A Betting Function for addressing Concept Drift with Conformal Martingales

Charalambos Eliades ST009072@STUD.FREDERICK.AC.CY
Computational Intelligence (COIN) Research Lab
Frederick University, Cyprus
Harris Papadopoulos H.PAPADOPOULOS@FREDERICK.AC.CY
Computational Intelligence (COIN) Research Lab
Frederick University, Cyprus

Abstract

An important issue that appears when using Conformal Martingales (CM) for detecting Concept Drift (CD), is that martingale values get very close to zero when the data generating mechanism remains the same for a large number of instances. In such cases, the martingale takes a long time to recover, resulting in detection delays or even totally failing to detect the occurrence of a CD. To address this issue we propose a new betting function we call Cautious, that avoids betting when there is no evidence that any change is taking place, therefore preventing the continuous reduction of the martingale value. The proposed betting function can be built on top of any existing betting function to mitigate the aforementioned problem. In this work, we combine it with the kernel and histogram betting functions and compare its performance with that of the two original betting functions as well as that of existing methods for addressing CD on five datasets.

Keywords: Conformal, Martingales, Exchangeability,Drift

1. Introduction

In this study we are interested in detecting the point at which CD occurs while performing data stream classification. CD corresponds to a change in the underlying data generating mechanism and therefore results in loss of classification performance. Our approach detects when the data observed so far violates the exchangeability assumption (EA) for a prespecified significance level and retrains the classifier to regain performance.

Formally a CD can be defined as follows: Given a data stream $S = \{(x_0, y_0), (x_1, y_1), \dots, \}$, where x_i is a feature vector and y_i a label; if S can be divided in two sets $S_{0,t} = \{(x_0, y_0), \dots, (x_t, y_t)\}$ and $S_{t+1,\dots} = \{(x_{t+1}, y_{t+1}), \dots\}$, such that $S_{0,t}$ and $S_{t+1,\dots}$ are generated by two different distributions, then a CD occurred at time stamp $t + 1$. This can be extended to any number of CDs with any number of different distributions.

CD can be produced from three sources. Recall that $f_{X,Y,t} = f_{X|Y,t} \cdot f_{Y,t}$ and $f_{X,Y,t+1} = f_{X|Y,t+1} \cdot f_{Y,t+1}$, where $f_{X,Y,t}$ is the joint probability density function of a pair (x, y) at time t . To have a change in the joint distribution of (X, Y) one of the following might have happened: (a) $f_{X,Y,t} = f_{X|Y,t}$ and $f_{Y,t} \neq f_{Y,t+1}$ in this case we have a change in the labels distribution while the decision boundaries remain unchanged, this has also been considered as label shift (Vovk, 2020) and virtual drift (Bagui and Jin, 2020) (b) $f_{X|Y,t} \neq f_{X|Y,t+1}$ and $f_{Y,t} = f_{Y,t+1}$ here the decision boundaries change and lead to decrease in accuracy, it has also been considered as actual drift (Bagui and Jin, 2020) or concept shift (Vovk, 2020)

and (c) $f_{X|Y,t} \neq f_{X|Y,t+1}$ and $f_{Y,t} \neq f_{Y,t+1}$ which is a combination of the two previously mentioned sources.

CD types can be classified in four categories: (a) sudden drift where the data generating mechanism changes instantly (b) gradual drift where the data distribution is replaced with a new one over time (i.e. over time we experience fewer examples belonging to the initial distribution and more belonging to the new distribution), (c) incremental where a new data generating mechanism incrementally replaces the existing mechanism (i.e. each example is generated by a mixture of distributions but over time the impact of the initial distribution disappears) d) reoccurring drift when a previously seen data distribution reappears (Lu et al., 2019),(Bagui and Jin, 2020) .

The CD detection algorithms can be classified in three categories based on the statistics they apply (Lu et al., 2019). The first category is the 'Error rate-based algorithms', which monitor increases or decreases of the online error rate, if these changes are statistically significant a drift alarm is raised. The second and biggest category is the 'Data Distribution-based', here the algorithms quantify the dissimilarity between the historical data and the new data. A distance function is used to measure the dissimilarity between the two distributions and a statistical hypothesis test with respect to a significance level determines if a CD occurs. The last category 'Multiple Hypothesis Test', applies similar techniques with the ones mentioned above, but employ the use of multiple hypothesis tests to determine the presence of a CD. They can be divided in two groups: parallel hypothesis tests and hierarchical multiple hypothesis tests for more information refer to (Lu et al., 2019). In this study our CD detection algorithm belongs to the second category.

In our previous study (Eliades and Papadopoulos, 2021) we examined the use of Inductive Conformal Martingales (ICM) combined with the histogram betting function for detecting violation of the EA and therefore CD. A problem we faced was that if the sequential data distribution remained the same for a large number of instances, then the ICM often got a value close to zero and this increased the chances of a failure to reject the EA when a CD occurred. We tried to address this by not allowing the martingale value decrease below a very small threshold, however this solution resulted in loss of validity. In this work we successfully tackle this problem without violating validity, by proposing a new betting function that avoids betting when there is no evidence that any change is taking place therefore preventing very low martingale values. The proposed betting function, called Cautious, is very flexible in the sense that it can be built on top of any existing betting function to deal with the aforementioned issue. Here we combine it with the histogram betting function proposed in (Eliades and Papadopoulos, 2021) and the Gaussian kernel estimator used in (Fedorova et al., 2012) and (Volkhonskiy et al., 2017).

The rest of the paper starts with an overview of related work on CD in Section 2. Section 3 gives a brief overview of the ideas behind ICM. Section 4 describes the proposed approach and defines the proposed betting function. Section 5 presents the experimental setting and performance measures used in our evaluation and reports the obtained experimental results. Finally, Section 6 gives our conclusions and plans for future work.

2. Related Work

In the literature there are many papers dealing with CD. Due to the large volume of research, we will present only a few works related with the method we follow.

Lu et al. (2019) give an overview of over 130 high quality publications in CD related research areas, they list and discuss 10 popular synthetic datasets and 14 publicly available benchmark datasets used for evaluating the performance of learning algorithms dealing with CD.

Bagui and Jin (2020) surveyed works dealing with CD, they presented a comprehensive study of public synthetic and real datasets that can be used to benchmark such a problem. They review the different types of drifts and approaches that are used to handle such changes in the data.

Wang et al. (2003) proposed a new method called Accuracy Weighted Ensemble (AWE). This ensemble based method assigns weights to the base classifiers based on the classification error.

Kolter and Maloof (2007) presented an ensemble method for CD that creates, weights or removes online learners based on their performance. The authors call their method Dynamic Weighted Majority (DWM) and combine it with Naive Bayes (DWM-NB).

Elwell and Polikar (2011) introduced an ensemble based approach for incremental learning of CD where the underlying data distributions change over time. The proposed algorithm learns from consecutive batches of data without making any assumptions on the nature or rate of drift, a new classifier is trained for each batch of data it receives and combines these classifiers using a dynamically weighted majority voting.

Li et al. (2015) implemented a classification algorithm based on Ensemble Decision Trees for Concept-drifting data streams (EDTC). Extensive studies on synthetic and on real streaming databases demonstrate that the EDTC performs very well compared to several known online algorithms based on single models and ensemble models.

Bu et al. (2017) proposed an incremental least squares density difference (LSDD) change detection method, their method is based on examining the difference between two distributions using two non overlapping windows. They tested their method on 6 synthetic datasets and on one real world dataset.

Fok et al. (2017) used a particle filter-based learning method (PF-LR), for training logistic regression models from evolving data streams. Here the step particles are sampled from the ones that maximize the classification accuracy on the current data batch. Their experiments show that this method gives good performance, even with relatively small batch sizes. They tested the proposed methods on both synthetic and real data sets and find that PF-LR outperforms some other state-of-the-art stream mining algorithms on most of the data sets tested.

Vovk et al. (2003) proposed a way of testing exchangeability in an online manner based on Conformal Prediction (CP) and CM. This technique consists of calculating a sequence of p-values by applying conformal prediction. The p-values are calculated in an online manner where the p-value of each new example is calculated from the new example and the previously seen examples. After the p-values are calculated a betting function is applied on each p-value and the product of the betting function outputs is the value of the Martingale. When the value M of the Martingale, becomes large enough we can reject the exchangeability

assumption at significance level 1/M. This method can be used as a tool for testing if a set of data satisfies the EA and for change point detection in time series and consequently for CD. Some related techniques are presented below.

Ho (2005) implemented a CM based on a simple betting mixture function extending the idea of detecting exchangeability online to detect concept changes in time-varying data streams. Two martingale tests were implemented based on: (i) martingale values and (ii) the martingale difference. The Martingale was calculated using the mixture betting function.

Fedorova et al. (2012) tested the exchangeability of data on two data-sets, USPS and Statlog Satellite data. The data is tested in an online manner i.e. the examples arrive one by one and then the value of the CM is calculated which is a valid measure indicating if the EA should be rejected. In this article the authors used a density estimator of the observed p-values as a betting function. The kernel density estimation has been employed and it has been shown to outperform the simple mixture betting function.

Volkhonskiy et al. (2017) implemented an Inductive version of CM to detect when a change occurs in a time series. In this study the underlying model is trained on the first observations of the time sequence. All the nonconformity scores are calculated via the underlying model. The authors used several betting functions and showed that the pre-computed kernel betting function provides the most efficient results (i.e. lowest mean delay). They tested their methods on synthetic datasets and showed that their results are comparable with those of many other methods such as CUSUM, Shiryaev-Roberts and Posterior Probability statistics.

Ho et al. (2019) proposed a real time novel martingale-based approach driven by Gaussian process regression (GPR) to predict and detect anomalous flight behavior as observations arrive one by one. The authors here use multiple CM tests allowing them to reduce the number of false alarms and also the delay time required for anomaly detection. Here again the Martingale was calculated using the mixture betting function.

3. Inductive Conformal Martingales

In this section we describe the basic concepts of ICM and how our nonconformity scores and p-values are calculated.

3.1. Data Exchangeability

Let $(Z_1, Z_2, \dots)$ be an infinite sequence of random variables. Then the joint probability distribution $\mathbb{P}(Z_1, Z_2, \dots, Z_N)$ (where N is natural number) is exchangeable if it is invariant under any permutation of these random variables. The joint distribution of the infinite sequence $(Z_1, Z_2, \dots)$ is exchangeable if the marginal distribution of $(Z_1, Z_2, \dots, Z_N)$ is exchangeable for every N. Testing if the data is exchangeable is equivalent to testing if it is independent and identically distributed (i.i.d.); this is an outcome of de Finetti's theorem (Schervish, 1995): any exchangeable distribution on the data is a mixture of distributions under which the data is i.i.d.

3.2. Exchangeability Martingale

A test exchangeability Martingale is a sequence of random variables $(S_1, S_2, S_3, \dots)$ being equal to or greater than zero that keep the conditional expectation $\mathbb{E}(S_{n+1}|S_1, \dots, S_n) = S_n$.

To understand how a martingale works consider a fair game where a gambler with infinite wealth follows a strategy that is based on the distribution of the events in the game. The gain acquired by the gambler can be described by the value of a Martingale. I.e. Ville's inequality (Ville, 1939) indicates that the probability to have high profit (C) would be small, $\mathbb{P}\{\exists n : S_n \geq C\} \leq 1/C$.

According to Ville's inequality (Ville, 1939) for the case of the EA a large final value of the Martingale suggests rejection of the assumption with a significance level equal to the inverse of the Martingale value. I.e. a Martingale value such as 10 or 100 rejects the hypothesis of exchangeability at 10% or 1% significance level, respectively.

3.3. Calculating Non-conformity scores and p-values

Let $\{z_1, z_2, \dots\}$ be a sequence of examples, where $z_i = (x_i, y_i)$ with x_i an object given in the form of an input vector, and y_i the label of the corresponding input vector. The CM approach generates a sequence of p-values corresponding to the given sequence of examples and then calculates the martingale as a function of these p-values. As mentioned in Section 1, this work employs the computationally efficient inductive version of CM. ICM uses the first k examples $\{z_1, z_2, \dots, z_k\}$ in the sequence to train a classification algorithm, which it then uses to generate the p-values for the next examples. Consequently, it starts checking for violations of the EA from example z_{k+1} on, i.e. the sequence $\{z_{k+1}, z_{k+2}, \dots\}$.

Our aim is to examine how strange or unusual a new example $z_j \in \{z_{k+1}, z_{k+2}, \dots\}$ is compared to the training examples. To make this possible we define a function $A(z_i, \{z_1, \dots, z_k\})$, where $i \in \{k+1 \dots\}$, called a nonconformity measure (NCM) that assigns a numerical value α_i to each example z_i, called nonconformity score (NCS). The NCM is based on the trained underlying classification algorithm and the bigger the NCS value of an example, the less nonconforming it is with $\{z_1, \dots, z_k\}$ with respect to the underlying algorithm.

For every new example z_j we generate the sequence $H_j = \{\alpha_{k+1}, \alpha_{k+2}, \dots, \alpha_{j-1}, \alpha_j\}$ to calculate its p-value. Note that the NCSs in H_j are calculated when the underlying algorithm is trained on $\{z_1, z_2, \dots, z_k\}$. Given the sequence H_j we can calculate the corresponding p-value (p_j) of the new example z_j with the function:

$$p_j = \frac{|\{\alpha_i \in H_j | \alpha_i > \alpha_j\}| + U_j \cdot |\{\alpha_i \in H_j | \alpha_i = \alpha_j\}|}{j - k}, \tag{1}$$

where α_j is the NCS of the new example and α_i is the NCS of the i^{th} element in the example sequence set and U_j is a random number from the uniform distribution (0,1). For more information refer to Vovk et al. (2003).

3.4. Inductive Conformal Martingales

An ICM is an exchangeability test Martingale (see Subsection 3.2) which is calculated as a function of p-values such as the ones described in Subsection 3.3.

Given a sequence of p-values $(p_1, p_2, \dots)$ the martingale S_n is calculated as:

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

where $f_i(p_i) = f_i(p_i|p_1, p_2, \dots, p_{i-1})$ is the betting function (Vovk et al., 2003).

The betting function should satisfy the constraint: $\int_0^1 f_i(p)dp = 1, f_i(p) \geq 0$ and also the S_n must keep the conditional expectation: $\mathbb{E}(S_{n+1}|S_0, S_1, \dots, S_n) = S_n$.

The integral $\int_0^1 f_i(p)dp$ equals to 1 because $f_i(p)$ is the p-values $(p_1, p_2, \dots, p_{i-1})$ density estimator. We also need to prove that $\mathbb{E}(S_{n+1}|S_0, S_1, \dots, S_n) = S_n$ under any exchangeable distribution.

Proof $\mathbb{E}(S_{n+1}|S_0, S_1, \dots, S_n) = \int_0^1 \prod_{i=1}^n f_i(p_i) \cdot f_{n+1}(p)dp = \prod_{i=1}^n f_i(p_i) \cdot \int_0^1 f_{n+1}(p)dp = \prod_{i=1}^n f_i(p_i) = S_n$ ■

Using equation (2) it is easy to show that $S_n = S_{n-1} \cdot f_n(p_n)$ which allows us to update the martingale online. Let's say that the value of S_n is equal to M then Ville's inequality (Ville, 1939) suggests that we can reject the EA with a significance level equal to $1/M$.

Note that to deal with precision issues we can perform the calculation of equation (2) in the logarithmic scale.

4. Proposed Approach

In this section we describe the proposed approach, which combines ICM with the proposed Cautious betting function. We start by describing the kernel and histogram density estimator before defining the Cautious betting function, which here is built on top of the first two. Finally we explain the process of detecting CD with ICM.

4.1. Previously proposed Betting Functions

4.1.1. Kernel Estimator

This betting function is based on the kernel density estimate (KDE), which is a non parametric method, for approximating the p-value distribution. One drawback of the kernel density estimator is that it is computational expensive. Another drawback is that in some cases the estimation of the optimum bandwidth is a very time consuming task, for this reason we have used Silverman's "rule of thumb" (Silverman, 1986) for bandwidth selection. The KDE will be equal to:

$$\hat{f}_n(x) = \frac{1}{nh} \sum_{i=n-L+1}^{n} k(\frac{|x - x_i|}{h}) \tag{3}$$

where h is a bandwidth parameter and k is the simple Gaussian:

$$k(z) = \frac{1}{\sqrt{2\pi}} \exp(-\tfrac{1}{2}z^2). \tag{4}$$

Note that while calculating the KDE we have used the reflection method as in (Fedorova et al., 2012) to improve performance for points that are near the bounds [0,1]. Also to

eliminate the risk of having an x_0 with $\hat{f}_n(x_0) = 0$ and leading to martingale values equal to zero we add a negligible constant to the $\hat{f}_n(x)$. Because this constant is set to be extremely small (10^{-10}) it does not disturb the performance of KDE. The integral of $A = \int_0^1 (f_n(x) \hat{+} 10^{-10})dx \approx 1$ thus practically there is no need to multiply the integral with any constant to force equality to 1.

4.1.2. Histogram Estimator

Compared to KDE this estimator is faster (Eliades and Papadopoulos, 2021) and needs less computational effort to tune. The p-values $p_i \in [0, 1]$, so we partition $[0, 1]$ into a predefined number of bins k and calculate the frequency of the observations that lie in each bin. Dividing these frequencies by the total number of observations and multiplying it by the number of bins gives us the histogram estimator.

Let us take a fixed number of bins κ this will partition $[0, 1]$ into $B_1 = [0, 1/\kappa)$, $B_2 = [1/\kappa, 2/\kappa)$,..., $B_{\kappa-1} = [(\kappa - 2)/\kappa, (\kappa - 1)/\kappa)$ and $B_\kappa = [(\kappa - 1)/\kappa, 1]$. Then for a p-value $p_n \in B_j$ the density estimator will be equal to:

$$\hat{f}_n(p_n) = \frac{n_j.\kappa}{n-1}, \tag{5}$$

where $n-1$ is the number of p-values seen so far and n_j is the number of p-values belonging to B_j. Note that when n is small it is possible that $\exists x : \hat{f}_n(x) = 0$, in that case until a sufficient number of observations arrives we reduce the number of bins κ by 1, the reduction of κ is repeated until $\nexists x : \hat{f}_n(x) = 0$.

4.2. Improving the betting functions

As we mention in Section 1 an issue of the CM and ICM is that they might need a lot of time to recover from a value very close to zero (Volkhonskiy et al., 2017).

Theorem 1 *When the distribution of the p-values is uniform. for any betting function other than $f = 1$ then $S_\infty \equiv 0$.*

Proof [Proof of Theorem 1]

Let $P \sim U(0, 1)$, we will show that the $\lim_{n\to\infty} S_n = 0$ if the betting function $f \neq 1$.
$S_n = \prod_{i=1}^n f(p_i) \Leftrightarrow ln(S_n) = \ln \prod_{i=1}^n f(p_i) = \sum_{i=1}^n \ln(f(p_i))$
By LLN and Jensen's inequality for a concave function we have that that:
$\lim_{n\to\infty} \sum_{i=1}^{i=n} \frac{\ln(f(p_i))}{n} = \mathbb{E}(ln(f(P))) < \ln \mathbb{E}(f(P)) < 0$
$\mathbb{P}(S_n > \epsilon) = \mathbb{P}(\sum_{i=1}^n \ln(f(p_i)) > \ln(\epsilon))$
$\lim_{n\to\infty} \mathbb{P}(\sum_{i=1}^n \ln(f(p_i)) > \ln(\epsilon)) = \lim_{n\to\infty} \mathbb{P}(\frac{\sum_{i=1}^n \ln(f(p_i))}{n} > \frac{\ln(\epsilon)}{n}) \to 0 \implies S_n \xrightarrow{\mathbb{P}} 0$
But $S_n \xrightarrow{a.s} S_\infty \implies S_n \xrightarrow{\mathbb{P}} S_\infty$ because of the Martingale convergence theorem
By Uniqueness of limit $S_\infty \equiv 0$

■

For this reason we propose a new betting function h_n called **Cautious** which avoids betting (i.e. $h_n = 1$) when not enough evidence is available to reject the EA, thus managing

to keep the value of S_n from getting close to zero and reducing the time needed to detect a CD. The proposed betting function h_n is defined as follows:

$$h_n(x) = \begin{cases} 1 & \text{if } S1_{n-1}/\min_k S1_{n-k} \leq \epsilon \\ f_n & \text{if } S1_{n-1}/\min_k S1_{n-k} > \epsilon \end{cases} \tag{6}$$

with $S1_n = \prod_{i=1}^{n} f_i(p_i)$, $k \in 1, \ldots, W$, $\epsilon > 0$ and $W \in \{1, \ldots, n-1\}$. Here ϵ and W are parameters of the proposed betting function and f_n is a betting function.

Theorem 2 *h_n is a betting function iff f_n is a betting function or equivalently a probability distribution function.*

Proof [Proof of Theorem 2]

h_n is obviously always non negative. We also have to show that it integrates to 1:

$$\int_0^1 h_n(p)dp = \begin{cases} \int_0^1 1dp = 1 & \text{if } S1_{n-1}/\min_k S1_{n-k} \leq \epsilon \\ \int_0^1 f_n(p|p_1, p_2, \ldots, p_{n-1}) = 1 & \text{if } S1_{n-1}/\min_k S1_{n-k} > \epsilon \end{cases} \tag{7}$$

■

As mentioned in Section 3.2 the way a CM works can be parallelized with a gambler whose decisions are based on the distributions of the events in a game. To understand how this betting function works consider two players: The strategy of the first player is based on the density estimator f_n, which in this study is the histogram or kernel estimator. The second player observes how the first player performs in the game at a time window $[n-k, n-1]$, specifically in this time window he calculates the maximum profit that the first player could make if he started playing when $S1_i$ was at its minimum, in other words he calculates $S1_{n-1}/\min_k S1_{n-k}$. If the first player has losses or very small profit then player two will not bet i.e. he will set $h_n = 1$, if player one could make profit greater than a threshold then player two will bet i.e. he will set $h_n = f_n$. Note that the proposed betting function can be combined with any statistical test of uniformity i.e. Kolmogorov Smirnoff test (KST), or be extended to more than one players.

4.3. Detecting CD using ICM

To reject the EA and thus detecting CD for a pre-specified significance level δ the value of the Martingale must exceed $1/\delta$. In Algorithm 1 we summarize this process. Therefore if the value of the Martingale S_k at point k exceeds 100 then a CD is detected with a significance level of 1%. Note that in this work when we use only one player strategy the betting function is equal to a density estimator $B_n = \hat{f}_n$ (betting functions mentioned at Section 4.1), while when we use two players the betting function $B_n = h_n$ and is calculated according to formula (6).

Data: Training set $\{z_1, z_2, \ldots, z_k\}$, Test set$\{z_{k+1}, \ldots, z_n\}$, significance level δ
Initialize $S_1 = 1$
for *i=1,...,n-k* **do**
  $\alpha_i = A(z_{k+i}, \{z1, \ldots, z_k\})$
  $p_i = \frac{\#\{j:\alpha_j > \alpha_i\} + U_j \#\{j:\alpha_j = \alpha_i\}}{i}$
  Calculate betting function $B_i = B(p_1, \ldots, p_{i-1})$
  $S_i = S_{i-1} \cdot B_i(p_i)$
  if $S_i > \frac{1}{\delta}$ **then**
    Raise an Alarm
  end
end
Algorithm 1: Detect CD using ICM

5. Experiments and Results

This section evaluates experimentally the performance of the proposed approach and compares it to that of existing approaches. In the first set of experiments we examine in detail the performance improvement of the Cautious betting function using a synthetic dataset generated for this purpose. Specifically we compare the Cautious betting function recovery time with that of the gaussian kernel and histogram betting functions.

In the second set of experiments we compare the performance of the Cautious betting function with the histogram and the kernel ones using two publicly available synthetic datasets (STAGGER,SEA), while in our third set of experiments we compare the three betting functions on two publicly available real datasets (ELEC,AIRLINES).

For the four publicly available datasets we compare the proposed approach with two state-of-the-art methods from the literature: the DWM method (Kolter and Maloof, 2007) and the AWE method (Wang et al., 2003). Note that our main aim in this study is to improve the recovery time of existing betting functions.

5.1. Datasets

5.1.1. Recovery time dataset

This dataset was constructed it in a way that responds to the needs of our first set of experiments, i.e. examining the martingale recovery time. It consists of 100100 instances described by one numeric variable in the range [0,1] and a binary label. For the first 10100 instances the label is True if the numeric input is higher than 0.5, otherwise the label is False, this is our first concept. After the 10100th observation, CD starts occurring. The remaining 90000 instances have a True label if their numeric input is higher than 0.55, otherwise the label is set to False, this is our second concept. In our experiments on this dataset we set the training set size to 100 instances.

Although this synthetic dataset consists of only one variable, it is constructed in such way that when the CD occurs the distance between the decision boundaries is very small making it hard to detect the CD. Furthermore, it highlights some important issues of ICM when combined with the kernel and histogram betting functions.

5.1.2. Synthetic Benchmark Datasets

The STAGGER (Schlimmer and Granger, 1986) is a standard benchmark dataset widely used for evaluating CD detection. For our simulations we have generated 1000000 instances, where each example is described by 3 categorical attributes and has binary target output. The drift type here is sudden while there are 4 different concepts, a drift occurs every 10000 examples (i.e. each chunk consists of 10000 examples). In our experiments on this dataset we set the training set size equal to 200. Here the training set size compared to the number of instances before a drift occurs is very small, specifically it is 2% of the total number of instances of each chunk.

The SEA (Street and Kim, 2001) dataset is a popular synthetic dataset that contains sudden CD. For our simulations we have generated 1000000 instances, where each example is described by 3 numeric attributes and has a binary label. There are 4 concepts and a drift occurs every 250000 examples (i.e. each chunk consists of 250000 examples). In our experiments the classifier training set size is set to 1000. In this dataset the three variables take random values in the range [0-10] and if the sum of the first two variables is less than or equal to a pre-specified threshold then the instance is assigned to class 1 otherwise to 0, the third variable is irrelevant. Also compared to the number of observations before a CD occurs this training size is negligible. Note that here we test the transition from concept $a \rightarrow b \rightarrow c \rightarrow d$ while in the case of STAGGER we test the transition from concept $a \rightarrow b \rightarrow c \rightarrow d \rightarrow a, \ldots$. Note that in this study we also inject noise in both datasets by changing 10% of the labels and examine the performance of our approach with and without noise. Also note that when a CD is detected we wait for enough new observations for formulating an equally sized training set with the initial one before retraining the model and restarting to apply Algorithm 1.

5.1.3. Real World Benchmark Datasets

The ELEC dataset (Harries et al., 1999) is a time series containing 45312 instances recorded at 30-min intervals with a binary class label (identifying if we have a rise or a drop in price compared with a moving average of the last 24 hours). Each example consists of eight variables. The data has been collected from the Australian New South Wales Electricity Market. In our experiments we have excluded the variables time, date, day, and period, so we only used nswprice, nswdemand, transfer, vicprice, and vicdemand. The training set size was set to 300, corresponding to trying to predict future values using as training set the observations of about for less than 1 week.

The Airlines dataset (Ikonomovska E, 2010) is from the Data Expo Competition 2009. The data has been collected from USA airports and consists of flight arrival and departure records of commercial flights within the USA from October 1987 to April 2008. It contains 539383 instances each described by 7 features. The purpose is to classify if a flight was Delayed or Not delayed. In our experiments we have used a training set size of 200 observations.

5.2. Performance Measures

To evaluate the performance of the proposed approach we consider four measures (some of these are not applicable in all cases):

(a) Accuracy: Average accuracy of the classifier (excluding the training set).

(b) Mean delay: Average number of observations before detecting a CD after it has occurred.

(c) True alarm rate(TAR): Average rate of CDs that have been correctly detected per chunk.

(d) False alarm rate(FAR): Average rate of CDs erroneously detected per chunk.

5.3. Results

In this section we present the experimental results of the three sets of experiments we performed. For the four benchmark datasets we used a tree-bagger (with 40 trees) while for the recovery time dataset we used a simple tree classifier. For each example the tree-bagger or tree classifier will output the posterior probability $\tilde{p}_j$ for the true label y_j. Therefore we define the NCM: $\alpha_j = -\tilde{p}_j$.

For all of our experiments we calculate the histogram density estimator using the p-values of the last $L = 1000$ observations instead of $L = 5000$ as in (Eliades and Papadopoulos, 2021), this was done in order to accelerate the process of calculating f_n since 1000 p-values seem sufficient for estimating the p-value density function. Additionally when a CD occurs our density estimator will be able to adapt to the new p-value distribution faster.

When we use the KDE we set $L = 500$ on the recovery time dataset, while for the rest of our experiments we set $L = 100$ as in (Volkhonskiy et al., 2017).

In all experiments the parameters ϵ and W of the Cautious betting function were set to 100 and 5000 respectively, although we believe that a very high value of W would also work.

In subsection 5.3.1 we perform simulations using the recovery time dataset. Our aim here is to compare the time needed of the Cautious betting function to recover when a CD occurs with that of the kernel and the histogram functions. For this experiment we have performed 10 simulations and average the results.

In the remaining two subsections we perform simulations on the two Synthetic and two Real world datasets. We focus on CD detection and model retraining to regain accuracy. The presented results are averaged over 5 simulations on each dataset. For all experiments with Algorithm 1 we set $\delta = 100$.

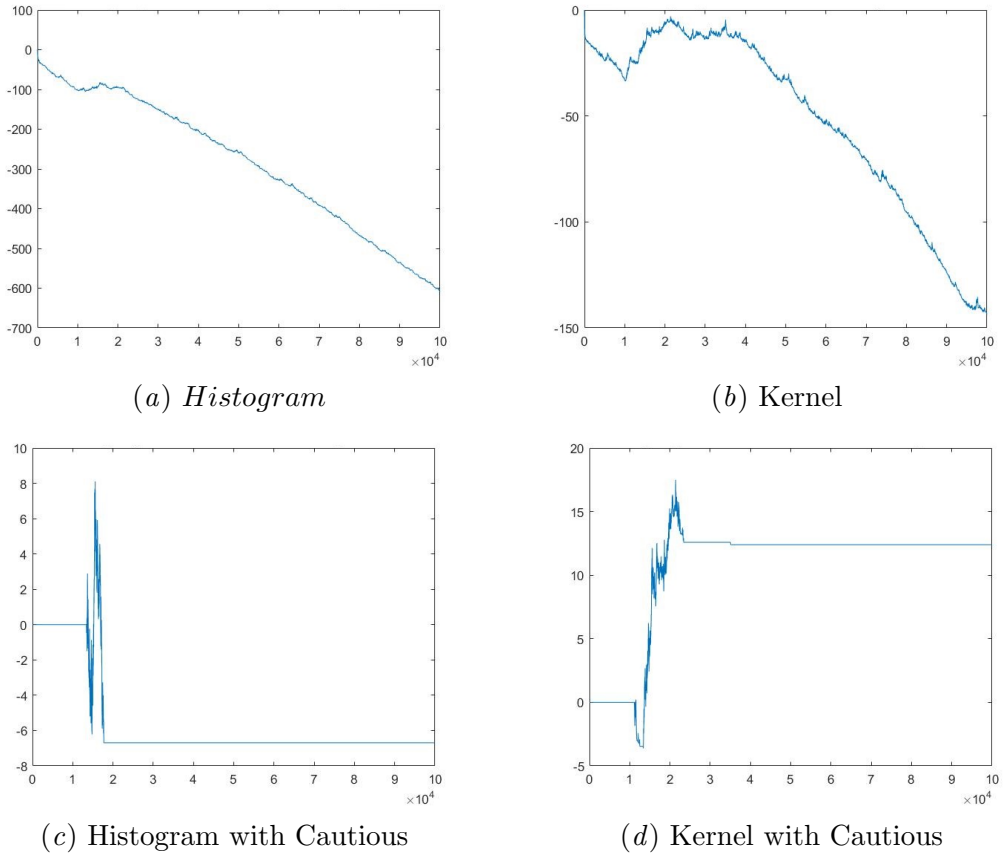
We empirically demonstrate that the use of the Cautious betting function keeps the number of false alarms low and accelerates the detection of CD. We also compare the accuracy of the proposed approach with that of two state-of-the-art algorithms: AWE and DWM-NB described in Section 2. The accuracies of these two algorithms were obtained from Sarnovsky and Kolarik (2021).

5.3.1. Betting Function recovery time comparison

In this set of experiments we use the histogram, kernel and Cautious (combined with kernel and histogram) betting functions to calculate the value of the martingale S_n on the recovery time dataset.

We have trained a decision tree on the first 100 observations and evaluated CD detection performance on the rest.

Figure 1 presents the S_n growth calculated by equation (2).



(*a*) *Histogram* (*b*) Kernel

(*c*) Histogram with Cautious (*d*) Kernel with Cautious

Figure 1: $log(S_n)$ growth for Recovery time dataset

To give an explanation of what Figure 1 presents we have to keep in mind that the first 10000 NCSs correspond to observations belonging to the first concept and the remaining NCSs correspond to observations that belong to the second concept. Recall from equation (1) how the p-values are calculated, the H_i sequence consists of $\{\alpha_{101}, \alpha_{102}, \ldots, \alpha_{i-1}, \alpha_i\}$ NCSs. If $101 \leq i \leq 10100$ the H_i sequence will contain NCSs of instances generated by the first concept, on the other hand when $i > 10100$ the H_i sequence consists of NCSs belonging to the first and the second concept. Thus as observations arrive with $i > 10100$ the H_i sequence will be a mixture of distributions and as i increases the impact of the first concept on the p-value calculation is reduced.

In Subfigures 1(*a*) and 1(*b*) we present the resulting Martingales using the histogram and kernel betting function respectively, as we can see at time $t = 10100$ the Martingale switches from a decreasing to an increasing behavior, but unfortunately it fails to recover to a big value and reject the EA. As more observations arrive the martingale starts to decrease, this is not surprising. Recall from equation (1) how the p-values are calculated, before the 10101^{th} observation the H_i sequence contains only NCSs of instances generated by the first concept, then as more observations arrive the H_i sequence will contain NCSs generated by the first and the second concept, thus H_i will be a mixture of the two distributions.

As observations increase the 'impact' of the first distribution will become smaller, thus the produced p-values will be asymptotically uniform making it even harder for the ICM to detect a CD. The fact that a uniform or a close to uniform distribution results in the martingale value having a decreasing behavior suggests the need for a mechanism that will prevent the martingale from getting a low value thus enabling the quick detection of CD.

In Subfigures 1(c) and 1(d) we show the resulting Martingales using the Cautious betting function. Notice that after time $t = 10100$ our strategy switches from no betting to betting. After a constant behavior started to fluctuate and then switches to an increasing behavior and finally manages to detect the CD with a significance level much smaller than 1%. When the martingale reaches its maximum value it starts to reduce and remains stable because the betting function switches to a no bet strategy ($h = 1$), this is an outcome of the fact that the resulting p-values will be asymptotically uniform or close to uniform for the reasons explained above.

Table 1 compares the performance of the four betting functions. As we can see when we combine the Cautious betting function with the kernel and histogram, the TAR increases. Specifically, the histogram betting function fails to detect any CD but when we combine it with the Cautious betting function the TAR increases to 0.7. The kernel betting function was able to detect 30% of the CDs but combined with the Cautious betting function managed to reach a TAR of 0.8. Generally speaking the Cautious betting function manages to increase the TAR and reduce the mean delay while keeping the FAR equal to zero.

Table 1: RECOVERY TIME DATASET

Betting Function	**FAR**	**TAR**	**Mean Delay**
Histogram	0	0	—
Kernel	0	0.3	14835
Cautious with Histogram	0	0.7	12335
Cautious with Kernel	0	0.8	12339

Note that in this experiment while calculating the kernel density estimator we used a frame window of 500 p-values, while for the rest of our experiments we used 100 p-values only. When we use 500 p-values the kernel density will be able to detect CD while when we use 100 p-values unfortunately it will fail even if we combine it with the Cautious betting function. A possible explanation is that it the kernel density estimator has reduced performance near the bounds and we therefore need to fine tune the bandwidth, which is a very computationally intensive task.

5.3.2. Synthetic Datasets

In this subsection, we present the STAGGER and the SEA dataset results while using the four different betting functions. In Table 2 we show the results while performing simulations on the noise free STAGGER dataset. In this dataset, when we use the histogram betting function, as the number of bins increases, the time required to detect a CD increases and consequently accuracy reduces. This is due to the fact that when we have long time intervals with no concept change, the martingale value will tend to zero, thus the time needed to

recover increases. When we combine the histogram estimator with the Cautious betting function, we can see that in most cases all performance measures have been dramatically improved, also we can observe that when the number of bins increases the mean delay and consequently the accuracy improves, this is an outcome of the betting strategy we use which avoids betting when not necessary. The kernel betting function performs well on the STAGGER dataset, specifically it outperforms the histogram betting function, but it does not perform better than the Cautious combined with the histogram. If we combine the kernel with the Cautious betting function it performs well but not better than the Cautious combined with a 15 bin histogram.

In all cases the TAR is 100% and the FAR is very low. The best CD detection time is obtained when we use the Cautious betting function combined with a 15 bin histogram estimator.

Table 2: STAGGER (0% noise)

Betting function	**No of Bins**	**Accuracy**	**Mean delay**	**TAR**	**FAR**
Histogram	5	0.98661	223.2	1	0.008
Histogram	10	0.98621	228.5	1	0.004
Histogram	15	0.98559	240.5	1	0.008
Histogram with Cautious	5	0.99287	118.6	1	0
Histogram with Cautious	10	0.99509	96.8	1	0.002
Histogram with Cautious	15	0.99587	69.6	1	0
Kernel	—	0.99220	136.7	1	0.008
Kernel with Cautious	—	0.99720	77.9	1	0.004

The simulated results on the STAGGER dataset with 10% noise are presented in Table 3. Here the betting functions have similar behavior to the noise free experiment. In this case, because the dataset is noisy, the accuracy is reduced by about 5%. When we use the histogram betting function the mean delay increases as the number of bins increases, but when we combine it with the Cautious betting function the mean delay decreases. When the kernel betting function is used, it performs better than the histogram but not better than the Cautious combined with the histogram. When we combine the kernel with the Cautious betting function it outperforms all other betting functions. In all cases the TAR is equal to 100% and the FAR is close to zero. The best CD detection time is obtained when we use the Cautious betting function combined with the Kernel.

Table 4 presents our results on the noiseless SEA dataset. Unfortunately some of our simulations on the SEA dataset show poor performance, this is due to the fact that the concept durations are too long and the Martingale fails to recover. The histogram betting function with a number of bins equal to 5 and 10 fails to detect any CD. When we set the number of bins equal to 15 most of the CDs are detected but with a big delay. When we combine the Cautious betting function with the histogram estimator and a number of bins equal to 10 or 15 all CDs can be detected and we have no false alarms. The kernel betting function fails to detect any CDs even if we combine it with Cautious betting function. The best mean delay is obtained when we combine the Cautious betting function with a 15 bin histogram estimator.

Table 3: STAGGER (10% noise)

Betting function	**No of Bins**	**Accuracy**	**Mean delay**	**TAR**	**FAR**
Histogram	5	0.93690	227.6	1	0.002
Histogram	10	0.93604	236.7	1	0.004
Histogram	15	0.93535	250.8	1	0.008
Histogram with Cautious	5	0.94225	86.4	1	0.002
Histogram with Cautious	10	0.94440	77.2	1	0
Histogram with Cautious	15	0.94438	70.2	1	0
Kernel	—	0.94290	113.8	1	0.002
Kernel with Cautious	—	0.94550	48.4	1	0

Table 4: SEA (0% noise)

Betting Function	**No of Bins**	**Accuracy**	**Mean delay**	**TAR**	**FAR**
Histogram	5	0.92110	-	0	0
Histogram	10	0.92080	-	0	0
Histogram	15	0.96867	6497.1	0.996	0
Histogram with Cautious	5	0.92090	-	0	0
Histogram with Cautious	10	0.98290	1367.2	1	0
Histogram with Cautious	15	0.98190	830.5	1	0
Kernel	—	0.92060	-	0	0
Kernel with Cautious	—	0.92100	-	0	0

The results shown in Table 5 are obtained while performing simulations on the SEA dataset with 10% noise. As in the case of the noiseless dataset our experiments here indicate poor performance i.e. failure to raise an alarm when a CD occurs, for the same reason explained above. The histogram betting function with a number of bins equal to 5 and 15 fails to detect any CD. When we set the number of bins equal to 10 most of the CD are detected but with a big delay. When we set the number of bins to 10 or 15 on the histogram estimator and combine it with the Cautious betting function all CDs can be detected and we have no false alarms. The kernel betting function fails to detect any CDs but when we combine it with Cautious betting function it manages to detect all CDs. The best CD detection time is obtained when we combine the Cautious betting function with a 15 bin histogram.

Table 5: SEA(10% noise)

Betting Function	**No of Bins**	**Accuracy**	**Mean delay**	**TAR**	**FAR**
Histogram	5	0.8610	-	0	0
Histogram	10	0.8809	6023.6	0.998	0
Histogram	15	0.8606	-	0	0
Histogram with Cautious	5	0.8606	-	0	0
Histogram with Cautious	10	0.9149	1367.2	1	0
Histogram with Cautious	15	0.9140	441.1	1	0
Kernel	—	0.8607	-	0	0
Kernel with Cautious	—	0.9151	518.3	1	0

5.3.3. Real Datasets

In this subsection, we present the simulated results for the ELEC and the AIRLINES datasets while using the four different betting functions. In Table 6 we show the results obtained while performing simulations on the ELEC dataset. Here when using the histogram betting function the number of bins does not play a significant role in the improvement of accuracy, but when the histogram estimator is combined with the Cautious betting function the accuracy improves because the CM does not get a value close to zero, allowing it to recover faster and thus reject the EA. In this dataset the kernel betting function provides an accuracy comparable with that of the Cautious combined with the histogram, a possible explanation is that the time duration of the different concepts is short, thus the CM does not get values close to zero and can recover fast. When we combine the kernel with the Cautious betting function the accuracy improves and the number of detected CDs increases. This is because the Cautious betting function allows the CM to recover faster and more CDs are detected without false alarms which allows model retraining . The best accuracy is achieved when we use the Cautious betting function combined with the kernel density estimator i.e. 75.93%.

In Table 7 we show our results while performing simulations on the AIRLINES dataset. The histogram betting function with a number of bins equal to 5 performs better. When ce combine the histogram estimator with the Cautious betting function the accuracy improves

Table 6: ELEC

Betting function	**No of Bins**	**Accuracy**	**Number of CD detected**
Histogram	5	0.73470	116.2
Histogram	10	0.73711	103.8
Histogram	15	0.73472	83.6
Histogram with Cautious	5	0.75429	102.2
Histogram with Cautious	10	0.75101	107.4
Histogram with Cautious	15	0.74836	108.8
Kernel	—	0.75178	128.8
Kernel with Cautious	—	0.75929	146.2

as the number of bins increases. When the kernel betting function is used the accuracy is comparable to that obtained in the histogram betting function. If we combine the Cautious betting function with the kernel the accuracy further improves. A possible explanation is that in the AIRLINES dataset a CD appears after a long time, thus when more bins or the kernel betting function are used the Martingale gets a value close to zero making it more difficult to recover resulting in a decrease in accuracy. This decrease is correlated with the number of CDs detected. On the other hand the use of the Cautious betting function increases the number of the detected CDs, the fact that the accuracy increases as the number of CDs increases suggests that the raised alarms are not false, thus model retraining is performed when necessary to recover accuracy. Our simulations on the AIRLINES dataset indicate that the Cautious betting function combined with the kernel density estimator achieved the best accuracy of 60.18%.

Table 7: AIRLINES

Betting Function	**Number of bins**	**Accuracy**	**Number of cd detected**
Histogram	5	0.57177	21.2
Histogram	10	0.56229	5.8
Histogram	15	0.55354	3
Histogram with Cautious	5	0.57377	39.6
Histogram with Cautious	10	0.59633	59.6
Histogram with Cautious	15	0.59088	51
Kernel	—	0.56597	11.8
Kernel with Cautious	—	0.60184	71.4

5.4. Discussion

In all cases the Cautious betting function outperforms the previously proposed betting functions in all performance measures. Specifically the FAR is reduced, the TAR is increased and also the mean delay is reduced . This improvement allows model retraining when a CD

occurs and therefore leads to a higher accuracy. Table 8 compares the best betting function of our simulations for each dataset with two state of the art algorithms in terms of accuracy.

As can be seen from the results presented in this table, the proposed approach has similar accuracy to the two state-of-the-art approaches, in all cases, while it outperforms both in the case of the SEA dataset.

Table 8: Comparison of Cautious betting function with state of the art methods

Dataset	**CAUTIOUS**	**AWE**	**DWE-NB**
STAGGER	0.946	0.948	0.901
SEA	0.915	0.879	0.876
ELEC	0.759	0.756	0.800
AIRLINES	0.602	0.618	0.640

6. Conclusions

We have proposed a novel betting function that is built on top of existing betting functions and greatly reduces the time needed to detect a CD, especially when the change occurs after a long time. Our betting function manages to prevent the martingale from getting a very low value. This is done by switching to a no bet strategy when there is no evidence that any change is taking place. We empirically demonstrated that the Cautious betting function reduces the time needed to detect a CD, while in the case of real datasets it provides an improvement in accuracy. Furthermore, the performance achieved is close to, and in some cases even better than, that of two state-of-the-art CD detection techniques. Based on our experimental results, we believe that the proposed approach is quite promising for addressing CD, especially since there is still room for further improvements. Thus our future plans include examining the combination of the Cautious betting function with more than one uniformity tests as well as employing strategies for selecting a representative training set for the underlying classifier.

References

Sikha Bagui and Katie Jin. A survey of challenges facing streaming data. *Transactions on Machine Learning and Artificial Intelligence*, 8(4):63–73, Aug. 2020. doi: 10.14738/tmlai.84.8579. URL https://journals.scholarpublishing.org/index.php/TMLAI/article/view/8579.

Li Bu, Dongbin Zhao, and Cesare Alippi. An incremental change detection test based on density difference estimation. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(10):2714–2726, 2017. doi: 10.1109/TSMC.2017.2682502.

Charalambos Eliades and Harris Papadopoulos. Using inductive conformal martingales for addressing concept drift in data stream classification. In Lars Carlsson, Zhiyuan Luo, Giovanni Cherubin, and Khuong An Nguyen, editors, *Proceedings of the Tenth Symposium on Conformal and Probabilistic Prediction and Applications*, volume 152 of

Proceedings of Machine Learning Research, pages 171–190. PMLR, 08–10 Sep 2021. URL https://proceedings.mlr.press/v152/eliades21a.html.

Ryan Elwell and Robi Polikar. Incremental learning of concept drift in nonstationary environments. *IEEE Transactions on Neural Networks*, 22(10):1517–1531, 2011. doi: 10.1109/TNN.2011.2160459.

Valentina Fedorova, Alex Gammerman, Ilia Nouretdinov, and Vladimir Vovk. Plug-in martingales for testing exchangeability on-line. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, page 923–930, Madison, WI, USA, 2012. Omnipress. ISBN 9781450312851.

Ricky Fok, Aijun An, and Xiaogang Wang. Mining evolving data streams with particle filters. *Computational Intelligence*, 33(2):147–180, 2017. doi: https://doi.org/10.1111/coin.12071. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/coin.12071.

Michael Harries, U Nsw cse tr, and New South Wales. Splice-2 comparative evaluation: Electricity pricing. Technical report, 1999.

Shen-Shyang Ho. A martingale framework for concept change detection in time-varying data streams. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML 05, page 321–327, New York, NY, USA, 2005. Association for Computing Machinery. ISBN 1595931805. doi: 10.1145/1102351.1102392. URL https://doi.org/10.1145/1102351.1102392.

Shen-Shyang Ho, Matthew Schofield, Bo Sun, Jason Snouffer, and Jean Kirschner. A martingale-based approach for flight behavior anomaly detection. In *2019 20th IEEE International Conference on Mobile Data Management (MDM)*, pages 43–52, 2019. doi: 10.1109/MDM.2019.00-75.

Dveroski S Ikonomovska E, Gama J. Learning model trees from evolving data streams. *Data Mining and Knowledge Discovery*, 23:128–168, 2010.

J. Zico Kolter and Marcus A. Maloof. Dynamic weighted majority: An ensemble method for drifting concepts. *J. Mach. Learn. Res.*, 8:2755–2790, December 2007. ISSN 1532-4435.

Peipei Li, Xindong Wu, Xuegang Hu, and Hao Wang. Learning concept-drifting data streams with random ensemble decision trees. *Neurocomputing*, 166:68–83, 2015. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2015.04.024. URL https://www.sciencedirect.com/science/article/pii/S0925231215004713.

Jie Lu, Anjin Liu, Fan Dong, Feng Gu, João Gama, and Guangquan Zhang. Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering*, 31(12):2346–2363, 2019. doi: 10.1109/TKDE.2018.2876857.

Martin Sarnovsky and Michal Kolarik. Classification of the drifting data streams using heterogeneous diversified dynamic class-weighted ensemble. *PeerJ Computer Science*, 7, 04 2021. doi: 10.7717/peerj-cs.459.

Mark J. Schervish. *Theory of Statistics*. Springer, New York, 1995.

Jeffrey C. Schlimmer and Richard H. Granger. Incremental learning from noisy data. *Mach. Learn.*, 1(3):317–354, March 1986. ISSN 0885-6125. doi: 10.1023/A:1022810614389. URL https://doi.org/10.1023/A:1022810614389.

Bernard W Silverman. *Density estimation for statistics and data analysis*. Chapman and Hall, London, 1986. URL https://cds.cern.ch/record/1070306.

W. Nick Street and YongSeog Kim. A streaming ensemble algorithm (sea) for large-scale classification. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '01, page 377–382, New York, NY, USA, 2001. Association for Computing Machinery. ISBN 158113391X. doi: 10.1145/502512.502568. URL https://doi.org/10.1145/502512.502568.

J. Ville. Étude critique de la notion de collectif. by j. ville. pp. 144. 75 francs. 1939. monographies des probabilités, calcul des probabilités et ses applications, publiées sous la direction de m. Émile borel, fascicule iii. (gauthier-villars, paris). *The Mathematical Gazette*, 23(257):490–491, 1939. doi: 10.2307/3607027.

Denis Volkhonskiy, Evgeny Burnaev, Ilia Nouretdinov, Alexander Gammerman, and Vladimir Vovk. Inductive conformal martingales for change-point detection. In Alex Gammerman, Vladimir Vovk, Zhiyuan Luo, and Harris Papadopoulos, editors, *Proceedings of the Sixth Workshop on Conformal and Probabilistic Prediction and Applications*, volume 60 of *Proceedings of Machine Learning Research*, pages 132–153, Stockholm, Sweden, 13–16 Jun 2017. PMLR. URL http://proceedings.mlr.press/v60/volkhonskiy17a.html.

Vladimir Vovk. Testing for concept shift online, 2020.

Vladimir Vovk, Ilia Nouretdinov, and Alexander Gammerman. Testing exchangeability on-line. pages 768–775, 01 2003.

Haixun Wang, Wei Fan, Philip S. Yu, and Jiawei Han. Mining concept-drifting data streams using ensemble classifiers. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, page 226–235, New York, NY, USA, 2003. Association for Computing Machinery. ISBN 1581137370. doi: 10.1145/956750.956778. URL https://doi.org/10.1145/956750.956778.