Enhancement of prediction algorithms by betting

Vladimir Vovk

практические выводы
теории вероятностей
могут быть обоснованы
в качестве следствий
гипотез о *предельной*
при данных ограничениях
сложности изучаемых явлений

On-line Compression Modelling Project (New Series)

Working Paper #34

First posted May 18, 2021. Last revised May 20, 2021.

Project web site:
http://alrw.net

Abstract

This note proposes a procedure for enhancing the quality of probabilistic prediction algorithms via betting against their predictions. It is inspired by the success of the conformal test martingales that have been developed recently.

Contents

1 Introduction

This note is inspired by the power of betting martingales used in conformal prediction; e.g., the conformal test martingales developed in [8, 9, 11] are much more successful as compared with the older ones [7, 10]. However, the method that it proposes is independent of conformal prediction and, in particular, does not depend on the IID assumption and can be applied, e.g., to time series.

Lévy's [4, Section 39] probability integral transform translates testing a prediction algorithm that outputs probabilistic predictions (a probability forecasting system) into testing independent uniformly distributed random variables in $[0, 1]$. The numerous betting martingales developed in conformal testing often allow us to multiply the initial capital manyfold on real-world datasets (see, e.g., the results for the USPS dataset in [8]). Such betting martingales translate into test martingales that gamble successfully against the original probability forecasting system. However, since a test martingale is essentially the same thing as the likelihood ratio with the original probability forecasting system in the denominator, a successful test martingale provides us with a new (*enhanced*) probability forecasting system (the one in the numerator) that outperforms the original probability forecasting system.

The idea of this note is to use testing procedures (namely, betting martingales, such as the Sleepy Jumper and Mean Jumper) for developing better prediction algorithms. In this respect it is reminiscent of the method of defensive forecasting [6, Chapter 12], which starts from a test martingale (more generally, a strategy for Sceptic) and then develops a prediction algorithm that prevents the test martingale (more generally, Sceptic's capital) from growing. An advantage of our current procedure is that in typical cases it is computationally more efficient (in particular, it never requires finding fixed points or solving equations, as in defensive forecasting).

We will start in Section 2 from discussing Lévy's probability integral transform, which generates IID random variables distributed uniformly in $[0, 1]$ (the analogue of p-values in conformal prediction). The topic of Section 3 is online testing of the output of the probability integral transform for uniformity. Section 4 combines results of the previous two sections for the purpose of enhancing prediction algorithms. A toy simulation study is described in 5, but the enhancement procedure of Section 4 is general and widely applicable; this will be further discussed in Section 6.

2 Probability integral transform

The key fact that makes conformal testing possible is that conformal prediction outputs p-values that are independent and distributed uniformly in $[0, 1]$. Without assuming that the observations are IID, we have a similar phenomenon for the probability integral transform: if a probability forecasting system outputs probabilistic forecasts with distribution functions $F_1, F_2, \dots$ and $y_1, y_2, \dots$ are the corresponding observations, the values $F_1(y_1), F_2(y_2), \dots$ are independent

Algorithm 1 Simple Jumper betting martingale ($(u_1, u_2, \dots) \mapsto (S_1, S_2, \dots)$)

$C_{-1} := C_0 := C_1 := 1/3$
$C := 1$
for $n = 1, 2, \dots$:
    for $\epsilon \in \{-1, 0, 1\}$: $C_\epsilon := (1 - J)C_\epsilon + (J/3)C$
    for $\epsilon \in \{-1, 0, 1\}$: $C_\epsilon := C_\epsilon b^{(\epsilon)}(u_n)$
    $S_n := C := C_{-1} + C_0 + C_1$

and distributed uniformly in $[0, 1]$. (To avoid complications, let us assume that all F_n are continuous.)

The uniformity of the probability integral transform was used by Lévy [4, Section 39] as the foundation of his theory of denumerable probabilities (which allowed him to avoid using the then recent axiomatic foundation suggested by Kolmogorov in his *Grundbegriffe* [3]). Modern papers, including [1], usually refer to Rosenblatt [5], who disentangled Lévy's argument from his concern with the foundations of probability; Rosenblatt, however, refers to Lévy's 1937 book [4] in his paper.

The probability integral transform can be used for testing the underlying probability forecasting system considered as the data-generating distribution. See, e.g., [1, Sections 3.8 and 4.7].

3 Testing probability forecasting systems

To transform the probability integral transforms $u_1, u_2, \dots$ into a test martingale we use, as in [8, 9, 11], the *Simple Jumper* betting martingale given as Algorithm 1, where

$$b^{(\epsilon)}(u) := 1 + \epsilon(u - 0.5). \tag{1}$$

In the next section we set $J := 0.01$, as in [11]. For the intuition behind Simple Jumper, see [11] (and the more complicated Sleepy Jumper is described in detail in [10, Section 7.1]).

A safer option than the Simple Jumper is the Mean Jumper betting martingale [8], which is defined to be the average of Simple Jumpers over a finite set $\mathcal{J}$ of J including $J = 1$ (such as $J \in \mathcal{J} := \{10^{-3}, 10^{-2}, 10^{-1}, 1\}$). The inclusion of $J = 1$ is convenient since the corresponding Simple Jumper is identical 1, and so the Mean Jumper never drops in value below $1/|\mathcal{J}|$.

4 The enhancement procedure

Given a prediction algorithm A and a betting martingale S, our enhancement procedure produces the prediction algorithm A' such that S is the likelihood ratio process $\mathrm{d}A'/\mathrm{d}A$.

Algorithm 2 Simple Jumper enhancement $((F_1, F_2, \dots) \mapsto (F'_1, F'_2, \dots))$

$C_{-1} := C_0 := C_1 := 1/3$
$C := 1$
for $n = 1, 2, \dots$:
    for $\epsilon \in \{-1, 0, 1\}$: $C_\epsilon := (1 - J)C_\epsilon + (J/3)C$
    $\bar{\epsilon} := (C_1 - C_{-1})/C$
    $F'_n := B^{(\bar{\epsilon})}(F_n)$
    for $\epsilon \in \{-1, 0, 1\}$: $C_\epsilon := C_\epsilon b^{(\epsilon)}(F_n(y_n))$
    $S_n := C := C_{-1} + C_0 + C_1$

If the predictive distribution function output by A is $F_n = F_n(y)$, the corresponding predictive density is $f_n = f_n(y) = F'_n(y)$, and the betting function output by S is b_n, the enhanced predictive density is $b_n(F_n)f_n$. It integrates to 1 since $b_n(F_n)f_n = (B_n(F_n))'$, where B_n is the indefinite integral $B_n(v) := \int_0^v b_n$ of b_n, so that $B'_n = b_n$. We can see that the distribution function for the enhanced algorithm is $B_n(F_n)$.

The procedure of enhancement is given as Algorithm 2, where

$$B^{(\epsilon)}(v) := \int_0^v b^{(\epsilon)}(u)\,\mathrm{d}u = \frac{\epsilon}{2}v^2 + \left(1 - \frac{\epsilon}{2}\right)v$$

(cf. (1)). Algorithm 2 uses the fact that the Simple Jumper outputs betting functions (1) for $\epsilon = \bar{\epsilon}$. Let us check this fact. According to Algorithm 1, the value of the martingale at the last step is

$$\begin{aligned}\sum_\epsilon C_\epsilon b^{(\epsilon)}(u) &= \sum_\epsilon C_\epsilon(1 + \epsilon(u - 0.5)) \\ &= C_E(1 + E(u - 0.5)) + C_0 + C_{-E}(1 - E(u - 0.5)) \\ &= (C_E + C_0 + C_{-E}) + (C_E - C_{-E})E(u - 0.5) \\ &\propto 1 + \frac{C_E - C_{-E}}{C_E + C_0 + C_{-E}}E(u - 0.5),\end{aligned}$$

where we assume that the range of ϵ is $\{-E, 0, E\}$. (At this time $E = 1$, but we will use $E = 2$ in Section 5.) We can see that the slope of the resulting straight line is

$$\epsilon := \frac{C_E - C_{-E}}{C_E + C_0 + C_{-E}}E.$$

In this note we evaluate the performance of the original and enhanced algorithms using the log loss function; namely the loss of a predictive density f for a realized outcome y is $-\log f(x)$ (two pictures in the experimental Section 5 will use base 10 logarithms). It is a proper loss function, meaning that the optimal expected loss is attained by the true predictive density [2, Section 4].

When S is a Mean Jumper martingale, the enhanced algorithm is guaranteed not to lose much as compared with the original algorithm when the quality is measured by the log loss function: namely, the cumulative log loss for A' is at most the cumulative log loss for A plus $\log|\mathcal{J}|$.

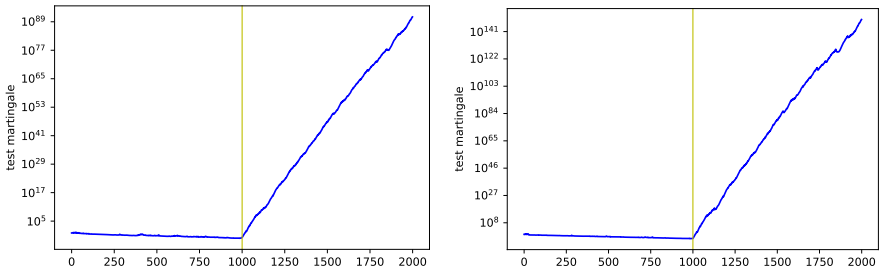

Figure 1: The Simple Jumper test martingale. Left panel: the standard one ($\epsilon \in \{-1, 0, 1\}$). Right panel: $\epsilon \in \{-2, 0, 2\}$.

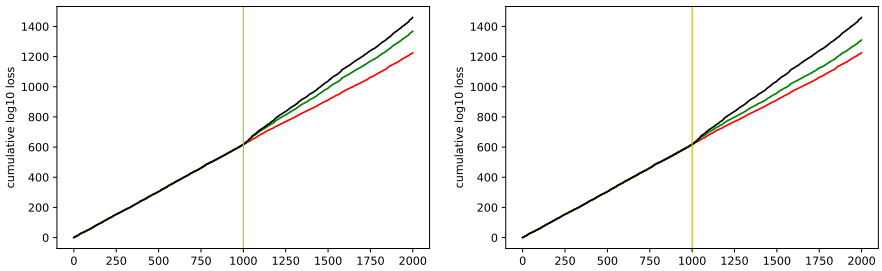

Figure 2: The cumulative log losses of three prediction algorithms (to the left of the changepoint the three lines coincide or are visually indistinguishable). Left panel: $\epsilon \in \{-1, 0, 1\}$. Right panel: $\epsilon \in \{-2, 0, 2\}$.

5 A simulation study

We consider a dataset that consists of independent Gaussian observations: the first 1000 are generated from $N(0, 1)$, and another 1000 from $N(1, 1)$. Our base prediction algorithm does not know that there is a changepoint at time 1000 and always predicts $N(0, 1)$. The seed of the pseudo random number generator (in NumPy) is always 2021.

First we run the Simple Jumper martingale (Algorithm 1 with $J = 0.01$) on our dataset. The left panel of Figure 1 shows its trajectory; it loses capital before the changepoint, but quickly regains it afterwards. Its final value is 1.100×10^{91}.

Remark 1. The possibility of losing so much capital before the changepoint (the value of the Simple Jumper at the changepoint is 0.0114) shows that using the Simple Jumper is risky. If we want to play safe, we can use the Mean Jumper instead of the Simple Jumper. As mentioned above, this will bound our loss to $\log |\mathcal{J}|$ as compared with the original algorithm.

The cumulative log loss of the enhanced version of the base prediction algo-

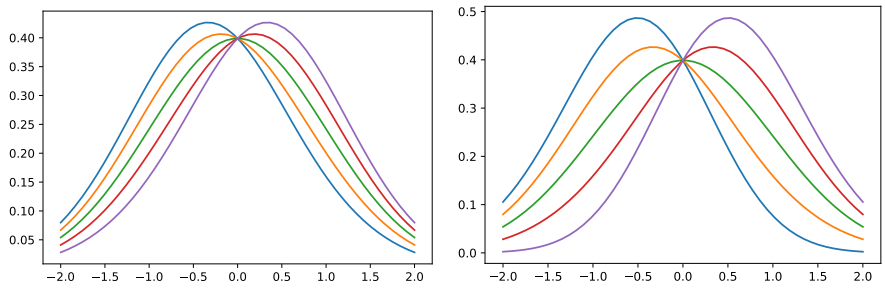

Figure 3: The enhanced predictive distributions. Left panel: the range of ϵ is $\{-1, -0.5, 0, 0.5, 1\}$. Right panel: $\epsilon \in \{-2, -1, 0, 1, 2\}$.

rithm is shown as the green line in the left panel of Figure 2. The black line corresponds to the base algorithm, and the red line to the impossible *oracle algorithm*, which knows the truth and predicts with $N(0, 1)$ before the changepoint and $N(1, 1)$ afterwards. According to Figure 1 (left panel), the difference between the final values of the black and green lines is about 91.

To understand better the mechanism of enhancement in this case, notice that the Simple Jumper outputs betting functions b of the form (1), where $\epsilon \in [-1, 1]$ (usually $\epsilon \notin \{-1, 0, 1\}$). The corresponding predictive distributions $b(F)f$ (where f is the standard normal density and F its distribution function) are shown in the left panel of Figure 3 for five values of ϵ. We can see that our range of ϵ, $\epsilon \in [-1, 1]$, is not sufficiently ambitious and does not allow us to approximate $N(1, 1)$ well.

Replacing the range $\{-1, 0, 1\}$ for ϵ by $\{-2, 0, 2\}$, we obtain the right panel of Figure 3. The right-most graph in that panel now looks closer to the density of $N(1, 1)$. We cannot extend the range of ϵ further without (1) ceasing to be a calibrator. (Of course, the calibrator does not have to be linear, but let us stick to the simplest choices in this version of the paper.)

Using the range $\{-2, 0, 2\}$ for ϵ leads to the right panels of Figures 1 and 2. We can see that in the right panel of Figure 2 the performance of the enhanced algorithm is much close to that of the oracle algorithm than in the left panel.

Figure 2 provides useful and precise information, but it is not very intuitive. A cruder approach is to translate the probabilistic forecasts into point predictions. Figure 4 uses the medians of predictive distributions as point predictions. In the case of the base algorithm, the prediction is always 0 (the median of $N(0, 1)$), for the oracle algorithm it is 0 before the changepoint and 1 afterwards, and for the enhanced algorithm the predictions are shown in the figure. We can see that the right panel of Figure 4 is a better approximation to the oracle predictions.

Remark 2. To compute the point predictions shown in Figure 4, we can use the representation of the betting function b for the Simple Jumper in the form

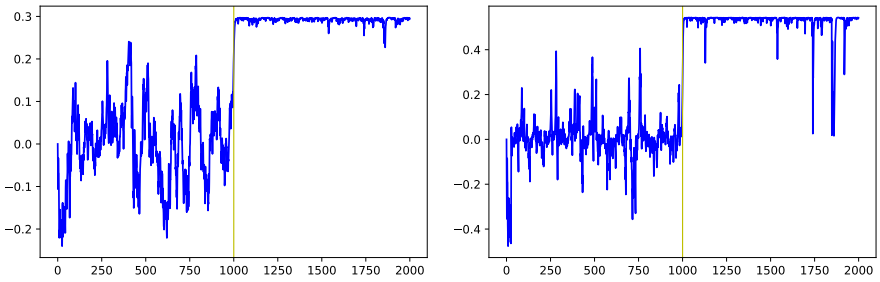

Figure 4: The enhanced point predictions (medians of the enhanced predictive distributions). Left panel: $\epsilon \in \{-1, 0, 1\}$. Right panel: $\epsilon \in \{-2, 0, 2\}$.

(1) with

$$\epsilon := \frac{C_E - C_{-E}}{C_E + C_0 + C_{-E}} E,$$

where $\{-E, 0, E\}$ is the range of ϵ (so that $E = 1$ in the left-hand panels and $E = 2$ in the right-hand ones). The indefinite integral of the betting function is

$$B(v) = \int_0^v b(u)\,\mathrm{d}u = \int_0^v (1 + \epsilon(u - 0.5))\,\mathrm{d}u = \left(1 - \frac{\epsilon}{2}\right) v + \frac{\epsilon}{2} v^2.$$

Solving the quadratic equation $B(v) = 0.5$ we get

$$v = \frac{\epsilon - 2 + \sqrt{\epsilon^2 + 4}}{2\epsilon}. \tag{2}$$

Since the distribution function of the enhanced probability forecast is $B(F)$, where $F = N(0, 1)$ is the distribution function of the original probability forecast, we obtain the median of the enhanced distribution as the v quantile of $N(0, 1)$, with v defined by (2).

6 Conclusion

This section briefly discusses possible directions of further research.

To understand better the potential of the new method, further simulation studies and, more importantly, empirical studies are required. In particular, this note uses only one proper loss function, namely the log loss function. An interesting alternative is CRPS, or continuous ranked probability score [2, Section 4.2].

Another direction is to improve the performance of test martingales and, therefore, enhanced prediction algorithms in various model situations, similarly to [9]. The framework of Section 5 is an example of such a model situation.

It is important to get rid of the assumption that the predictive distribution is continuous, which we made in Section 2. This is needed, e.g., to cover the case

of classification. This could be achieved by adapting the smoothing procedure [10, (2.20)], which is standard in conformal prediction.

This note assumes that the observations y_n are real numbers, whereas the standard setting of machine learning is where the observations are pairs (x_n, y_n). Our method is applicable in this case as well if we assume that the x_n are constant. The cleanest approach, however, would be not to assume anything about the x_n and use the game-theoretic foundations of probability [6]. For the game-theoretic treatment of the probability integral transform, see [1, Theorem 2(b)].

Acknowledgments

Glenn Shafer's and Nell Painter's advice is gratefully appreciated. This work has been supported by Amazon and Stena Line.

References

[1] A. Philip Dawid and Vladimir Vovk. Prequential probability: Principles and properties. *Bernoulli*, 5:125–162, 1999.

[2] Tilmann Gneiting and Adrian E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102:359–378, 2007.

[3] Andrei N. Kolmogorov. *Grundbegriffe der Wahrscheinlichkeitsrechnung*. Springer, Berlin, 1933. English translation: *Foundations of the Theory of Probability*. Chelsea, New York, 1950.

[4] Paul Lévy. *Théorie de l'addition des variables aléatoires*. Gauthier-Villars, Paris, 1937. Second edition: 1954.

[5] Murray Rosenblatt. Remarks on a multivariate transformation. *Annals of Mathematical Statistics*, 23:470–472, 1952.

[6] Glenn Shafer and Vladimir Vovk. *Game-Theoretic Foundations for Probability and Finance*. Wiley, Hoboken, NJ, 2019.

[7] Vladimir Vovk. Testing randomness online, On-line Compression Modelling project (New Series), `http://alrw.net`, Working Paper 24, June 2019. Journal version: *Statistical Science* (to appear).

[8] Vladimir Vovk. Testing for concept shift online, On-line Compression Modelling project (New Series), `http://alrw.net`, Working Paper 31, December 2020.

[9] Vladimir Vovk. Conformal testing in a binary model situation, On-line Compression Modelling project (New Series), `http://alrw.net`, Working Paper 33, April 2021.

[10] Vladimir Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer, New York, 2005.

[11] Vladimir Vovk, Ivan Petej, Ilia Nouretdinov, Ernst Ahlberg, Lars Carlsson, and Alex Gammerman. Retrain or not retrain: Conformal test martingales for change-point detection, On-line Compression Modelling project (New Series), `http://alrw.net`, Working Paper 32, February 2021.