Plug-in martingales for testing exchangeability on-line

Valentina Fedorova VALENTINA@CS.RHUL.AC.UK
Alex Gammerman ALEX@CS.RHUL.AC.UK
Ilia Nouretdinov ILIA@CS.RHUL.AC.UK
Vladimir Vovk V.VOVK@RHUL.AC.UK
Computer Learning Research Centre, Royal Holloway, University of London, Egham, Surrey, TW20 0EX, UK

Abstract

A standard assumption in machine learning is the exchangeability of data, which is equivalent to assuming that the examples are generated from the same probability distribution independently. This paper is devoted to testing the assumption of exchangeability on-line: the examples arrive one by one, and after receiving each example we would like to have a valid measure of the degree to which the assumption of exchangeability has been falsified. Such measures are provided by exchangeability martingales. We extend known techniques for constructing exchangeability martingales and show that our new method is competitive with the martingales introduced before. Finally we investigate the performance of our testing method on two benchmark datasets, USPS and Statlog Satellite data; for the former, the known techniques give satisfactory results, but for the latter our new more flexible method becomes necessary.

1. Introduction

Many machine learning algorithms have been developed to deal with real-life high dimensional data. In order to state and prove properties of such algorithms it is standard to assume that the data satisfy the exchangeability assumption (although some algorithms make different assumptions or, in the case of prediction with expert advice, do not make any statistical assumptions at all). These properties can be violated if the assumption is not satisfied, which makes it important to test the data for satisfying it.

Appearing in *Proceedings of the 29th International Conference on Machine Learning*, Edinburgh, Scotland, UK, 2012.

Note that the popular assumption that the data is i.i.d. (independent and identically distributed) has the same meaning for testing as the exchangeability assumption. A joint distribution of an infinite sequence of examples is exchangeable if it is invariant w.r. to any permutation of examples. Hence if the data is i.i.d., its distribution is exchangeable. On the other hand, by de Finetti's theorem (see, e.g., Schervish, 1995, p. 28) any exchangeable distribution on the data (a potentially infinite sequence of examples) is a mixture of distributions under which the data is i.i.d. Therefore, testing for exchangeability is equivalent to testing for being i.i.d.

Traditional statistical approaches to testing are inappropriate for high dimensional data (see, e.g., Vapnik, 1998, pp. 6–7). To address this challenge a previous study (Vovk et al., 2003) suggested a way of on-line testing by employing the theory of conformal prediction and calculating exchangeability martingales. Basically testing proceeds in two steps. The first step is implemented by a conformal predictor that outputs a sequence of p-values. The sequence is generated in the on-line mode: examples are presented one by one and for each new example a p-value is calculated from this and all the previous examples. For the second step the authors introduced exchangeability martingales that are functions of the p-values and track the deviation from the assumption. Once the martingale grows up to a large value (20 and 100 are convenient rules of thumb) the exchangeability assumption can be rejected for the data.

This paper proposes a new way of constructing martingales in the second step of testing. To construct an exchangeability martingale based on the sequence of p-values we need a betting function, which determines the contribution of a p-value to the value of the martingale. In contrast to the previous studies that use a fixed betting function the new martingale tunes its betting function to the sequence to detect any deviation from

the assumption. We show that this martingale, which we call a plug-in martingale, is competitive with all the martingales covered by the previous studies; namely, asymptotically the former grows faster than the latter.

1.1. Related work

The first procedure of testing exchangeability on-line is described in Vovk et al. (2003). The core testing mechanism is an exchangeability martingale. Exchangeability martingales are constructed using a sequence of p-values. The algorithm for generating p-values assigns small p-values to unusual examples. It implies the idea of designing martingales that would have a large value if too many small p-values were generated, and suggests corresponding power martingales. Other martingales (simple mixture and sleepy jumper) implement more complicated strategies, but follow the same idea of scoring on small p-values.

Ho (2005) applies power martingales to the problem of change detection in time-varying data streams. The author shows that small p-values inflate the martingale values and suggests to use the martingale difference as another test for the problem.

1.2. This paper

To the best of our knowledge, no study has aimed to find any other ways of translating p-values into a martingale value. In this paper we propose a new more flexible method of constructing exchangeability martingales for a given sequence of p-values.

The rest of the paper is organised as follows. Section 2 gives the definition of exchangeability martingales. Section 3 presents the construction of plug-in exchangeability martingales, explains the rationale behind them, and compares them to the power martingales, which have been used previously. Section 4 shows experimental results of testing two real-life datasets for exchangeability; for one of these datasets power martingales work satisfactorily and for the other one the greater flexibility of plug-in martingales becomes essential. Section 5 summarises the paper.

2. Exchangeability martingales

This section outlines necessary definitions and results of the previous studies.

2.1. Exchangeability

Consider a sequence of random variables $(Z_1, Z_2, \ldots)$ that all take values in the same example space. Then the joint probability distribution $\mathbf{P}(Z_1, \ldots, Z_N)$ of a finite number of the random variables is *exchangeable* if it is invariant under any permutation of the random variables. The joint distribution of infinite number of random variables $(Z_1, Z_2, \ldots)$ is *exchangeable* if the marginal distribution $\mathbf{P}(Z_1, \ldots, Z_N)$ is exchangeable for every N.

2.2. Martingales for testing

As in Vovk et al. (2003), the main tool for testing exchangeability on-line is a martingale. The value of the martingale reflects the strength of evidence against the exchangeability assumption. An *exchangeability martingale* is a sequence of non-negative random variables $S_0, S_1, \ldots$ that keep the conditional expectation:

$$S_n \geq 0$$
$$S_n = \mathbf{E}(S_{n+1} \mid S_1, \ldots, S_n),$$

where $\mathbf{E}$ refers to the expected value with respect to any exchangeable distribution on examples. We also assume $S_0 = 1$. Note that we will obtain an equivalent definition if we replace "any exchangeable distribution on examples" by "any distribution under which the examples are i.i.d." (remember the discussion of de Finetti's theorem in Section 1).

To understand the idea behind martingale testing we can imagine a game where a player starts from the capital of 1, places bets on the outcomes of a sequence of events, and never risks bankruptcy. Then a martingale corresponds to a strategy of the player, and its value reflects the acquired capital. According to Ville's inequality (see Ville, 1939, p. 100),

$$\mathbf{P}\Big\{\exists n : S_n \geq C\Big\} \leq 1/C, \quad \forall C \geq 1,$$

it is unlikely for any S_n to have a large value. For the problem of testing exchangeability, if the final value of a martingale is large then the exchangeability assumption for the data can be rejected with the corresponding probability.

2.3. On-line calculation of p-values

Let $(z_1, z_2, \ldots)$ denote a sequence of examples. Each example z_i is the vector representing a set of attributes x_i and a label y_i: $z_i = (x_i, y_i)$. In this paper we use conformal predictors to generate a sequence of p-values that corresponds to the given examples. The general idea of conformal prediction is to test how well a new example fits to the previously observed examples. For this purpose a "nonconformity measure" is defined. This is a function that estimates the strangeness of one example with respect to others:

$$\alpha_i = A\Big(z_i, \{z_1, \ldots, z_n\}\Big),$$

Algorithm 1 Generating p-values on-line

Input: $(z_1, z_2, \ldots)$ data for testing
Output: $(p_1, p_2, \ldots)$ sequence of p-values
for $i = 1, 2, \ldots$ **do**
  observe a new example z_i
  for $j = 1$ **to** i **do**
    $\alpha_j = A\Big(z_j, \{z_1, \ldots, z_i\}\Big)$
  end for
  $p_i = \frac{\#\{j:\alpha_j > \alpha_i\} + \theta_i \#\{j:\alpha_j = \alpha_i\}}{i}$
end for

where in general $\{\ldots\}$ stands for a multiset (the same element may be repeated more than once) rather than a set. Typically, each example is assigned a "nonconformity score" α_i based on some prediction method. In this paper we deal with the classification problem and the 1-Nearest Neighbor (1-NN) algorithm is used as the underling method to compute the nonconformity scores. The algorithm is simple but it works well enough in many cases (see, e.g., Hastie et al., 2001, pp. 422–427). A natural way to define the nonconformity score of an example is by comparing its distance to the examples with the same label to its distance to the examples with a different label:

$$\alpha_i = \frac{\min_{j \neq i: y_i = y_j} d(x_i, x_j)}{\min_{j \neq i: y_i \neq y_j} d(x_i, x_j)}, \tag{1}$$

where $d(x_i, x_j)$ is the Euclidean distance. According to the chosen nonconformity measure, α_i is high if the example is close to another example with a different label and far from any examples with the same label.

Using the calculated nonconformity scores of all observed examples, the p-value p_n that corresponds to an example z_n is calculated as

$$p_n = \frac{\#\{i : \alpha_i > \alpha_n\} + \theta_n \#\{i : \alpha_i = \alpha_n\}}{n},$$

where θ_n is a random number from $[0, 1]$ and the symbol $\#$ means the cardinality of a set. Algorithm 1 summarises the process of on-line calculation of p-values (it is clear that it can also be applied to a finite dataset $(z_1, \ldots, z_n)$ producing a finite sequence $(p_1, \ldots, p_n)$ of p-values).

The following is a standard result in the theory of conformal prediction (see, e.g., Vovk et al. 2003, Theorem 1).

Theorem 1. *If examples $(z_1, z_2, \ldots)$ (resp. $(z_1, z_2, \ldots, z_n)$) satisfy the exchangeability assumption, Algorithm 1 produces p-values $(p_1, p_2, \ldots)$ (resp. $(p_1, p_2, \ldots, p_n)$) that are independent and uniformly distributed in $[0, 1]$.*

The property that the examples generated by an exchangeable distribution provide uniformly and independently distributed p-values allows us to test exchangeability by calculating martingales as functions of the p-values.

3. Martingales based on p-values

This section focuses on the second part of testing: given the sequence of p-values a martingale is calculated as a function of the p-values.

For each $i \in \{1, 2, \ldots\}$, let $f_i : [0, 1]^i \to [0, \infty)$. Let $(p_1, p_2, \ldots)$ be the sequence of p-values generated by Algorithm 1. We consider martingales S_n of the form

$$S_n = \prod_{i=1}^{n} f_i(p_i), \quad n = 1, 2, \ldots, \tag{2}$$

where we denote $f_i(p) = f_i(p_1, \ldots, p_{i-1}, p)$ and call the function $f_i(p)$ a *betting function*.

To be sure that (2) is indeed a martingale we need the following constraint on the betting functions f_i:

$$\int_0^1 f_i(p)dp = 1, \quad i = 1, 2, \ldots$$

Then we can check:

$$\mathbf{E}(S_{n+1} \mid S_0, \ldots, S_n) = \int_0^1 \prod_{i=1}^{n} \Big(f_i(p_i)\Big) f_{n+1}(p)dp$$
$$= \prod_{i=1}^{n} \Big(f_i(p_i)\Big) \int_0^1 f_{n+1}(p)dp = \prod_{i=1}^{n} f_i(p_i) = S_n.$$

Using representation (2) we can update the martingale on-line: having calculated a p-value p_i for a new example in Algorithm 1 the current martingale value becomes $S_i = S_{i-1} \cdot f_i(p_i)$. To define the martingales completely we need to describe the betting functions f_i.

3.1. Previous results: power and simple mixture martingales

Previous studies (Vovk et al., 2003) have proposed to use a fixed betting function

$$\forall i : \ f_i(p) = \varepsilon p^{\varepsilon - 1},$$

where $\varepsilon \in [0, 1]$. Several martingales were constructed using the function. The *power martingale* for some ε, denoted as M_n^{ε}, is defined as

$$M_n^{\varepsilon} = \prod_{i=1}^{n} \varepsilon p_i^{\varepsilon - 1}.$$

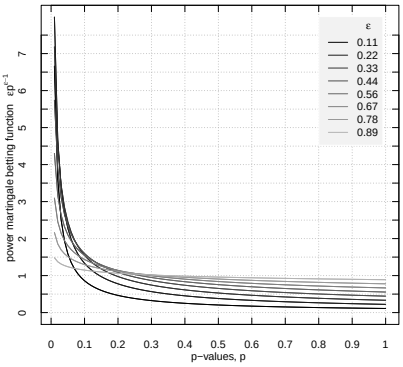

Figure 1. The betting functions that are used to construct the power and simple mixture martingales.

The *simple mixture* martingale, denoted as M_n, is the mixture of power martingales over different $\varepsilon \in [0,1]$:

$$M_n = \int_0^1 M_n^\varepsilon d\varepsilon.$$

Such a martingale will grow only if there are many small p-values in the sequence. This follows from the shape of the betting functions: see Figure 1. If the generated p-values concentrate in any other part of the unit interval, we cannot expect the martingale to grow. So it might be difficult to reject the assumption of exchangeability for such sequences.

3.2. New plug-in approach

3.2.1. Plug-in martingale

Let us use an estimated probability density function as the betting function $f_i(p)$. At each step the probability density function is estimated using the accumulated p-values:

$$\rho_i(p) = \widehat{\rho}(p_1, \dots, p_{i-1}, p), \tag{3}$$

where $\widehat{\rho}(p_1, \dots, p_{i-1}, p)$ is the estimate of the probability density function using the p-values $p_1, \dots, p_{i-1}$ output by Algorithm 1.

Substituting these betting functions into (2) we get a new martingale that we call a *plug-in* martingale. The martingale avoids betting if the p-values are distributed uniformly, but if there is any peak it will be used for betting.

Estimating a probability density function. In our experiments we have used the statistical environment and language R. The `density` function in its `Stats` package implements kernel density estimation with different parameters. But since p-values always lie in the unit interval, the standard methods of kernel density estimation lead to poor results for the points that are near the boundary. To get better results for the boundary points the sequence of p-values is reflected to the left from zero and to the right from one. Then the kernel density estimate is calculated using the extended sample $\cup_{i=1}^n \{-p_i, p_i, 2-p_i\}$. The estimated density function is set to zero outside the unit interval and then normalised to integrate to one. For the results presented in this paper the parameters used are the Gaussian kernel and Silverman's "rule of thumb" for bandwidth selection. Other settings have been tried as well, but the results are comparable and lead to the same conclusions.

The values S_n of the plug-in martingale can be updated recursively. Suppose computing the nonconformity scores $(\alpha_1, \dots, \alpha_n)$ from $(z_1, \dots, z_n)$ takes time $g(n)$ and evaluating (3) takes time $h(n)$. Then updating S_{n-1} to S_n takes time $O(g(n) + n + h(n))$: indeed, it is easy to see that calculating the rank of α_n in the multiset $\{\alpha_1, \dots, \alpha_n\}$ takes time $\Theta(n)$.

The performance of the plug-in martingale on real-life datasets will be presented in Section 4. The rest of the current section proves that the plug-in martingale provides asymptotically a better growth rate than any martingale with a fixed betting function. To prove this asymptotical property of the plug-in martingale we need the following assumptions.

3.2.2. Assumptions

Consider an infinite sequence of p-values $(p_1, p_2, \dots)$. (This is simply a deterministic sequence.) For its finite prefix $(p_1, \dots, p_n)$ define the corresponding empirical probability measure $\mathbf{P}_n$: for a Borel set A in $\mathbf{R}$,

$$\mathbf{P}_n(A) = \frac{\#\{i = 1, \dots, n : p_i \in A\}}{n}.$$

We say that the sequence $(p_1, p_2, \dots)$ is *stable* if there exists a probability measure $\mathbf{P}$ on $\mathbf{R}$ such that:

1. $\mathbf{P}_n \xrightarrow[n\to\infty]{\text{weak}} \mathbf{P}$;
2. there exists a positive continuous density function $\rho(p)$ for $\mathbf{P}$: for any Borel set A in $\mathbf{R}$, $\mathbf{P}(A) = \int_A \rho(p) dp$.

Intuitively, the stability means that asymptotically the sequence of p-values can be described well by a probability distribution.

Consider a sequence $(f_1(p), f_2(p), \dots)$ of betting functions. (This is simply a deterministic sequence of functions $f_i : [0,1] \to [0,\infty)$, although we are particularly interested in the functions $f_i(p) = \rho_i(p)$, as defined in (3).) We say that this sequence is *consistent for* $(p_1, p_2, \dots)$ if

$$\log\bigl(f_n(p)\bigr) \xrightarrow[n\to\infty]{\text{uniformly in } p} \log\bigl(\rho(p)\bigr).$$

Intuitively, consistency is an assumption about the algorithm that we use to estimate the function $\rho(p)$; in the limit we want a good approximation.

3.2.3. Growth rate of plug-in martingale

The following result says that, under our assumptions, the logarithmic growth rate of the plug-in martingale is better than that of any martingale with a fixed betting function (remember that by a betting function we mean any function mapping $[0,1]$ to $[0,\infty)$).

Theorem 2. *If a sequence $(p_1, p_2, \dots) \in [0,1]^\infty$ is stable and a sequence of betting functions $\bigl(f_1(p), f_2(p), \dots\bigr)$ is consistent for it then, for any positive continuous betting function f,*

$$\liminf_{n\to\infty} \left(\frac{1}{n} \sum_{i=1}^{n} \log\bigl(f_i(p_i)\bigr) - \frac{1}{n} \sum_{i=1}^{n} \log\bigl(f(p_i)\bigr) \right) \geq 0$$

First we explain the meaning of Theorem 2 and then prove it. According to representation (2) after n steps the martingale grows to

$$\prod_{i=1}^{n} f_i(p_i). \tag{4}$$

Note that if for any p-value $p \in [0,1]$ we have $f_i(p) = 0$ then the martingale can become zero and will never change after that. Therefore, it is reasonable to consider positive $f_i(p)$. Then we can rewrite product (4) as sum of logarithms, which gives us the logarithmic growth of the martingale:

$$\sum_{i=1}^{n} \log\Bigl(f_i(p_i)\Bigr).$$

We assume that the sequence of p-values is stable and the sequence of estimated probability density functions that is used to construct the plug-in martingale is consistent. Then the limit inequality from Theorem 2 states that the logarithmic growth rate of the plug-in martingale is asymptotically at least as high as that of any martingale with a fixed betting function (which have been suggested in previous studies).

To prove Theorem 2 we will use the following lemma.

Lemma 1. *For any probability density functions ρ and f (so that $\int_0^1 \rho(p)dp = 1$ and $\int_0^1 f(p)dp = 1$),*

$$\int_0^1 \log\Bigl(\rho(p)\Bigr)\rho(p)dp \geq \int_0^1 \log\Bigl(f(p)\Bigr)\rho(p)dp.$$

Proof of Lemma 1. It is well known (Kullback, 1959, p. 14) that the Kullback–Leibler divergence is always non-negative:

$$\int_0^1 \log\Bigl(\frac{\rho(p)}{f(p)}\Bigr)\rho(p)dp \geq 0.$$

This is equivalent to the inequality asserted by Lemma 1. □

Proof of Theorem 2. Suppose that, contrary to the statement of Theorem 2, there exists $\delta > 0$ such that

$$\liminf_{n\to\infty} \left(\frac{1}{n} \sum_{i=1}^{n} \log\bigl(f_i(p_i)\bigr) - \frac{1}{n} \sum_{i=1}^{n} \log\bigl(f(p_i)\bigr) \right) < -\delta. \tag{5}$$

Then choose an ε satisfying $0 < \varepsilon < \delta/4$.

Substituting the definition of $\rho(p)$ into Lemma 1 we obtain

$$\int_0^1 \log\Bigl(\rho(p)\Bigr)d\mathbf{P} \geq \int_0^1 \log\Bigl(f(p)\Bigr)d\mathbf{P}. \tag{6}$$

From the stability of $(p_1, p_2, \dots)$ it follows that there exists a number $N_1 = N_1(\varepsilon)$ such that, for all $n > N_1$,

$$\left| \int_0^1 \log\Bigl(f(p)\Bigr)d\mathbf{P}_n - \int_0^1 \log\Bigl(f(p)\Bigr)d\mathbf{P} \right| < \varepsilon$$

and

$$\left| \int_0^1 \log\Bigl(\rho(p)\Bigr)d\mathbf{P}_n - \int_0^1 \log\Bigl(\rho(p)\Bigr)d\mathbf{P} \right| < \varepsilon.$$

Then inequality (6) implies that, for all $n \geq N_1$,

$$\int_0^1 \log\Bigl(\rho(p)\Bigr)d\mathbf{P}_n \geq \int_0^1 \log\Bigl(f(p)\Bigr)d\mathbf{P}_n - 2\varepsilon.$$

By the definition of the probability measure $\mathbf{P}_n$, the last inequality is the same thing as

$$\frac{1}{n} \sum_{i=1}^{n} \log\Bigl(\rho(p_i)\Bigr) \geq \frac{1}{n} \sum_{i=1}^{n} \log\Bigl(f(p_i)\Bigr) - 2\varepsilon. \tag{7}$$

By the consistency of $\bigl(f_1(p), f_2(p), \dots\bigr)$ there exists a number $N_2 = N_2(\varepsilon)$ such that, for all $i > N_2$ and all $p \in [0,1]$,

$$\Bigl|\log\bigl(f_i(p)\bigr) - \log\bigl(\rho(p)\bigr)\Bigr| < \varepsilon. \tag{8}$$

Let us define the number

$$M = \max_{i,p} \left| \log\bigl(f_i(p)\bigr) - \log\bigl(\rho(p)\bigr) \right|. \tag{9}$$

From (8) and (9) we have

$$\left| \log\bigl(f_i(p)\bigr) - \log\bigl(\rho(p)\bigr) \right| \le \begin{cases} M, & i \le N_2 \\ \varepsilon, & i > N_2. \end{cases} \tag{10}$$

Denote $N_3 = \max(N_1, N_2)$. Then, using (10) and (7), we obtain, for all $n > N_3$,

$$\frac{1}{n} \sum_{i=1}^{n} \log\Bigl(f_i(p_i)\Bigr) \ge \frac{1}{n} \sum_{i=1}^{n} \log\Bigl(f(p_i)\Bigr) - 3\varepsilon - \frac{MN_3}{n}.$$

Denoting $N_4 = \max(N_3, \frac{MN_3}{\varepsilon})$, we can rewrite the last inequality as

$$\frac{1}{n} \sum_{i=1}^{n} \log\Bigl(f_i(p_i)\Bigr) \ge \frac{1}{n} \sum_{i=1}^{n} \log\Bigl(f(p_i)\Bigr) - 4\varepsilon,$$

for all $n > N_4$. Finally, recalling that $\varepsilon < \frac{\delta}{4}$, we have, for all $n > N_4$,

$$\frac{1}{n} \sum_{i=1}^{n} \log\Bigl(f_i(p_i)\Bigr) - \frac{1}{n} \sum_{i=1}^{n} \log\Bigl(f(p_i)\Bigr) \ge -\delta.$$

This contradicts (5) and therefore completes the proof of Theorem 2. □

4. Empirical results

In this section we investigate the performance of our plug-in martingale and compare it with that of the simple mixture martingale. Two real-life datasets have been tested for exchangeability: the USPS dataset and the Statlog Satellite dataset.

4.1. USPS dataset

Data The US Postal Service (USPS) dataset consists of 7291 training examples and 2007 test examples of handwritten digits, from 0 to 9. The data were collected from real-life zip codes. Each example is described by the 256 attributes representing the pixels for displaying a digit on the 16×16 gray-scaled image and its label. It is well known that the examples in this dataset are not perfectly exchangeable (Vovk et al., 2003), and any reasonable test should reject exchangeability there. In our experiments we merge the training and test sets and perform testing for the full dataset of 9298 examples.

Figure 2 shows the typical performance of the martingales when the exchangeability assumption is satisfied for sure: all examples have been randomly shuffled before the testing.

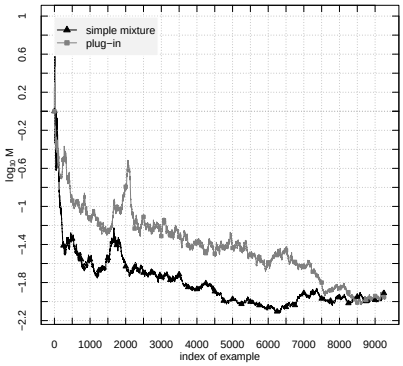

Figure 2. The growth of the martingales for the USPS dataset randomly shuffled before on-line testing. The exchangeability assumption is satisfied: the final martingale values are about 0.01.

Figure 4 shows the performance of the martingales when the examples arrive in the original order: first 7291 of the training set and then 2007 of the test set. The p-values are generated on-line by Algorithm 1 and the two martingales are calculated from the same sequence of p-values. The final value for the simple mixture martingale is 2.0×10^{10}, and the final value for the plug-in martingale is 3.9×10^{8}.

Figure 6 shows the betting functions that correspond to the plug-in martingale and the "best" power martingale. For the plug-in martingale, the function is the estimated probability density function calculated using the whole sequence of p-values. The betting function for the family of power martingale corresponds to the parameter ε^* that provides the largest final value among all power martingales. It gives a clue why we could not see advantages of the new approach for this dataset: both martingales grew up to approximately the same level. There is not much difference between the best betting functions for the old and new methods, and the new method suffers because of its greater flexibility.

4.2. Statlog Satellite dataset

Data The Statlog Satellite dataset (Frank & Asuncion, 2010) consists of 6435 satellite images (divided into 4435 training examples and 2000 test examples). The examples are 3×3 pixel sub-areas of the satellite picture, where each pixel is described by four spectral

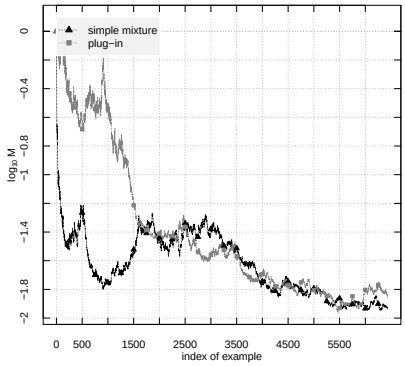

Figure 3. The growth of the martingales for the Statlog Satellite dataset randomly shuffled before on-line testing. The exchangeability assumption is satisfied: the final martingale values are about 0.01.

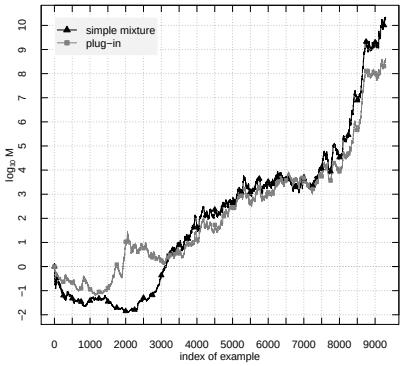

Figure 4. The growth of the martingales for the USPS dataset. For the examples in the original order the exchangeability assumption is rejected: the final martingale values are greater than 3.8×10^8.

values in different spectral bands. Each example is represented by 36 attributes and a label indicating the classification of the central pixel. Labels are numbers from 1 to 7, excluding 6. The testing results are described below.

Figure 3 shows the performance of the martingales for randomly shuffled examples of the dataset. As expected, the martingales do not reject the exchangeability assumption there.

Figure 5 presents the performance of the martingales when the examples arrive in the original order. The final value for the simple mixture martingale is 5.6×10^2 and the final value for the plug-in martingale is 1.8×10^{17}. Again, the corresponding betting functions for the plug-in martingale and the "best" power martingale are presented in Figure 7. For this dataset the generated p-values have a tricky distribution. The family of power betting functions $\varepsilon p^{\varepsilon-1}$ cannot provide a good approximation. The power martingales lose on p-values close to the second peak of the p-values distribution. But the plug-in martingale is more flexible and ends up with a much higher final value.

It can be argued that both methods, old and new, work for the Statlog Satellite dataset in the sense of rejecting the exchangeability assumption at any of the commonly used thresholds (such as 20 or 100). However, the situation would have been different had the dataset consisted of only the first 1000 examples: the final value of the simple mixture martingale would have been 0.013 whereas the final value of the plug-in martingale would have been 3.74×10^{15}.

5. Discussion and conclusions

In this paper we have introduced a new way of constructing martingales for testing exchangeability online. We have shown that for stable sequences of p-values the new more adaptive martingale provides asymptotically the best result compared with any other martingale with a fixed betting function. The experiments of testing two real-life datasets have been presented. Using the same sequence of p-values the plug-in martingale extracts approximately the same amount or more information about the data-generating distribution as compared to the previously introduced power martingales.

Remark. The previous studies were based on the natural idea that lack of exchangeability leads to new examples looking strange as compared to the old ones and therefore to small p-values (for example, if the data-generating mechanism changes its regime and starts producing a different kind of examples). This is, however, a situation where lack of exchangeability makes the p-values cluster around 1: we observe examples that are ideal shapes of several kinds distorted by random noise, and the amount of noise decreases with time. Predicting the kind of a new example using the nonconformity measure (1) will then tend to produce large p-values.

Our goal has been to find an exchangeability martingale that does not need any assumptions about the p-values generated by the method of conformal prediction. Our proposed martingale adapts to the unknown distribution of the p-values by estimating a good bet-

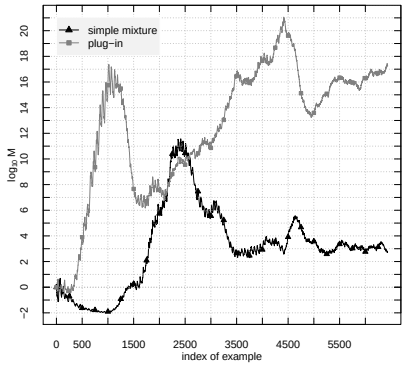

Figure 5. The growth of the martingales for the Statlog Satellite dataset. For the examples in the original order the exchangeability assumption is rejected: the final value of the simple mixture martingale is 5.6×10^2, and the final value of the plug-in martingale is 1.8×10^{17}.

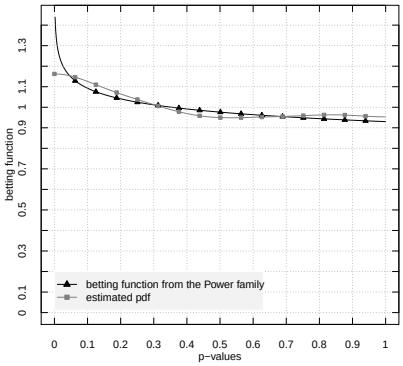

Figure 6. The betting functions for testing the USPS dataset for examples in the original order.

ting function from the past data. This is an example of the plug-in approach. It is generally believed that the Bayesian approach is more efficient than the plug-in approach (see, e.g., Bernardo & Smith, 2000, p. 483). In our present context, the Bayesian approach would involve choosing a prior distribution on the betting functions and integrating the exchangeability martingales corresponding to these betting functions over the prior distribution. It is not clear yet whether this can be done efficiently and, if yes, whether this can improve the performance of exchangeability martingales.

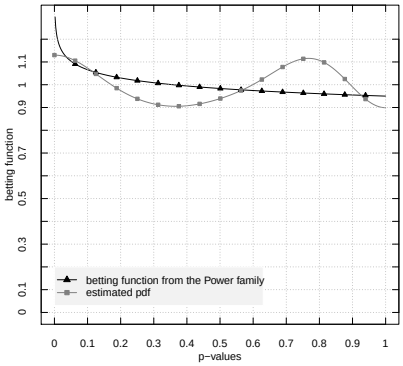

Figure 7. The betting functions for testing the Statlog Satellite dataset for examples in the original order.

Acknowledgments

We are indebted to Royal Holloway, University of London, for continued support and funding. This work has also been supported by the EraSysBio+ grant SHIPREC from the European Union, BBSRC and BMBF and by the VLA grant on machine learning algorithms.

We thank all reviewers for their valuable suggestions for improving the paper.

References

Bernardo, José M. and Smith, Adrian F. M. *Bayesian Theory*. Wiley, Chichester, 2000.

Frank, A. and Asuncion, A. UCI repository, 2010. URL http://archive.ics.uci.edu/ml.

Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag, 2001.

Ho, S.-S. A martingale framework for concept change detection in time-varying data streams. In *Proc. Int. Conf. on Machine Learning (ICML 2005)*, 2005.

Kullback, S. *Information theory and statistics*. Wiley, New York, 1959.

Schervish, M. J. *Theory of statistics*. Springer, New York, 1995.

Vapnik, V. N. *Statistical learning theory*. John Wiley & Sons, Inc., New York, 1998.

Ville, J. *Etude critique de la notion de collectif*. Gauthier-Villars, Paris, 1939.

Vovk, V., Nouretdinov, I., and Gammerman, A. Testing exchangeability on-line. In *Proceedings of the 20th International Conference on Machine Learning (ICML 2003)*, pp. 768–775, Washington, DC, 2003.