Competitive Online Generalized Linear Regression under Square Loss

Fedor Zhdanov and Vladimir Vovk

Computer Learning Research Centre and Department of Computer Science,
Royal Holloway, University of London, Egham, Surrey TW20 0EX, England
{fedor,vovk}@cs.rhul.ac.uk

Abstract. We apply the Aggregating Algorithm to the problem of online regression under the square loss function. We develop an algorithm competitive with the benchmark class of generalized linear models (our "experts"), which are used in a wide range of practical tasks. This problem does not appear to be analytically tractable. Therefore, we develop a prediction algorithm using the Markov chain Monte Carlo method, which is shown to be fast and reliable in many cases. We prove upper bounds on the cumulative square loss of the algorithm. We also perform experiments with our algorithm on a toy data set and two real world ozone level data sets and give suggestions about choosing its parameters.

Keywords: online prediction, Aggregating Algorithm, square loss function, generalized linear models.

1 Introduction

Generalized linear models have been found very useful in statistical analysis (see [14]) for solving bounded regression problems. Classification problems as well are often solved by means of these models.

We use the Aggregating Algorithm in a way which is somewhat similar to the Aggregating Algorithm for Regression (AAR) introduced in [18]. Whereas the AAR only covers the class of linear experts, our new algorithm also covers other popular classes of experts, which are more efficient in that their predictions always belong to the interval $[Y_1, Y_2]$ assumed to contain the label that is being predicted. When specialized to the case of linear experts, our general loss bound coincides with the known optimal bound for the AAR. A disadvantage of our algorithm is that we need to know $[Y_1, Y_2]$ *a priori* to be able to apply it.

Another popular field related to competitive prediction is online convex optimization introduced in [20]. The two fields cover a common special case: a compact set of experts under loss functions of a specific form (the square loss for our application). The problems which cover in part generalized linear models are analyzed in [10]. In the general case, online convex optimization significantly relaxes the condition on loss functions, whereas competitive prediction removes the compactness requirement. Since many algorithms for competitive prediction

J.L. Balcázar et al. (Eds.): ECML PKDD 2010, Part III, LNAI 6323, pp. 531–546, 2010.

are designed for a narrower class of loss functions, they have a better coefficient in front of log in the regret term. Our algorithm can be applied to some non-convex functions, which is impossible for the methods of online convex optimization even for a compact set of experts.

Generalized linear models are popular in Bayesian statistics for solving classification problems (see Relevance Vector Machines in Section 7.2.3 of [4]). From the competitive prediction perspective, Bayesian mixtures are analogous to the Aggregating Algorithm competing under the logarithmic loss function. Upper bounds on the logarithmic loss are proved in [8], [12], [13], and [2] using different approaches. In this paper we prove upper bounds on the square loss, which is more often used in practice to compare the performances of different algorithms.

Our prediction problem does not appear to be analytically tractable. Therefore, we develop a prediction algorithm using the Markov chain Monte Carlo method, which is shown to be fast and reliable in many cases. Monte Carlo methods are well known in the Bayesian community [16]. They are also applied, for example, in [7] to explore exponential weighting schemes in problems with high dimension of the input vectors.

We give suggestions about choosing the parameters of our algorithm and perform experiments with it on a toy data set and two real world ozone level data sets.

Our paper is organized as follows. Section 2 describes the prediction protocol and the main theorem, and Section 3 covers a set of important examples. In Section 4 we derive the algorithm competitive with generalized linear models. Section 5 describes experiments with the new algorithm. In Section 6 we prove the main theorem.

2 Prediction Algorithm and Loss Bound

This is our prediction protocol:

Protocol 1. Online bounded regression protocol

for $t = 1, 2, \ldots$ **do**
  Reality announces $x_t \in \mathbb{R}^n$
  Learner predicts $\gamma_t \in \mathbb{R}$
  Reality announces $y_t \in [Y_1, Y_2]$
end for

Protocol 1 describes a perfect-information game between two players, Learner (male) and Reality (female). At each step t Reality announces a vector $x_t \in \mathbb{R}^n$, which is, intuitively, a hint that she gives to Learner to help him predict her next move y_t, called the *outcome*. The vector x_t will be called the *object*, or *input vector*, and its components $x_{t,i}, i = 1, \ldots, n$, will be called *attributes*. After that Learner announces his prediction γ_t for y_t and Reality announces the outcome $y_t \in [Y_1, Y_2]$. Sometimes we will say that y_t is the *label* of the object x_t. Learner's loss is measured by the square loss function. Learner wishes to compete with the

class of *generalized linear experts* indexed by $\theta \in \Theta = \mathbb{R}^n$. Expert θ's prediction at step t is denoted ξ_t^θ and is equal to

$$\xi_t^\theta = Y_1 + (Y_2 - Y_1)\sigma(\theta' x_t). \tag{1}$$

Here $\sigma : \mathbb{R} \to \mathbb{R}$ is a fixed *activation function*. We have $\sigma : \mathbb{R} \to [0,1]$ in all the cases except linear regression (see below). If the range of the function σ is $[0,1]$, the experts necessarily give predictions from $[Y_1, Y_2]$.

Learner records the cumulative losses suffered by himself, $L_T = \sum_{t=1}^T (\gamma_t - y_t)^2$, and each expert θ, $L_T^\theta = \sum_{t=1}^T (\xi_t^\theta - y_t)^2$, over the first T steps. Intuitively, Learner's goal is to ensure that $L_T \le L_T^\theta$, or at least $L_T \approx L_T^\theta$, for a vast majority of experts $\theta \in \Theta$.

Assume that the function

$$b(u,z) := \left(\frac{d\sigma(z)}{dz}\right)^2 + (\sigma(z) - u)\frac{d^2\sigma(z)}{dz^2} \tag{2}$$

is uniformly bounded: $b := \sup_{u\in[0,1], z\in\mathbb{R}} |b(u,z)| < \infty$. We will further see that this assumption holds for the most popular generalized linear regression models. We prove the following upper bound on Learner's loss.

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

$$L_T \le L_T^\theta + a\|\theta\|^2 + \frac{(Y_2 - Y_1)^2}{4} \ln\det\left(I + \frac{b(Y_2 - Y_1)^2}{a}\sum_{t=1}^T x_t x_t'\right), \tag{3}$$

where $b := \sup_{u\in[0,1], z\in\mathbb{R}} |b(u,z)|$ and $b(u,z)$ is defined by (2). If in addition $\|x_t\|_\infty \le X$ for all t then

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

The order of the regret term in bound (4) is logarithmic in the number of steps. It matches the order of the best bounds for the linear regression problem under square loss proved in [18] and for the classification problem using generalized linear regression experts under logarithmic loss proved in [12].

The prediction strategy achieving (3) is formulated as Algorithm 1, calculated with the number of iterations $M \to \infty$; we also call Algorithm 1 the Aggregating Algorithm for Generalized Linear Models (AAGLM). In Section 4 we derive it and describe its parameters. Even though Algorithm 1 is an online algorithm, it is easy to apply it in the batch setting: it suffices to consider each example in the test set as the next example after the training set.

3 Examples of the Models and Performance Guarantees

Now we give some examples of generalized linear models and bounds on the losses of the corresponding algorithms.

3.1 Linear Regression

We have for linear regression

$$\sigma(z) = \frac{z - Y_1}{Y_2 - Y_1}, \quad z \in \mathbb{R} \tag{5}$$

(so that the experts predict $\xi_t^\theta = \theta' x_t$). The derivatives are equal to

$$\frac{d\sigma(z)}{dz} = \frac{1}{Y_2 - Y_1}$$

and

$$\frac{d^2\sigma(z)}{dz^2} = 0.$$

Using these expressions in the derivatives of σ in (2) we obtain

$$b(u, z) = \frac{1}{(Y_2 - Y_1)^2}.$$

Using $b = \frac{1}{(Y_2 - Y_1)^2}$ in (4) we have the following corollary for the linear regression experts.

Corollary 1. *There exists a prediction strategy for Learner achieving*

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

for any expert $\theta \in \mathbb{R}^n$ predicting according to (1) *and* (5) *(i.e., $\xi_t^\theta = \theta' x_t$).*

As we can see, the upper bound is the same as the upper bound for the AAR with $Y_2 = Y$, $Y_1 = -Y$ (see Theorem 1 in [18]). This bound is known to have the best possible order (see Theorem 2 in [18]) for the linear experts.

3.2 Logistic Regression

We have for logistic regression

$$\sigma(z) = \frac{1}{1 + e^{-z}}, \quad z \in \mathbb{R}. \tag{6}$$

The derivatives are equal to

$$\frac{d\sigma(z)}{dz} = \sigma(z)(1 - \sigma(z))$$

and

$$\frac{d^2\sigma(z)}{dz^2} = \sigma(z)(1 - \sigma(z))(1 - 2\sigma(z)).$$

Using these expressions in the derivatives of σ in (2) we obtain

$$b(u, z) = \sigma^2(z)(1 - \sigma(z))^2 + (\sigma(z) - u)\sigma(z)(1 - \sigma(z))(1 - 2\sigma(z)).$$

We have $|b(u, z)| \le b \le \frac{5}{64}$. Using this in (4) we have the following corollary for the logistic regression experts.

Corollary 2. *Let $a > 0$. There exists a prediction strategy for Learner achieving*

$$L_T \le L_T^\theta + a\|\theta\|^2 + \frac{n(Y_2 - Y_1)^2}{4} \ln\left(1 + \frac{5(Y_2 - Y_1)^2 X^2 T}{64a}\right)$$

for any expert $\theta \in \mathbb{R}^n$ predicting according to (1) *and* (6).

3.3 Probit Regression

We have for probit regression

$$\sigma(z) = \Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-v^2/2} dv, \quad z \in \mathbb{R}, \tag{7}$$

where Φ is the cumulative distribution function of the normal distribution with zero mean and unit variance. The derivatives are equal to

$$\frac{d\sigma(z)}{dz} = \frac{1}{\sqrt{2\pi}} e^{-z^2/2}$$

and

$$\frac{d^2\sigma(z)}{dz^2} = -\frac{z}{\sqrt{2\pi}} e^{-z^2/2}.$$

Using these expressions in the derivatives of σ in (2) we obtain

$$b(u, z) = \frac{1}{2\pi} e^{-z^2} - (\sigma(z) - u) \frac{z}{\sqrt{2\pi}} e^{-z^2/2}.$$

We have $|b(u, z)| \le b \le \frac{25}{128}$. Using this in (4) we have the following corollary for the probit regression experts.

Corollary 3. *Let $a > 0$. There exists a prediction strategy for Learner achieving*

$$L_T \le L_T^\theta + a\|\theta\|^2 + \frac{n(Y_2 - Y_1)^2}{4} \ln\left(1 + \frac{25(Y_2 - Y_1)^2 X^2 T}{128a}\right)$$

for any expert $\theta \in \mathbb{R}^n$ predicting according to (1) *and* (7).

3.4 Complementary Log-Log Regression

We have for the complementary log-log regression

$$\sigma(z) = 1 - \exp(-\exp(z)), \quad z \in \mathbb{R}. \tag{8}$$

When the argument z of the complementary log-log function $1 - \exp(-\exp(z))$ approaches minus infinity, this function is similar to the logistic function $\frac{1}{1+e^{-z}}$ and tends to zero. When the argument approaches infinity, the function tends to

one more quickly than the logistic function. This can be used in problems with asymmetry in outcomes. The derivatives of $\sigma(z)$ are equal to

$$\frac{d\sigma(z)}{dz} = e^z(1-\sigma(z))$$

and

$$\frac{d^2\sigma(z)}{dz^2} = e^z(1-\sigma(z))(1-e^z).$$

Using these expressions in the derivatives of σ in (2) we obtain

$$b(u,z) = e^{2z}(1-\sigma(z))^2 + e^z(1-\sigma(z))(1-e^z).$$

We have $|b(u,z)| \le b \le \frac{17}{64}$. Using this in (4) we have the following corollary for the complementary log-log regression experts.

Corollary 4. *Let $a > 0$. There exists a prediction strategy for Learner achieving*

$$L_T \le L_T^\theta + a\|\theta\|^2 + \frac{n(Y_2-Y_1)^2}{4} \ln\left(1 + \frac{17(Y_2-Y_1)^2X^2T}{64a}\right)$$

for any expert $\theta \in \mathbb{R}^n$ predicting according to (1) *and* (8).

4 Derivation of the Prediction Algorithm

Our prediction algorithm differs from many algorithms commonly used to fit a generalized linear model (see, for example, [14]). First, instead of fitting the data with the best parameter θ (as in logistic regression) it uses the regularization parameter $a > 0$ preventing θ to be too large, and thus preventing overfitting to a certain extent. Second, it tries to minimize the square loss instead of the logarithmic loss. To predict at step T it works with the function

$$\sum_{t=1}^{T-1} (\xi_t^\theta - y_t)^2 + a\|\theta\|^2 \tag{9}$$

with ξ_t^θ from (1). Third, it does not look for the best regularized expert θ, but at each prediction step it mixes all the experts in a way which is similar to Bayesian scheme, thus preventing overfitting even further.

We use the Aggregating Algorithm (AA) to derive the prediction algorithm. The AA works as follows.

It is given a parameter η and an initial distribution on the experts. We choose the normal distribution

$$P_0(d\theta) = (a\eta/\pi)^{n/2} \exp(-a\eta\|\theta\|^2)d\theta \tag{10}$$

for some $a > 0$, the other parameter of the algorithm. After each step t the AA updates the weights of the experts according to their losses:

$$P_t(d\theta) = e^{-\eta(\xi_t^\theta - y_t)^2} P_{t-1}(d\theta)$$

and then normalizes the weights: $P_t^*(d\theta) = \frac{P_t(d\theta)}{\int_\Theta P_t(d\theta)}$. It is easy to see that at the step t the unnormalized weight of an expert θ can be expressed as

$$P_t(d\theta) = (a\eta/\pi)^{n/2} e^{-\eta(L_t^\theta + a\|\theta\|^2)} d\theta. \tag{11}$$

To give a prediction the AA first calculates the *generalized prediction*

$$g_t(y) = -\frac{1}{\eta} \ln \int_\Theta e^{-\eta(\xi_t^\theta - y)^2} P_{t-1}^*(d\theta) \tag{12}$$

for any possible outcome $y \in [Y_1, Y_2]$ using the normalized weights computed on the previous step. It then gives its prediction $\gamma_t = \Sigma(g)$ using a *substitution function* Σ such that $(\gamma_t - y)^2 \le g_t(y)$ for any $y \in [Y_1, Y_2]$. It is known [18] that for the square loss game such a substitution function exists if $\eta \in \left(0, \frac{2}{(Y_2 - Y_1)^2}\right]$. The following substitution function outputs a permitted prediction even if the generalized predictions are calculated from the weights scaled by any constant (for example, unnormalized):

$$\gamma_t = \frac{1}{2}\left(Y_2 + Y_1 - \frac{G_t(Y_2) - G_t(Y_1)}{Y_2 - Y_1}\right) \tag{13}$$

for

$$G_t(y) = -\frac{1}{\eta} \ln \int_\Theta e^{-\eta((\xi_t^\theta - y)^2 + L_{t-1}^\theta + a\|\theta\|^2)} d\theta. \tag{14}$$

We obtain (14) by calculating the generalized prediction (12) with unnormalized weights (11) and omitting the factor $(a\eta/\pi)^{n/2}$. Normalization is avoided because calculating the normalizing constant is a computationally inefficient operation. We will use the maximum value for η, $\eta = \frac{2}{(Y_2 - Y_1)^2}$. We denote

$$w_T(\theta) := \exp\left(-a\eta\|\theta\|^2 - \eta \sum_{t=1}^{T} \left(\xi_t^\theta - y_t\right)^2\right). \tag{15}$$

Algorithm 1 is based on the MCMC technique of numerical integration in (14) at $y = Y_1$ and $y = Y_2$; a good MCMC survey is given in [1]. The function $e^{-\eta(\xi_t^\theta - y)^2}$ needs to be integrated with respect to the unnormalized posterior distribution $P_{t-1}(d\theta)$. We choose to use the simple Metropolis sampling to sample θ from the posterior.

Metropolis sampling from a distribution $\mathcal{P}$ is an iterative process, with the initial value θ^0 and a simple proposal distribution. We choose the Gaussian proposal distribution $N(0, \sigma^2)$ with the parameter σ^2 chosen on the data. At each iteration $i = 1, \ldots, M$, the update for θ is sampled from the proposed distribution:

$$\theta^i = \theta^{i-1} + \zeta^i, \;\; \zeta^i \sim N(0, \sigma^2).$$

The updated θ^i is accepted with the probability $\min\left(1, \frac{f_\mathcal{P}(\theta^i)}{f_\mathcal{P}(\theta^{i-1})}\right)$. Here $f_\mathcal{P}(\theta)$ is the value of the density function for the distribution $\mathcal{P}$ at the point θ. By

accepting and rejecting the updates the values of the parameter θ move closer to the value where the maximum of the density function $f_{\mathcal{P}}$ is achieved. Thus sampling is performed from the area with high density of $\mathcal{P}$ and covers the tails of it only occasionally. Therefore numerical integration using sampling with MCMC is often more efficient than the usual Monte Carlo sampling from the uniform distribution. Sometimes the updates are accepted even if they do not move the next θ closer to the maximum (this happens when $\frac{f_{\mathcal{P}}(\theta^i)}{f_{\mathcal{P}}(\theta^{t-1})} < 1$). This may allow the algorithm to move away from the local maxima of the density function.

Algorithm 1. AAGLM

Require: Bounds Y_1, Y_2 for the outcomes,
maximum number $M > 0$ of MCMC iterations,
standard deviation $\sigma > 0$,
regularization coefficient $a > 0$

calculate $\eta := \frac{2}{(Y_2 - Y_1)^2}$
initialize $\theta_0^M := 0 \in \Theta$
define $w_0(\theta) := \exp(-a\eta \|\theta\|^2)$
for $t = 1, 2, \ldots$ **do**
  $G_{t,1} := 0$, $G_{t,2} := 0$
  read $x_t \in \mathbb{R}^n$
  initialize $\theta_t^0 := \theta_{t-1}^M$
  for $m = 1, 2, \ldots, M$ **do**
    $\theta^* := \theta_t^{m-1} + N(0, \sigma^2 I)$
    if $w_{t-1}(\theta^*) \geq w_{t-1}(\theta_t^{m-1})$ **then**
      $\theta_t^m := \theta^*$
    else
      flip a coin with success probability $w_{t-1}(\theta^*)/w_{t-1}(\theta_t^{m-1})$
      if success **then**
        $\theta_t^m := \theta^*$
      else
        $\theta_t^m := \theta_t^{m-1}$
      end if
    end if
    $G_{t,1} := G_{t,1} + e^{-\eta(\xi_t^{\theta_t^m} - Y_1)^2}$, $G_{t,2} := G_{t,2} + e^{-\eta(\xi_t^{\theta_t^m} - Y_2)^2}$
  end for
  output prediction $\gamma_t := \frac{1}{2}\left(Y_2 + Y_1 + \frac{\ln G_{t,2} - \ln G_{t,1}}{\eta(Y_2 - Y_1)}\right)$
  read $y_t \in [Y_1, Y_2]$
  define $w_t(\theta) := w_{t-1}(\theta) \exp(-\eta(\xi_t^\theta - y_t)^2)$
end for

It is common when using the MCMC approach to have a "burn-in" stage, at which the integral is not calculated, but the algorithm is looking for the best "locality" for θ. This stage is used to avoid the error accumulated while the algorithm is still looking for the correct location of the main mass of the distribution. Instead of that, at each prediction step t we take the new starting

point θ^0 for the Metropolis sampling to be the last point θ^M achieved on the previous step $t-1$.

In the case when the dimension n of input vectors is large and the sampling is not very efficient, one can use more advanced techniques, such as adaptive sampling or Slice sampling [16]. However, when we tried several versions of Slice sampling, the convergence speed on our data sets was slower than for Metropolis sampling.

The function (9) is not necessarily convex in θ, so it may have several local minima. Thus we cannot use the Laplace approximation for the integral and obtain reliable Iteratively Reweighted Least Square estimation of θ, the common approach to give predictions when working with generalized linear models. The MCMC approach to calculating similar integrals for Bayesian prediction models was analyzed in [15]. It is stated there that it takes $O(n^3)$ operations to calculate a general integral.

5 Experiments

In this section we investigate empirical properties of our algorithm on toy and real data sets and suggest ways to choose the parameters for it.

5.1 Toy Data Set

In this experiment we aim to emphasize the main properties of competitive online algorithms: how they behave if the data follow the assumptions of one of the experts, and when the data fail to do so; how quickly online algorithms adjust to the substantial changes in the properties of the data.

Consider the following online classification problem. Let $x_t \in \mathbb{R}$ be the input vectors $x_1 = -50, x_2 = -49.9, \ldots, x_T = 100$, real numbers from -50 to 100 with step 0.1. Let the outcomes y_t be

$$y_t = \begin{cases} 1 & \text{if } x_t < -10 \text{ or } 10 < x_t < 50, \\ 0 & \text{otherwise.} \end{cases}$$

We add the bias one as the second component of each input vector (input vector dimension becomes equal to 2). We try to predict this sequence online step by step by Algorithm 1 competing with logistic experts. The result is presented on Figure 1. We also show the best fitted logistic predictor achieving the minimum cumulative square loss. Notice three following interesting properties of the picture.

The predictions of the AAGLM tend to the predictions of the best logistic regression fitted to the whole data as T becomes large. This matches the fact that the mean loss of the AAGLM converges to the mean loss of the best logistic regression; see Corollary 2.

When $x \in [-10, 50)$ both algorithms suffer large loss. Corollary 2 ensures that the AAGLM does not suffer much more than the best logistic regression.

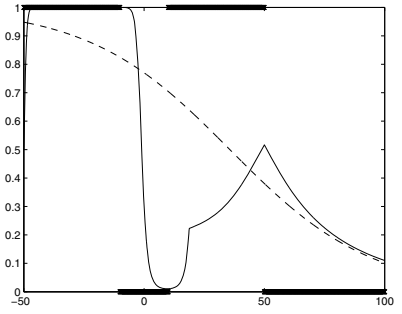

Fig. 1. Sequential predictions of the AAGLM algorithm for the two-class classification problem. The dashed curve is the predictions of the best logistic regression (under square loss) on all the data. The horizontal axis contains the input vectors, the vertical axis contains the outcomes and the predictions of the outcomes by the two algorithms.

During each period the AAGLM tries to fit a logistic regression function in this and the previous periods. The dependence on the previous periods is due to the fact that the trained AAGLM is equivalent to the untrained AAGLM which starts predicting with initial weights distribution $P_t^*(d\theta)$, where t is the number of steps in the previous periods.

In order for expert-based algorithms to predict well on a certain type of data, the best expert should suffer small loss on these data. If the sequence of outcomes has several regimes which rarely switch from one to another, like in our figure, "tracking the best expert" [11,17] may be a more suitable framework.

The parameters of the AAGLM used in these studies are $Y_1 = 0, Y_2 = 1$, $M = 1000$ (we did not use "burn-in" stage), $\sigma = 0.00001$, $a = 10^{-100}$. Increasing the regularization coefficient a leads to the regularization towards 0.5 (as expected). Increasing M accelerates somewhat the reaction in the very beginning of each turn, but the main trend does not change. Too low value for σ leads to slower convergence. Too high value of this parameter forces the algorithm to fluctuate between two classes, and never find a stable solution (this is expected as well since with large σ the numerical integration becomes less precise). It is common when applying the MCMC technique to use the following rule of thumb to determine the value of the parameter σ: the rejection rate should be 30–70%.

5.2 Ozone Data Set

We perform empirical studies on two real-world data sets described in [19].

Data sets. The data sets contain various meteorology and ozone data for the Houston, Galveston, and Brazoria (HGB) area in Texas, USA, day by day for 7 years, 1998–2004. We use both one-hour (ozone1) and eight-hour (ozone8) ozone data sets: they contain the the daily maximum of 1 hour (ozone1) ozone concentration and the daily maximum of the average over 8 consecutive hours (ozone8)

ozone concentration. Each observation is one day. Each observation has 72 features of various measures of air pollutant and meteorological information for the target area in the study. Each observation is assigned the label 1 (we say its outcome is equal to 1) if the ozone level exceeds the danger threshold, which is 120 parts per billion (ppb) for ozone1 and 80 ppb for ozone8; otherwise, the observation has the label 0 (outcome is equal to 0). The data are collected online, so online prediction algorithms are more appropriate for the study than batch algorithms. They are able to predict ozone levels day by day incorporating the information from all the previous days. Therefore we consider online two-class classification problems.

It is shown in [19] that all 72 features may be relevant to the prediction problem, and thus we decided to use all of them to train our algorithms. We replace the missing values of the features by the mean of the available values of them from the first year (we use the first year data as our training set).

The data sets are very skewed: the number of positive examples is very low (73 for ozone1 and 160 for ozone8 out of 2534 observations). It can be expected that for such data sets complementary log-log (comlog) regression performs better than logistic or probit regression. Indeed, the square loss suffered by logistic regression trained on the whole data set is 48.4178 for ozone1 and 96.2846 for ozone8. At the same time, the square loss suffered by comlog regression trained on the whole data set is 46.7815 and 94.8399 respectively. Thus we use the comlog activation function (8) in this experimental study.

Algorithms and results. We normalize all the features to have mean zero and maximum absolute value one over the first year. We also add an additional bias feature 1 to all the examples.

We compare different algorithms in two regimes: the online regime and the (incremental) batch regime. In the online regime the algorithms are retrained as soon as a new observation is obtained. In the batch regime the algorithms are only retrained yearly on all the past data. In [19] this regime is suggested as the most realistic for meteorologists (they did not consider the online regime though).

For the online regime the AAGLM parameters M, σ, a, and the length of the burn-in stage are chosen to suffer the least square loss over the first year. We choose σ from the range $10^{-k}, 5 \cdot 10^{-k}$, $k = 0, 1, \dots, 5$. We choose a from the range $10^{-k}, 5 \cdot 10^{-k}$, $k = -1, 0, 1, \dots, 10$. The best parameters are $M = 2500$, $\sigma = 0.01$, $a = 0.1$, and the length of the burn-in stage is 2000. It is interesting to note that for both data sets the best parameters are the same up to our precision. This may mean that the best parameters for the algorithm depend mostly on the input vectors, and not on the outcomes (because the input vectors are the same for our data sets).

The batch regime of the AAGLM can be understood as just one step of the online algorithm repeated for each new test example. During the training stage the batch algorithm does not calculate the integral but saves the values of θ obtained at each iteration $m = 1, 2, \dots, M$. At the prediction stage the algorithm calculates the integral using the saved values of θ computed on the iterations

between $B < M$ and M, where B is the length of the burn-in stage. We choose the same parameters $\sigma = 0.01$, $a = 0.1$ as for the online version, and $B = 5000$, $M = 15000$ to ensure good convergence. There are no theoretical guarantees for the batch setting.

The first algorithm with which we compare the AAGLM is the online comlog regression minimizing logarithmic loss (standard generalized regression model): at each step t it uses all the previous steps to find the best parameter $\hat{\theta}$ and then gives its prediction $\sigma(\hat{\theta}' x_t)$ according to this parameter. In terms familiar from the online prediction literature, it corresponds to the Follow the Best Expert predictor under the logarithmic loss function, the natural competitor to our algorithm.

In the batch regime in the beginning of each year the best parameter $\tilde{\theta}$ is found on all the previous years. All the predictions in the following year are made using this $\tilde{\theta}$.

We also calculate the performance of the linear Support Vector Machine (SVM) and the SVM with the RBF kernel [6]. The SVM with the RBF kernel showed the best performance on ozone8 in a different framework [19]. In the online regime the SVMs predict only one next outcome at each step and retrain after the actual outcome is announced. The parameter of the kernel and the parameter C are chosen to achieve the least square loss over the first year. Note that in the online regime one does not need the validation set: the training set at each step does not include the next test example at which prediction is made. Thus the risk of overfitting is less than if the testing was done on the training set.

In the batch regime at the beginning of each year the SVMs are retrained on all the previous years. All the predictions in the following year are made using these trained SVMs. The parameter of the kernel and the parameter C are chosen using 5-fold cross-validation: all the data from the previous years are randomly separated into 5 parts. Four parts are used to train the SVM, the fifth part is used for testing: the square loss is calculated over the fifth part. This procedure repeats 5 times with different combinations of the parts. The parameters of the SVMs are chosen to suffer the least average square loss.

Since the number of positive examples is small, it makes sense to compare the precision of the predicting algorithms with the precision of the zero predictor: the predictor which always predicts low ozone concentration.

The minimal square loss which is suffered by the comlog regression model trained on the whole data set is equal to 28.0001 for ozone1 and 75.0001 for ozone8. This loss is unrealistic since we do not know all the observations when start predicting, and included here to specify the limitations of the generalized regression model. The square loss of the online comlog regression over this period is equal to 105.1823 and 186.6147 respectively. The square loss of the AAGLM over this period is equal to 66.2727 and 120.8483 respectively. At the same time the upper bounds on its loss from Corollary 4 have the values 323.9632 for ozone1 and 371.3681 for ozone8. The zero predictor suffers the square loss 73 and 160 respectively.

Table 1. Average MSEs of different algorithms over the last 6 years of ozone1/ozone8

Algorithm	MSE online	MSE batch
AAGLM	8.2159/15.3288	8.2163/15.7552
SVM rbf	9.7607/14.8806	9.4497/15.9679
SVM linear	8.8624/16.1532	8.8500/16.5264
Comlog	14.4578/24.2686	13.8096/27.9474
Zeros	9.6667/21.3333	9.6667/21.3333

Table 1 contains average mean square errors for different algorithms over the last 6 years (2–7) in the data sets.

Figure 2 presents *precision* (the number of correctly identified high ozone days divided by the total number of the predicted high ozone days) and *recall* (the number of correctly identified high ozone days divided by the total number of the actual high ozone days) for different threshold values calculated for the last 6 years of ozone8. It contains information for the four algorithms applied in the online regime. The area under the curve for the AAGLM is larger than the area under the curve for the online comlog regression and under the curve for the linear SVM. This shows the superiority of our algorithm over these competitors for the classification task. We can also see that there is a point on the curve for the online comlog regression where the reduction of the recall does not lead to the increase in the precision. This means that the threshold becomes larger than the predicted probability of many high-ozone days.

As we can see from the figures in Table 1, the algorithms in the online regime perform better than the same algorithms in the batch regime on ozone8 and usually worse on ozone1.

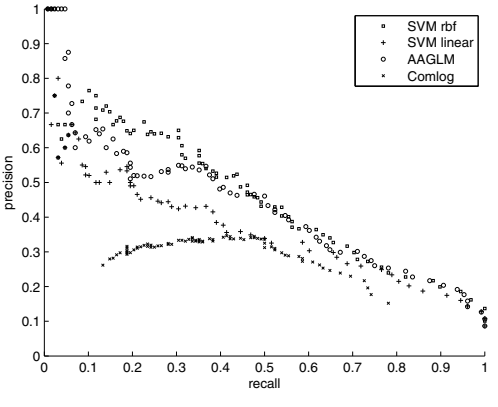

Fig. 2. Precision-recall curve for different threshold values for the algorithms applied in the online regime on ozone8

We can also see that the AAGLM significantly outperforms the simple zero predictor and the comlog predictor. Unlike the comlog predictor, the AAGLM is developed to work with the square loss and achieves better empirical performance in terms of the square loss. It performs a little worse on ozone8 than the SVM with the RBF kernel. It is not difficult to apply kernelization to the AAGLM as well (for the kernelization of standard generalized linear regression models see [5]) using an analogy with non-parametric Bayesian methods; the kernelized algorithm may achieve better performance. On ozone1 the AAGLM outperforms all the algorithms including SVMs.

The disadvantage of our algorithm against the competitors is in its training speed. Increasing the training speed of our algorithm is an interesting area of future research. It would be also interesting to apply our classifier to other data sets and find extreme data sets where the theoretical guarantee is tight.

6 Proof of Theorem 1

Let $\eta := \frac{2}{(Y_2-Y_1)^2}$ and set $\beta = e^{-\eta}$. The loss of the AA over the first T trials does not exceed the loss of the sum of the generalized predictions (12) over the first T trials, which in turn equals (see Lemma 1 in [18])

$$\log_\beta \int_\Theta \beta^{L_T(\theta)} P_0(d\theta) = -\frac{1}{\eta} \ln \left((a\eta/\pi)^{n/2} \int_\Theta e^{-\eta Q(\theta)} d\theta \right), \tag{16}$$

where

$$Q(\theta) := \sum_{t=1}^{T} \left((Y_2 - Y_1)\sigma(\theta' x_t) + Y_1 - y_t \right)^2 + a\|\theta\|^2.$$

We will lower bound the integral in (16) by upper bounding $Q(\theta)$ using a quadratic form.

Because of the second addend in the definition of $Q(\theta)$, $Q(\theta) \to \infty$ as $\|\theta\| \to \infty$. Therefore $\min_\theta Q(\theta)$ is attained at some point. Let θ_0 be the point where it is attained, and thus $\nabla Q(\theta_0) = 0$. We use Taylor expansion of $Q(\theta)$ at the point θ_0:

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

where ϕ is a convex combination of θ_0 and θ. Here H is the Hessian matrix of $Q(\theta)$, the matrix of its second derivatives. By δ_i^j we denote the Kronecker delta. The second partial derivative of Q is expressed as follows:

$$\begin{aligned}
\frac{\partial^2 Q}{\partial\theta_i \partial\theta_j} &= 2a\delta_i^j + 2(Y_2 - Y_1)^2 \\
&\quad \cdot \sum_{t=1}^{T} \left(\frac{\partial\sigma(\theta' x_t)}{\partial\theta_i} \frac{\partial\sigma(\theta' x_t)}{\partial\theta_j} - \left(\frac{y_t - Y_1}{Y_2 - Y_1} - \sigma(\theta' x_t) \right) \frac{\partial^2\sigma(\theta' x_t)}{\partial\theta_i \partial\theta_j} \right) \\
&= 2a\delta_i^j + 2(Y_2 - Y_1)^2 \sum_{t=1}^{T} x_{t,i} x_{t,j} b \left(\frac{y_t - Y_1}{Y_2 - Y_1}, \theta' x_t \right).
\end{aligned}$$

It is clear now that the matrix $H(\phi)$ can be represented as follows:

$$H(\phi) = 2aI + 2(Y_2 - Y_1)^2 X' \Gamma(\phi) X,$$

where X is the design matrix $T \times n$ consisting of the rows of the input vectors $x_1', \ldots, x_T'$. Here $\Gamma(\phi)$ is the diagonal $T \times T$ matrix which has the coefficients $b(u_1, \phi' x_1), \ldots, b(u_T, \phi' x_T)$ on the diagonal (with $u_i = \frac{y_i - Y_1}{Y_2 - Y_1}$, $i = 1, \ldots, T$). Since $\Gamma(\phi)$ is a symmetric matrix, we can see (Theorem 21.5.6 in [9]) that

$$\psi' \Gamma(\phi) \psi \le \psi' \lambda_{\max} \psi \tag{17}$$

for any $\psi \in \mathbb{R}^n$, where $\lambda_{\max}$ is the supremum over maximum eigenvalues of $\Gamma(\phi)$. Since $b(u_t, \phi' x_t)$ is uniformly bounded, we have $\lambda_{\max} \le b$.

We can take $\psi = X(\theta - \theta_0)$ and obtain from (17) that

$$Q(\theta) \le Q(\theta_0) + (\theta - \theta_0)'(aI + b(Y_2 - Y_1)^2 X' X)(\theta - \theta_0).$$

Thus the integral in (16) is lower bounded as follows:

$$\int_\Theta e^{-\eta Q(\theta)} d\theta \ge e^{-\eta Q(\theta_0)} \int_\Theta e^{-\eta(\theta - \theta_0)'(aI + b(Y_2 - Y_1)^2 X' X)(\theta - \theta_0)} d\theta.$$

The integral in the right-hand side can be analytically calculated (see Section 15.12 in [9]):

$$\int_\Theta e^{-\eta(\theta - \theta_0)'(aI + b(Y_2 - Y_1)^2 X' X)(\theta - \theta_0)} d\theta = \frac{(\pi/\eta)^{n/2}}{\sqrt{\det(aI + b(Y_2 - Y_1)^2 X' X)}}.$$

After taking the logarithm of $e^{-\eta Q(\theta_0)}$, we obtain $L_T^{\theta_0} + a\|\theta_0\|^2$. Substituting these expressions and $\eta = \frac{2}{(Y_2 - Y_1)^2}$ to (16) we obtain the upper bound (3).

The determinant of a symmetric positive definite matrix is upper bounded by the product of its diagonal elements (see Chapter 2, Theorem 7 in [3]):

$$\det\left(I + \frac{b(Y_2 - Y_1)^2}{a} \sum_{t=1}^{T} x_t x_t'\right) \le \left(1 + \frac{b(Y_2 - Y_1)^2 T X^2}{a}\right)^n.$$

This concludes the proof.

Acknowledgments

We are grateful for useful comments to Alexey Chernov, Yuri Kalnishkan, and anonymous referees. This work was supported in part by EPSRC (grant EP/F002998/1).

References

1. Andrieu, C., de Freitas, N., Doucet, A., Jordan, M.: An introduction to MCMC for machine learning. Machine Learning 50, 5–43 (2003)
2. Banerjee, A.: An analysis of logistic models: exponential family connections and online performance. In: Proceedings of the 2007 SIAM International Conference on Data Mining, pp. 204–215 (2007)
3. Beckenbach, E.F., Bellman, R.: Inequalities. Springer, Berlin (1961)
4. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, New York (2006)
5. Cawley, G.C., Janacek, G.J., Talbot, N.L.C.: Generalised kernel machines. In: Proceedings of the International Joint Conference on Neural Networks, pp. 1720–1725 (2007)
6. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2001)
7. Dalalyan, A., Tsybakov, A.B.: Sparse regression learning by aggregation and Langevin Monte-Carlo. In: Proceedings of the 22nd Annual Conference on Computational Learning Theory (2009)
8. Forster, J.: On relative loss bounds in generalized linear regression. In: Ciobanu, G., Păun, G. (eds.) FCT 1999. LNCS, vol. 1684, pp. 269–280. Springer, Heidelberg (1999)
9. Harville, D.A.: Matrix Algebra From a Statistician's Perspective. Springer, New York (1997)
10. Hazan, E., Agarwal, A., Kale, S.: Logarithmic regret algorithms for online convex optimization. Machine Learning 69, 169–192 (2007)
11. Herbster, M., Warmuth, M.K.: Tracking the best expert. Machine Learning 32, 151–178 (1998)
12. Kakade, S.M., Ng, A.Y.: Online bounds for Bayesian algorithms. In: Proceedings of the 18th Annual Conference on Neural Information Processing Systems (2004)
13. Kivinen, J., Warmuth, M.K.: Relative loss bounds for multidimensional regression problems. Machine Learning 45, 301–329 (2001)
14. McCullagh, P., Nelder, J.: Generalized Linear Models, 2nd edn. Chapman and Hall, Boca Raton (1989)
15. Neal, R.M.: Regression and classification using gaussian process priors. In: Bernardo, J.M., Berger, J.O., Dawid, A.P., Smith, A.F.M. (eds.) Bayesian Statistics, vol. 6, pp. 475–501. Oxford University Press, Oxford (1999)
16. Neal, R.M.: Slice sampling. The Annals of Statistics 31, 705–741 (2003)
17. Vovk, V.: Derandomizing stochastic prediction strategies. Machine Learning 35, 247–282 (1999)
18. Vovk, V.: Competitive on-line statistics. International Statistical Review 69, 213–248 (2001)
19. Zhang, K., Fan, W.: Forecasting skewed biased stochastic ozone days: analyses, solutions and beyond. Knowledge and Information Systems 14, 299–326 (2008)
20. Zinkevich, M.: Online convex programming and generalized infinitesimal gradient ascent. In: Proceedings of the 20th International Conference on Machine Learning, pp. 928–936 (2003)