds proven for the expert setting.

If we want to compare the loss of the algorithm to $L(y_t, \boldsymbol{u} \cdot \boldsymbol{x}_t)$ instead of $\boldsymbol{u} \cdot \boldsymbol{L}_t$, a better update might result from $\boldsymbol{v}_{t+1} = \mathrm{argmin}_{\boldsymbol{v}} \hat{U}_t(\boldsymbol{v})$ where

$$\hat{U}_t(\boldsymbol{v}) = c\, d_{\mathrm{re}}(\boldsymbol{v}, \boldsymbol{v}_t) + L(y_t, \boldsymbol{v} \cdot \boldsymbol{x}_t)$$

and again $\boldsymbol{v}$ is constrained to be a probability vector. If the loss function is convex then $L(y_t, \boldsymbol{v} \cdot \boldsymbol{x}_t) \leq \boldsymbol{v} \cdot \boldsymbol{L}_t$ and $U_t(\boldsymbol{v})$ bounds $\hat{U}_t(\boldsymbol{v})$ from above. The bounds that can be obtained for algorithms based on minimizing $\hat{U}_t$ [KW97,HKW95] differ significantly from the style of bounds we have here. When the loss $L(y_t, \widehat{y}_t)$ of the algorithm is compared to $L(y_t, \boldsymbol{u} \cdot \boldsymbol{x}_t)$, it is usually impossible to bound the additional loss by a constant (such as $\widetilde{c}_L \ln n$ here). However, bounds where the comparison is to $L(y_t, \boldsymbol{u} \cdot \boldsymbol{x}_t)$ are in some sense much stronger than the expert style bounds of this paper.

6 Multi-dimensional predictions

We now consider briefly the case of multi-dimensional predictions. In other words, instead of having real numbers as outcomes y_t, experts' predictions $x_{t,i}$, and predictions $\widehat{y}_t$, we now have vectors from (some subset of) $\mathbf{R}^k$, for some $k \geq 1$. For instance, the experts' predictions and the outcomes might be from the k-dimensional unit ball $\left\{ \boldsymbol{x} \in \mathbf{R}^k \mid ||\boldsymbol{x}||_2 \leq 1 \right\}$. Since the prediction of each individual expert at a given time t is a k-dimensional vector, all the expert predictions at time t constitute a $k \times n$ matrix $\boldsymbol{X}_t$. The prediction of the algorithm will still be a weighted average (i.e., convex combination) of the experts' predictions: $\widehat{\boldsymbol{y}}_t = \boldsymbol{X}_t \boldsymbol{v}_t$ where the weight vector $\boldsymbol{v}_t$ is maintained by multiplicative updates as before. A loss function is now defined on $\mathbf{R}^k \times \mathbf{R}^k$; a simple examples would be $L(\boldsymbol{y}, \widehat{\boldsymbol{y}}) = ||\boldsymbol{y} - \widehat{\boldsymbol{y}}||_2^2$.

Consider now the proof of our main result Theorem 2. The only place where we use the fact that the values y_t and $x_{t,i}$ are real numbers is in proving that

the function f_y defined by $f_y(x) = \exp(-L(y,x)/c)$ is concave for all y. We do this proof by considering the sign of the second derivative of f_y.

In the multi-dimensional case, we analogously need to prove that the function $f_{\boldsymbol{y}}$ defined by $f_{\boldsymbol{y}}(\boldsymbol{x}) = \exp(-L(\boldsymbol{y},\boldsymbol{x})/c)$ is concave. If we find a value for c such that this holds, then the rest of the proof goes as before and we again obtain the familiar bound $\mathrm{Loss}_{\mathrm{WAA}}(S) \le (\min_i \mathrm{Loss}_{\mathcal{E}_i}(S)) + c\ln n$. Alternatively we can use the relative entropy as in Sect. 5 and obtain the bound $\mathrm{Loss}_{\mathrm{WAA}}(S) \le \mathrm{Loss}^{\mathrm{avg}}_{\boldsymbol{u}}(S) + c\, d_{\mathrm{re}}(\boldsymbol{u}, \boldsymbol{v}_1)$ for any probability vector $\boldsymbol{u}$.

Consider now when $f_{\boldsymbol{y}}$ is concave. Let us denote the gradient and Hessian of $f_{\boldsymbol{y}}$ by $\nabla f_{\boldsymbol{y}}$ and $\mathrm{D}^2 f_{\boldsymbol{y}}$, respectively. We need to find out when $\mathrm{D}^2 f_{\boldsymbol{y}}$ is negative semidefinite everywhere. Thus, we have

$$\left(\nabla f_{\boldsymbol{y}}(\boldsymbol{x})\right)_i = \frac{\partial f_{\boldsymbol{y}}(\boldsymbol{x})}{\partial x_i} = -\frac{1}{c} f_{\boldsymbol{y}}(\boldsymbol{x}) \frac{\partial L(\boldsymbol{y},\boldsymbol{x})}{\partial x_i}$$

and

$$\left(\mathrm{D}^2 f_{\boldsymbol{y}}(\boldsymbol{x})\right)_{ij} = \frac{\partial^2 f_{\boldsymbol{y}}(\boldsymbol{x})}{\partial x_i \partial x_j} = \frac{1}{c} f_{\boldsymbol{y}}(\boldsymbol{x}) \left(\frac{1}{c} \frac{\partial L(\boldsymbol{y},\boldsymbol{x})}{\partial x_i} \frac{\partial L(\boldsymbol{y},\boldsymbol{x})}{\partial x_j} - \frac{\partial^2 L(\boldsymbol{y},\boldsymbol{x})}{\partial x_j \partial x_j} \right) \ .$$

For $\boldsymbol{z} \in \mathbf{R}^k$ we now have $\boldsymbol{z}^{\mathrm{T}} \mathrm{D}^2 f_{\boldsymbol{y}}(\boldsymbol{x})\, \boldsymbol{z} \le 0$ if and only if

$$\left(\boldsymbol{z} \cdot \nabla L_{\boldsymbol{y}}(\boldsymbol{x})\right)^2 / c - \boldsymbol{z}^{\mathrm{T}} \mathrm{D}^2 L_{\boldsymbol{y}}(\boldsymbol{x})\, \boldsymbol{z} \le 0 \ . \tag{17}$$

Note that in order to have this hold for all z we at least need to have $\boldsymbol{z}^{\mathrm{T}} \mathrm{D}^2 L_{\boldsymbol{y}}(\boldsymbol{x})\, \boldsymbol{z}$ positive, i.e., the loss $L(\boldsymbol{y},\boldsymbol{x})$ needs to be convex in $\boldsymbol{x}$. In this case we get for c the condition

$$c \ge \sup_{\boldsymbol{z},\boldsymbol{y},\boldsymbol{x}} \frac{\left(\boldsymbol{z} \cdot \nabla L_{\boldsymbol{y}}(\boldsymbol{x})\right)^2}{\boldsymbol{z}^{\mathrm{T}} \mathrm{D}^2 L_{\boldsymbol{y}}(\boldsymbol{x}) \boldsymbol{z}}$$

where $\boldsymbol{y}$ and $\boldsymbol{x}$ in the supremum range over the possible values of outcomes and (single) experts' predictions, respectively, and $\boldsymbol{z}$ ranges over $\mathbf{R}^k$. Comparing this with the constant $\widetilde{c}_L$ defined in (8), we see that the first and second derivatives there are here in some sense replaced with first and second derivatives in some direction $\boldsymbol{z}$, where the direction $\boldsymbol{z}$ is chosen as the worst case.

As a first example, consider the square loss $L(\boldsymbol{y},\boldsymbol{x}) = ||\boldsymbol{y} - \boldsymbol{x}||_2^2$. Then $\nabla L_{\boldsymbol{y}}(\boldsymbol{x}) = 2(\boldsymbol{x} - \boldsymbol{y})$, and $\mathrm{D}^2 L_{\boldsymbol{y}}(\boldsymbol{x}) = 2\boldsymbol{I}$ where $\boldsymbol{I}$ is the identity matrix. Hence, we get

$$\frac{\left(\boldsymbol{z} \cdot \nabla L_{\boldsymbol{y}}(\boldsymbol{x})\right)^2}{\boldsymbol{z}^{\mathrm{T}} \mathrm{D}^2 L_{\boldsymbol{y}}(\boldsymbol{x}) \boldsymbol{z}} = \frac{\left(\boldsymbol{z} \cdot (2\boldsymbol{x} - 2\boldsymbol{y})\right)^2}{2\boldsymbol{z}^2} \ ,$$

and this expression obtains its maximum value $2(\boldsymbol{x} - \boldsymbol{y})^2$ when $\boldsymbol{z}$ is parallel to $\boldsymbol{x} - \boldsymbol{y}$. Hence, if the outcomes $\boldsymbol{y}_t$ and the experts' predictions $\boldsymbol{x}_{t,i}$ are from a ball of radius R, so $(\boldsymbol{x} - \boldsymbol{y})^2 \le 4R^2$, we can take $c = 8R^2$, which gets us the bound

$$\mathrm{Loss}_{\mathrm{WAA}}(S) \le \mathrm{Loss}^{\mathrm{avg}}_{\boldsymbol{u}}(S) + 8R^2 \ln n$$

for any $\boldsymbol{u}$.

Since the square loss in the multi-dimensional case is simply the sum of square losses on individual components, we could try handling the k-dimensional case simply by running k copies of the Weighted Average Algorithm and predicting each component independently of each other. Let us denote the resulting algorithm by WAA(k) and compare this approach to the one analyzed above. It is easy to see that if we allow the experts' predictions and outcomes in the one-dimensional case to range over $[-B, B]$ instead of $[0, 1]$, we must for square loss replace the constant $\widetilde{c}_L = 2$ by $\widetilde{c}_L(2B)^2 = 8B^2$. The bound we get is then

$$\mathrm{Loss}_{\mathrm{WAA}(k)}(S) \le \sum_{j=1}^{k} \left(\min_i \sum_{t=1}^{\ell} (y_{t,j} - (x_{t,i})_j)^2 \right) + 8kB^2 \ln n \ .$$

Comparing this with the bound we have for the true multi-dimensional Weighted Average Algorithm (WAA), we see that the first term in the bound for WAA(k) can be much lower if there are experts that are good for predicting some but not all of the components. This potential for better fit is what WAA(k) gains by having kn instead of n weights. On the other hand, the second term in the bound for WAA(k) is linear in k, which is where WAA(k) loses for having so many weights. (Of course, depending on how the vectors $\boldsymbol{y}_t$ and $\boldsymbol{x}_{t,i}$ are located in R^k, the factor $8R^2$ in the bound for the true multi-dimensional WAA may also grow linearly in k.)

As another example, consider the relative entropy loss $L(\boldsymbol{y}, \boldsymbol{x}) = \sum_{j=1}^{k} y_j \ln(y_j/x_j)$, where we assume that $\boldsymbol{y}$ and $\boldsymbol{x}$ are in the probability simplex: $y_i, x_i \ge 0$ and $\sum_j x_j = \sum_j y_j = 1$. Then

$$\frac{\partial L_{\boldsymbol{y}}(\boldsymbol{x})}{\partial x_i} = -\frac{y_i}{x_i}$$

and

$$\frac{\partial^2 L_{\boldsymbol{y}}(\boldsymbol{x})}{\partial x_i \partial x_j} = \delta_{ij} \frac{y_i}{x_i^2} \ ,$$

where $\delta_{ij} = 1$ for $i = j$ and $\delta_{ij} = 0$ otherwise. Now, given $\boldsymbol{y}$, $\boldsymbol{x}$ and a vector $\boldsymbol{z} \in \mathbf{R}^k$, let Q be a random variable that for $i = 1, \ldots, k$ takes the value $q_i = z_i/x_i$ with probability y_i. We can then write

$$\boldsymbol{z} \cdot \nabla L_{\boldsymbol{y}}(\boldsymbol{x}) = -\sum_{j=1}^{k} z_i \frac{y_i}{x_i} = -\sum_{j=1}^{k} y_i q_i = -\mathrm{E}[Q] \ ,$$

and similarly

$$\boldsymbol{z}^{\mathrm{T}} \mathrm{D}^2 L_{\boldsymbol{y}}(\boldsymbol{x}) \boldsymbol{z} = \sum_{j=1}^{k} z_i^2 \frac{y_i}{x_i^2} = \sum_{j=1}^{k} y_i q_i^2 = \mathrm{E}[Q^2] \ .$$

Thus, we have

$$\frac{(\boldsymbol{z} \cdot \nabla L_{\boldsymbol{y}}(\boldsymbol{x}))^2}{\boldsymbol{z}^{\mathrm{T}} \mathrm{D}^2 L_{\boldsymbol{y}}(\boldsymbol{x}) \boldsymbol{z}} = \frac{E[Q]^2}{\mathrm{E}[Q^2]} \le 1$$

by the usual properties of random variables. Hence, for relative entropy loss we have

$$\mathrm{Loss}_{\mathrm{WAA}}(S) \le \left(\min_i \mathrm{Loss}_{\mathcal{E}_i}(S)\right) + \ln n$$

even in the multi-dimensional case.

A Old-style proof for continuous-valued outcomes

We use the notations and concepts of the earlier parts of this paper. Our goal is to provide a sufficient condition for the constant c such that the key inequality (10) holds. The main idea is to obtain something like the old proof [HKW98] that gives the tighter constant c_L, yet is more general in that it holds for continuous-valued outcomes $y_t \in [0,1]$ without the additional assumptions used in the earlier work.

Thus, we are set to prove $L(y_t, \widehat{y}_t) \le P_t - P_{t+1}$ for the potential defined in (11). Let us now define

$$\Delta_t(y) = -c \ln \sum_{i=1}^{n} v_{t,i} \exp(-L(y, x_{t,i})/c) \ . \tag{18}$$

That is, $\Delta_t(y)$ is the potential drop that would occur if the tth outcome were y. The key inequality then becomes $L(y_t, \widehat{y}_t) \le \Delta_t(y_t)$. Since the learner must choose its prediction $\widehat{y}_t$ so that the key inequality holds for all possible outcomes y_t, the condition for the prediction is that

$$L(y, \widehat{y}_t) \le \Delta_t(y) \tag{19}$$

holds for all $y \in [0,1]$. Since we assume $L(y, \widehat{y})$ to be continuous, and decreasing in $\widehat{y}$ for $\widehat{y} < y$ and increasing for $\widehat{y} > y$, for each y there is a continuous range of values $\widehat{y}_t$ that would satisfy (19) for that y. More specifically, let us define

$$A_t(y) = \min \{\, \widehat{y} \in [0,1] \mid L(y, \widehat{y}) \le \Delta_t(y) \,\}$$

and

$$B_t(y) = \max \{\, \widehat{y} \in [0,1] \mid L(y, \widehat{y}) \le \Delta_t(y) \,\} \ .$$

Notice that $\Delta_t(y)$ is always nonnegative, so since $L(y,y) = 0$ and L is continuous, $A_t(y)$ and $B_t(y)$ are always well-defined and $A_t(y) \le y \le B_t(y)$. Now (19) becomes

$$A_t(y) \le \widehat{y}_t \le B_t(y) \ .$$

For an acceptable $\widehat{y}_t$ to exist, the condition is then that

$$\bigcap_{y \in [0,1]} [A_t(y), B_t(y)] \ne \emptyset$$

or, equivalently, that

$$\max_{y \in [0,1]} A_t(y) \le \min_{y \in [0,1]} B_t(y) \ . \tag{20}$$

We now go on to prove that $A_t(q) \leq B_t(p)$ holds for all possible outcomes $p, q \in [0,1]$. Thus, fix arbitrary values $p, q \in [0,1]$. (To prove (20), we could just take $p = \text{argmax}_y A_t(y)$ and $q = \text{argmin}_y B_t(y)$, but this would not simplify the proof.) First we make some observations that simplify technical details. Since always $A_t(y) \leq y \leq B_t(y)$, we can without loss of generality assume $p < q$. If we now have $L(p,0) > L(p,1)$, we get $L(q,0) > L(p,0) > L(p,1) > L(q,1)$. Therefore, we may assume that either $L(p,0) \leq L(p,1)$ or $L(q,1) \leq L(q,0)$. We do the proof assuming $L(p,0) \leq L(p,1)$; the second case is similar.

Our proof for $A_t(q) \leq B_t(p)$ is based on considering the connection between $L(p,z)$ and $L(q,z)$ for $0 < z < 1$. In general, knowing that $L(q,z) = a$ is not enough to uniquely determine $L(q,z)$, since there can be two values $z_1 < q < z_2$ such that $L(q,z_1) = L(q,z_2) = a$ but $L(q,z_1) \neq L(q,z_2)$. However, for our purposes it is sufficient to obtain a mapping that connects $L(p,z)$ and $L(q,z)$ for z in a suitably restricted range. Hence, for $z \in [p,1]$ we define $G(z) = L(p,z)$. Since G is continuous and strictly increasing in its domain $[p,1]$, it has a strictly increasing and continuous inverse G^{-1} in the range of G, which is $[0, L(p,1)]$. Notice that by our assumption $L(p,0) \leq L(p,1)$, the value $L(p,z)$ is in the range of G also for $0 \leq z < p$. Notice also that if we have $\Delta_t(p) \geq L(p,1)$, then $B_t(p) = 1$ and our claim $A_t(q) \leq B_t(p)$ clearly holds. Hence, we assume without loss of generality that $\Delta_t(p) < L(p,1)$, so $\Delta_t(p)$ is in the range of G.

For $0 \leq z \leq 1$, define $\alpha(z) = \exp(-L(p,z)/c)$ and $\gamma(z) = \exp(-L(q,z)/c)$. We get a function f such that $f(\alpha(z)) = \gamma(z)$ for $p \leq z \leq 1$ by defining

$$f(r) = \exp(-L(q, G^{-1}(-c \ln r))/c)$$

for $\exp(-L(p,1)/c) \leq r \leq 1$. Our proof for $A_t(q) \leq B_t(p)$ consist of proving two claims.

Claim 1 If the function f is concave in $[p,1]$ then $A_t(q) \leq B_t(p)$.

Claim 2 If $c \geq R(z,p,q)$ for $0 < z < 1$ where

$$R(z,p,q) = \frac{L'_p(z)L'_q(z)^2 - L'_q(z)L'_p(z)^2}{L'_p(z)L''_q(z) - L'_q(z)L''_p(z)} \ ,$$

then the function f is concave in $[p,1]$.

Hence, since c_L is an upper bound for $R(z,p,q)$, we see that $A_t(q) \leq B_t(p)$ holds for $c \geq c_L$.

To prove Claim 1, assume now that f is concave, that is $f''(\alpha(z)) \leq 0$ holds for $p < z < 1$. Define $x'_{t,i} = G^{-1}(L(p, x_{t,i}))$. Thus $x'_{t,i} = x_{t,i}$ for $p \leq x_{t,i}$, and for $0 \leq x_{t,i} < p$ we still have $L(p, x'_{t,i}) = L(p, x_{t,i})$ and $L(q, x'_{t,i}) \leq L(q, x_{t,i})$. Since $\gamma(z)$ increases as $L(q,z)$ decreases, we have

$$\Delta_t(q) = -c \ln \sum_{i=1}^{n} v_{t,i} \gamma(x_{t,i}) \geq -c \ln \sum_{i=1}^{n} v_{t,i} \gamma(x'_{t,i}) \ .$$

Applying concavity of f we now get

$$\begin{aligned}
\Delta_t(q) &\geq -c \ln \sum_{i=1}^{n} v_{t,i} \gamma(x'_{t,i}) \\
&= -c \ln \sum_{i=1}^{n} v_{t,i} f(\alpha(x'_{t,i})) \\
&= -c \ln \sum_{i=1}^{n} v_{t,i} f(\alpha(x_{t,i})) \\
&\geq -c \ln f\left(\sum_{i=1}^{n} v_{t,i} \alpha(x_{t,i}) \right) \\
&= L\left(q, G^{-1}\left(-c \ln \sum_{i=1}^{n} v_{t,i} \alpha(x_{t,i}) \right) \right) \\
&= L(q, G^{-1}(\Delta_t(p))) \ .
\end{aligned}$$

Hence, we have $G^{-1}(\Delta_t(p)) \geq A_t(q)$, and since G is strictly increasing this is equivalent with $\Delta_t(p) \geq G(A_t(q)) = L(p, A_t(q))$. Therefore, $A_t(q) \leq B_t(p)$, which was our claim.

We now prove Claim 2. Consider $z \in [p, 1]$. We have $f(\alpha(z)) = \gamma(z)$ and thus $f'(\alpha(z)) = \gamma'(z)/\alpha'(z)$. Differentiating further, we obtain $f''(\alpha(z))\alpha'(z) = (\gamma''(z)\alpha'(z) - \gamma'(z)\alpha''(z))/\alpha'(z)^2$. Since $\alpha'(z) = -L'_p(z)\alpha(z)/c < 0$, we have $f''(\alpha(z)) \leq 0$ if and only if $\gamma''(z)\alpha'(z) - \gamma'(z)\alpha''(z) \geq 0$. By substituting $\alpha'(z) = -L'_p(z)\alpha(z)/c$ and $\alpha''(z) = (-L''_p(z)/c + (L'_p(z))^2/c^2)\alpha(z)$, and using similar expressions for $\gamma'(z)$ and $\gamma''(z)$, we see that $f''(\alpha(z)) \leq 0$ if and only if

$$\left(-L'_p(z)L'_q(z)^2 + L'_q(z)L'_p(z)^2 + c\left(L'_p(z)L''_q(z) - L'_q(z)L''_p(z)\right)\right) \frac{\alpha(z)\gamma(z)}{c^3} \geq 0.$$

Finally, since our assumptions imply $L'_p(z)L''_q(z) - L'_q(z)L''_p(z) > 0$, we conclude that $f''(\alpha(z)) \leq 0$ holds if $c \geq R(z, p, q)$.

References

[CBFH+97] N. Cesa-Bianchi, Y. Freund, D. Haussler, D. P. Helmbold, R. E. Schapire, and M. K. Warmuth. How to use expert advice. *Journal of the ACM*, 44(3):427–485, 1997.

[FS97] 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(1):119–139, August 1997.

[FSSW97] Y. Freund, R. E. Schapire, Y. Singer, and M. K. Warmuth. Using and combining predictors that specialize. In *Proc. 29th ACM Symposium on Theory of Computing*, pages 334–343. ACM, 1997.

[HKW95] D. P. Helmbold, J. Kivinen, and M. K. Warmuth. Worst-case loss bounds for sigmoided linear neurons. In *Proc. 1995 Neural Information Processing Conference*, pages 309–315. MIT Press, Cambridge, MA, November 1995.

[HKW98] D. Haussler, J. Kivinen, and M. K. Warmuth. Sequential prediction of individual sequences under general loss functions. *IEEE Transactions on Information Theory*, 44(5):1906–1925, September 1998.

[KW97] J. Kivinen and M. K. Warmuth. Additive versus exponentiated gradient updates for linear prediction. *Information and Computation*, 132(1):1–64, January 1997.

[LW94] N. Littlestone and M. K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212–261, 1994.

[Vov90] V. Vovk. Aggregating strategies. In *Proc. 3rd Annu. Workshop on Comput. Learning Theory*, pages 371–383. Morgan Kaufmann, 1990.