Protected probabilistic classification

Vladimir Vovk V.VOVK@RHUL.AC.UK
Ivan Petej I.PETEJ@RHUL.AC.UK
Alex Gammerman A.GAMMERMAN@RHUL.AC.UK
Centre for Reliable Machine Learning, Royal Holloway, University of London, Egham, Surrey, UK

Editor: Lars Carlsson, Zhiyuan Luo, Giovanni Cherubin, and Khuong An Nguyen

Abstract

This poster proposes a way of protecting algorithms for probabilistic binary classification against changes in the data distribution.

Keywords: adaptive calibration, binary classification, log-loss function, test martingales

1. Idea

A ubiquitous problem in applications of machine learning is that, soon after a predictor is trained, the distribution of the data changes, and so the predictor may need to be retrained. We propose a way of preventing a catastrophic drop in the quality of the trained predictor when the data distribution changes. To use Anscombe's (1960) insurance metaphor, our procedure provides an insurance policy against such changes. The case of regression was discussed in an earlier paper (Vovk, 2021), and in this extended abstract and its full version (Vovk et al., 2021a) we concentrate on the simpler case of binary classification. Notice that our task here is somewhat different from that of Vovk et al. (2021b): instead of detecting a changepoint in data distribution, we are merely trying to protect a given predictor from such changes.

Suppose we have a predictive system F (obtained by training a prediction algorithm) that outputs predictions $p_1, p_2, \dots \in [0,1]$ for the binary labels $y_1, y_2, \dots \in \{0,1\}$, where p_n is the predicted probability that $y_n = 1$. We can test these predictions using, e.g., the Simple Jumper test martingale S (Vovk et al., 2021b). This martingale is the likelihood ratio of a new predictive system F' to the original (or *base*) one F, and the idea is to use F' for prediction instead of F. As measured by the log loss function, the cumulative loss of the *protected predictive system* F' will be smaller than the cumulative loss of F by the log of the final value of S (in the next section we will use decimal logs).

The procedure is shown as Algorithm 1, where $B_p(\{y\}) := p1_{\{y=1\}} + (1-p)1_{\{y=0\}}$ stands for the Bernoulli distribution. One of its two parameters is a finite family $f_\epsilon : [0,1] \to [0,1]$, $\epsilon \in \mathbf{E}$, of *calibrating functions*. The intuition behind f_ϵ is that we are trying to improve the base predictions p_n, or *calibrate* them; the idea is to use a new prediction $f_\epsilon(p_n)$ instead of p_n. In our experiment described in the next section we use a subset of the family

$$f_\epsilon(p) := p + \epsilon p(1-p),$$

where $\epsilon \in [-1,1]$. For $\epsilon > 0$ we are correcting for the forecasts p being underestimates of the true probability of 1, while for $\epsilon < 0$ we are correcting for p being overestimates. The algorithm requires our family to be finite, and we choose $\mathbf{E} := \{-1, -0.5, 0, 0.5, 1\}$. The other parameter is the *jumping rate* J, which we set to 0.01, $J := 0.01$. The dependence on $\mathbf{E}$ and J is slight (Vovk et al., 2021a).

Algorithm 1: Simple Jumper protection $((p_1, p_2, \dots) \mapsto (p'_1, p'_2, \dots))$

$C_\epsilon := 1$ for all $\epsilon \in \mathbf{E}$
for $n = 1, 2, \dots$ **do**
  $C := \sum_{\epsilon \in \mathbf{E}} C_\epsilon$
  for $\epsilon \in \mathbf{E}$ **do**
    $C_\epsilon := C_\epsilon / C$
  for $\epsilon \in \mathbf{E}$ **do**
    $C_\epsilon := (1 - J) C_\epsilon + J / |\mathbf{E}|$
  $p'_n := \sum_{\epsilon \in \mathbf{E}} f_\epsilon(p_n) C_\epsilon$
  for $\epsilon \in \mathbf{E}$ **do**
    $C_\epsilon := C_\epsilon B_{f_\epsilon(p_n)}(\{y_n\})$

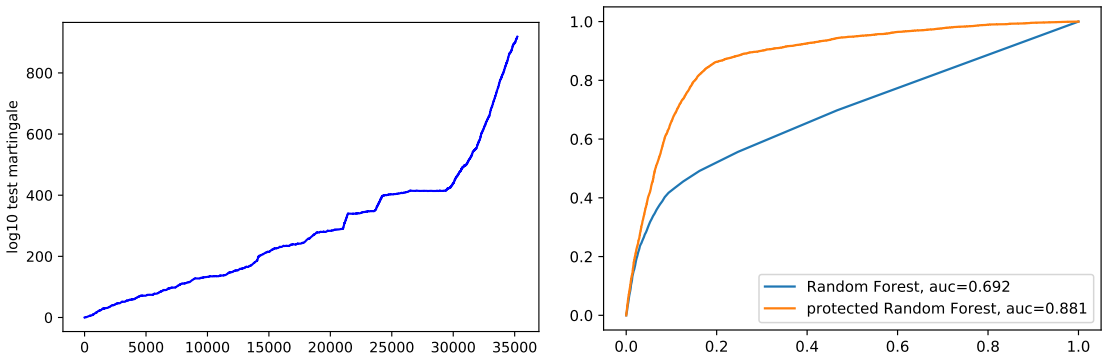

Figure 1: The Simple Jumper test martingale and the ROC curve for protection.

2. Experimental results

In this section we report results of our experiments with the popular `Bank Marketing` dataset, available from OpenML. The observations in the dataset are listed in chronological order. We take the first 10000 observations as the training set and train a random forest (`scikit-learn` function) with default parameters on it. The random forest often outputs probabilities of success that are equal to 0 or 1, and when such a prediction turns out to be wrong (which happens repeatedly), the log-loss is infinite; therefore, we truncate the predicted probabilities to the interval $[\epsilon, 1 - \epsilon]$, where we set $\epsilon := 0.1$. The resulting prediction rule is our base predictive system. After we find it, we never use the training set again, and the numbering of observations in our plot starts from the first element of the test set (i.e., the dataset in the chronological order without the training set).

The left panel of Figure 1 shows the trajectory $\log_{10} S_n$, $n = 1, \dots, 35211$, of the Simple Jumper test martingale over the test set on the log scale. It is interesting that the steepest growth of the logarithm of the test martingale starts towards the end of the dataset (covering 2008–2013), long after the financial crisis of 2007–2008 ended. The right panel gives the ROC curve for the random forest and the random forest protected by Algorithm 1. We can see that the improvement is substantial.

References

Francis J. Anscombe. Rejection of outliers. *Technometrics*, 2:123–147, 1960.

Vladimir Vovk. Protected probabilistic regression. Technical Report arXiv:2105.08669 [cs.LG], arXiv.org e-Print archive, May 2021. See the latest version at: http://alrw.net (Working Paper 34).

Vladimir Vovk, Ivan Petej, and Alex Gammerman. Protected probabilistic classification. Technical Report arXiv:2107.01726 [cs.LG], arXiv.org e-Print archive, July 2021a. See the latest version at: http://alrw.net (Working Paper 35).

Vladimir Vovk, Ivan Petej, Ilia Nouretdinov, Ernst Ahlberg, Lars Carlsson, and Alex Gammerman. Retrain or not retrain: Conformal test martingales for change-point detection. *Proceedings of Machine Learning Research*, 152, 2021b. COPA 2021, to appear.