# On the Classification of Dynamical Data Streams Using Novel "Anti-Bayesian" Techniques[*]

Hugo Lewi Hammer[†], Anis Yazidi[‡], B. John Oommen[§]

**Abstract**

The classification of dynamical data streams is among the most complex problems encountered in classification. This is, firstly, because the distribution of the data streams is non-stationary, and it changes without any prior "warning". Secondly, the manner in which it changes is also unknown. Thirdly, and more interestingly, the model operates with the assumption that the correct classes of previously-classified patterns become available at a juncture after their appearance. This paper pioneers the use of unreported novel schemes that can classify such dynamical data streams by invoking the recently-introduced "Anti-Bayesian" (AB) techniques. Contrary to the Bayesian paradigm, that compare the testing sample with the distribution's central points, AB techniques are based on the information in the distant-from-the-mean samples.

Most Bayesian approaches can be naturally extended to dynamical systems by dynamically tracking the mean of each class using, for example, the exponential moving average based estimator, or a sliding window estimator. The AB schemes introduced by Oommen *et al.*, on the other hand, work with a radically different approach and with the non-central *quantiles* of the distributions. Surprisingly and counter-intuitively, the reported AB methods work equally or close-to-equally well to an optimal supervised Bayesian scheme on a host of accepted Pattern Recognition problems. This thus begs its natural extension to the unexplored arena of classification for dynamical data streams. Naturally, for such an AB classification approach, we need to track the non-stationarity of the *quantiles* of the classes. To achieve this, in this paper, we develop an AB approach for the online classification of data streams by applying the efficient and robust quantile estimators developed by Yazidi and Hammer [12, 37].

[*]A very preliminary version containing a few of the initial results found in this paper will be presented at the CEC'2017, the 2017 IEEE Congress on Evolutionary Computation, Donostia, Spain, in June 2017 [11]. The last author is grateful for the partial support provided by NSERC, the Natural Sciences and Engineering Research Council of Canada. We are extremely grateful to the anonymous Referees of the earlier version of this paper for their feedback. Their input significantly improved the quality of this current version.

[†]Author's status: *Associate Professor*. This author can be contacted at: Oslo and Akershus University College, Department of Computer Science, Pilestredet 35, Oslo, Norway. E-mail: `hugo.hammer@hioa.no`.

[‡]Author's status: *Associate Professor*. This author can be contacted at: Oslo and Akershus University College, Department of Computer Science, Pilestredet 35, Oslo, Norway. E-mail: `anis.yazidi@hioa.no`.

[§]*Chancellor's Professor* ; *Fellow: IEEE* and *Fellow: IAPR*. This author can be contacted at: School of Computer Science, Carleton University, Ottawa, Canada : K1S 5B6. This author is also an *Adjunct Professor* with the University of Agder in Grimstad, Norway. E-mail address: `oommen@scs.carleton.ca`.

Apart from the methodology itself, in this paper, we compare the Bayesian and AB approaches using both real-life and synthetic data. The results demonstrate the intriguing and counter-intuitive results that the AB approach, sometimes, actually *outperforms* the Bayesian approach for this application both with respect to the peak performance obtained, and the robustness of the choice of the respective tuning parameters. Furthermore, the AB approach is much more robust against outliers, which is an inherent property of *quantile* estimators [12, 37], which is a property that the Bayesian approach cannot match, since it rather tracks the mean.

Keywords: *Anti-Bayesian Classification, Data Streams, Classification With delay, Incremental Quantile Estimation*

## 1 Introduction

### 1.1 Problem Statement

**The Pertinence of Data Streams**: Traditionally, Machine Learning (ML) methods are assumed to deal with static data stored in memory, which can be read several times. On the contrary, streaming data grows at an unlimited rate and arrives continuously in a single-pass manner that can only be read once. Further, there are space and time restrictions in analyzing streaming data. Consequently, one needs methods that are "automatically adapted" to update the training models based on the information gathered over the past observations whenever a change in the data is detected. In addition, a typical challenge in analyzing data streams is that the properties of the stream varies dynamically with time, where traditional static analysis tools cannot be applied.

The classification of such dynamical data streams is among the most complex problems encountered in Pattern Recognition (PR) and ML. This is primarily because the data stream's class conditional distribution is non-stationary. It changes to a new unknown distribution, i.e., the distribution of the new stream, without any indication that such a switch is going to occur. And the most interesting facet of this is that the model operates with the assumption that the correct classes of previously-classified patterns become available at a juncture after their initial appearance.

This scenario is more pertinent today than ever before. Indeed, in the past few years, due to the advances in computer hardware technology, large amounts of data have been generated and collected and are stored permanently from different sources. Some of the applications that generate data streams are financial tickers, log records or click-streams in web tracking and personalization, data feeds from sensor applications, and call detail records in telecommunications. Furthermore, data streams could be social media feeds from Twitter or online news, network data, economic or environmental data etc. The analysis of these data streams has received a lot of attention in the literature [19] and is considered as one of the most important challenges in the field of ML and PR.
**The Bayesian ML of Data Streams**: Almost all traditional classification techniques reported to-

date depend, either directly or implicitly, on the Bayesian principle which yields optimal classification rules. To be more specific, within a Bayesian paradigm, if one has to classify a testing sample by resorting to *a single* point in the feature space from each class, the *optimal* Bayesian strategy would be to achieve this based on the "distance" (for example, Euclidean or Mahalonabis) from the corresponding means or *central* points in the respective distributions. In this vein, in order to deal with the challenges that pertain to data streams, a large body of studies have focused on the idea of summarizing the characteristics of a data stream by rather tracking the properties of the stream, like its distributional moments (expectation, variance, skewness, kurtosis etc) or quantiles [12, 37]. In fact, those quantities are usually easy to compute in a incremental manner and can serve as a "footprint" of the data stream in question, whence the classification can be achieved.

From the above, the informed reader can observe that any classical ML or PR task can be re-written as a new problem within the framework of analyzing data streams. Typical examples include that of assigning arriving data samples to one of a set of classes, or to a cluster, where the true class (cluster) label is revealed subsequently – with a certain time delay. Several different methods have been suggested for these tasks, and an excellent review of this is found in [19]. Indeed, traditional clustering and classification techniques proposed for dynamic data streams, typically depend, either directly or implicitly, on the Bayesian principle of optimal classification.

**The "Anti-Bayesian" ML of Data Streams**: In this paper, we apply a novel alternative to the Bayesian classification approaches by operating in a diametrically opposite way, i.e., a so-called "Anti-Bayesian" (AB) manner. Indeed, we shall show the completely counter-intuitive result that by tracking a few points from each class which are distant from the mean, one can obtain remarkable classification performances for dynamic data streams — that can even outperform the Bayesian counter-part in some situations. Although we follow the steps of traditional ML and PR, we classify the data points to classes using completely different criteria, i.e., by invoking the AB paradigm. More specifically, unlike the traditional Bayesian classification strategies which rely on classifying based on the mean/central values of the classes, our paradigm advocates the classification of points to classes based on *quantiles distant from the means of each class* [29, 22], which is a concept that was previously unreported in the literature. Indeed, it is actually both un-intuitive and non-obvious.

It is fitting to mention that even though AB methods have found applications in classification and clustering, their corresponding application in dynamic streams is not consequential. This is because the samples that are "outliers" (and that represent the distant quantiles in any given distribution) may not continue to be "outliers" when the distribution changes. The fact that AB schemes are valid for even such non-stationary settings is one of the primary contributions of this paper.

***Multiplicative*** **Incremental Quantile Estimators**: As mentioned, the central concept of this paper involves using AB methods for dynamic streams. This, in turn, necessitates the dynamic estimation of the quantiles of a time-varying distribution. As is well known, the standard way of estimating a quantile related to some probability value $p$ in a static system, is to sort the quantiles and to then select the data point in position $\lfloor pn \rfloor$ or $\lceil pn \rceil$ (or using an appropriate weighting factor). As we will highlight later, such an approach can be non-functional, in practice, for dynamic data streams. However, incremental quantile estimators are estimators that do small (marginal) updates of the quantile estimates every time a new sample is received from the data stream. Quantile estimators, that are incremental in principle, have been reported in [33, 6, 5]. In this paper, we will, rather, invoke the more-recently introduced estimators due to Yazidi and Hammer [12, 37]. Being *multiplicative* incremental quantile estimators, they are not only more efficient than the current state-of-the-art quantile estimators for data streams, but are also far simpler to implement. The paper utilizes the Deterministic Update Based Multiplicative Incremental Quantile Estimator (DUMIQE) and its Multiple version, the MDUMIQE, which has proven consistent properties.

**An Enhanced Model of Computation**: In addition to all the issues mentioned here, as alluded to above, we also adopt the recently-proposed *online* classification model, with delay, proposed by Hanane *et al.* [28]. The model is composed of three stages. In the first phase, the model learns from the available labeled samples. In the second phase, the learned model predicts the class label of the unlabeled instance(s) currently observed. In the third phase, after knowing the true class label of these recently-classified instance(s), the classification model is adjusted in an *online* manner.

**Robustness against outliers**: When dealing with dynamic data, classical moving average estimation methods are inefficient as they are not able to deal with outlier observations which are well known to be susceptible to corrupting the estimated mean. However, the DUMIQE quantile estimator copes with this problem in a natural manner. This is an inherent part of the estimation process for quantiles, which, as such, makes AB classification much more robust against outliers.

**Experimentation Conducted**: The experimental portion of the paper compares the Bayesian and AB approaches using both synthetic and real-life data. In the first example of a real-life data set, we process outdoor air temperatures from different geographic locations, where the task is to classify the geographic location of each received temperature. The results show that the AB approach works very well, and in the best setting, it performs equally well when compared to the Bayesian approach. Further, a strength of the AB approach (when compared to the Bayesian approach) is that it is less sensitive to the choice of its tuning parameters. In the second real-life data set, the task encountered is to classify received tweets, which is also achieved in a very accurate manner using the AB scheme.

To summarize, the contributions of the paper are as follows:

- We develop a novel method to perform online classification in dynamically changing data stream.
- The method is based on combining the novel AB classification framework with the novel and state-of-the-art incremental quantile estimation techniques DUMIQE and MDUMIQE.

- The performance of the developed method is evaluated on one set of synthetic and two real data experiments.

### 1.2 Format of the Paper

First of all, in Section 2, we present a rather thorough overview of the current state-of-the-art. Section 3 presents the basic notation used in the rest of the paper. Then, in Section 4, we discuss the fundamentals of Bayesian and AB classification in static (or stationary) systems. In this section, we shall discuss the details of the techniques involved so that a practitioner can readily implement any of these methods. Section 4 then describes, in fair detail, the principles of Bayesian and AB classification in a static stream. Section 5 explains how we can efficiently track the quantiles and the mean value of a dynamic data stream, which leads, quite naturally to Section 6, where we explain the Bayesian and AB methodologies for classification in dynamical data streams. The next two sections describe the experimental results we have obtained for artificial and real-life data. Section 10 concludes the paper.

## 2 Related Work

In this section, we describe the related work both with respect to the relatively-new field of PR involving the AB paradigm. We will also briefly survey the state-of-the-art when it concerns classification in dynamic data streams.

### 2.1 Related Work on "Anti–Bayesian" PR

We first review the related work on AB classification. The review is, necessarily, very brief.

The first results on AB classification dates back to 2013, where Thomas and Oommen [29] proposed the use of the quantiles of the class conditions distributions to achieve classification, instead of using the information in the mean. They formally and experimentally showed that they could obtain optimal classification for various uni-dimensional symmetric distributions, and near-optimal accuracies for asymmetric distributions. For uni-dimensional quantile-based PR, their methodology is based on comparing the testing sample with the $\frac{n-k+1}{n+1}^{th}$ percentile of the first distribution and the $\frac{k}{n+1}^{th}$ percentile of the second distribution. These results were shown to be applicable for the distributions that are members of the symmetric and asymmetric exponential family. By considering the entire spectrum of the possible values of $k$, the results in [29], [31] and [22], showed that the specific value of $k$ is usually not so crucial. These authors also confirmed that the same results were true for *multi*-dimensional features.

In [30], the authors further proposed a new border identification algorithm, namely the AB Border Identification scheme. For each class, this method selects, as the corresponding border points, a small number of data points that lie close to the discriminant function's boundary, but where these points are not within the central part of the class conditional distributions.

The results of [29], [31] and [22] were used to design numerous Prototype Reduction Schemes in [32], and an AB text classification scheme in [20].

## 2.2 Other Learning Methods for Data Streams in NSE

According to the data stream mining literature, algorithms have one or more of the following modules: a Memory module, an Estimator module, and a Change Detector module [2]. The Memory module is a component that stores summaries of all the sample data and attempts to characterize the *current* data distribution. Data in non-stationary environments can be handled by three different approaches, namely, by using partial memory, by window-based approaches and by instance-based methods. The term "partial memory" refers to the case when only a part of the information pertaining to the training samples are stored and used regularly in the training. In window-based approaches, the data is presented as "chunks", and finally, in instance-based methods, the data is processed upon its arrival. In dynamic environments with non-stationary distributions, the Memory module indicates the forgetting mechanism of the mining algorithm in order to adapt the learning model to newer observations and to forget old information.

The Estimator module uses the information contained in the Memory or only the observed information to estimate the desired statistics of the time varying data stream. The Change Detector module involves the techniques or mechanisms utilized for detecting explicit drifts and changes, and provides an "alarm" signal whenever a change is detected based on the estimator's outputs.

Apart from the above schemes, many other incremental approaches have been proposed that infer change points during estimation, and use the new data to adapt the learning model trained from historical streaming data. The learning model in incremental approaches is adapted to the most recently-received instances of the streaming data. Let $X = \{x_1, x_2, \ldots, x_n\}$ be the set of training examples available at time $t = 1 \ldots n$. An incremental approach produces a sequence of hypotheses $\{\ldots, H_{i-1}, H_i, \ldots\}$ from the training sequence, where each hypothesis, $H_i$, is derived from the previous hypothesis, $H_{i-1}$, and the example $x_i$. In general, in order to detect concept changes in these types of approaches, some characteristics of the data stream (e.g., performance measures, data distribution, properties of data, or an appropriate statistical function) are monitored over time. When the parameters switch during the monitoring process, the algorithm should be able to adapt the model to these changes.

We now briefly review some *other* schemes used for learning in non-stationary environments. The review here will not be exhaustive because the methods explained can be considered to be the basis for other modified approaches.

### 2.2.1 FLORA

Widmer and Kubat [35], presented the FLORA family of algorithms as one of the first supervised incremental learning systems for a data stream. The initial FLORA algorithm used a fixed-size sliding window scheme. At each time step, the elements in the training window were used to incrementally update the learning model. The updating of the model involved two processes: an incremental *learning* process that updated the concept description based on the new data, and an incremental *forgetting* process that discarded the out-of-date (or stale) data.

The initial FLORA system did not perform well on large and complex data domains. Thus, FLORA2 was developed to solve the problem of working with a fixed window size, by using a heuristic approach to adjust the window size dynamically. Further improvements of the FLORA were presented to deal with recurring concepts (FLORA3) and noisy data (FLORA4).

### 2.2.2 Statistical Process Control (SPC)

The SPC was presented by Gama *et al.* [9] for change detection in the context of data streams. The principle motivating the detection of concept drift using the SPC is to trace the probability of the error rate for the streamed observations. While monitoring the errors, the SPC provides three possible states, namely, "in control", "out of control" and "warning" to define a state when a warning has to be given, and when levels of changes appear in the stream. When the error rate is lower than the first (lower) defined threshold, the system is said to be in an "in control" state, and the current model is updated considering the arriving data. When the error exceeds that threshold, the system enters the "warning" state. In the "warning" state, the system stores the corresponding time as the warning time, $t_w$, and buffers the incoming data that appears subsequent to $t_w$. In the "warning" mode, if the error rate drops below the lower threshold, the "warning" mode is canceled and the warning time is reset. However, in case of an increasing error rate that reaches the second threshold, a concept change is declared and the learning model is retrained from the buffered data that appeared after $t_w$.

### 2.2.3 ADWIN

Bifet and Gavalda [3, 4] proposed an adaptive sliding window scheme named ADWIN for change detection and for estimating statistics from the data stream. It was shown that the ADWIN algorithm outperforms the SPC approach and that it has the ability to provide rigorous guarantees on false positive and false negative rates. The initial version of ADWIN keeps a variable-length sliding window, $W$, of the most recent instances by considering the hypothesis that there is no change in the average value inside the window. To achieve this, the distributions of the sub-windows of the $W$ window are compared using the Hoeffding bound, and whenever there is a significant difference, the algorithm removes all instances of the older sub-windows and only keeps the new concepts for the next step. Thus, a change is reliably detected whenever the window shrinks, and the average over the existing window can be considered as an estimate of the current average in the data stream.

To be more specific, consider a sequence of real values $\{x_1, x_2, \ldots, x_t, \ldots\}$ that is generated according to the distribution $D_t$ at time $t$. Let $n$ denote the length of the $W$ window, $\hat{\mu_t}$ be the observed average of the elements in $W$, and $\mu_w$ be the true average value of $\mu_t$ for $t \in W$. Whenever two "large enough" sub-windows of $W$ demonstrate "distinct enough" averages, the system infers that the corresponding expected values are different, and the older fragment of the window should be dropped. The observed average in both sub-windows are "distinct enough" when they differ by more than the threshold $\epsilon_{cut}$:

$$\epsilon_{cut} = \sqrt{\frac{1}{2m} . \ln \frac{4}{\delta'}}, \text{where} \tag{1}$$

$$m = \frac{1}{1/n_0 + 1/n_1}, \quad \text{and} \quad \delta' = \frac{\delta}{n}, \tag{2}$$

where $n_0$ and $n_1$ denote the lengths of the two sub-windows, and where $\delta$ is a confidence bound.

Using the Hoeffding bound greatly over estimates the probability of large deviations for distributions with a small variance, which degrades the ADWIN's performance. Besides, it is also computationally demanding [23].

The ADWIN approach is, in fact, a linear estimator enhanced with a change detector. In order to improve the basic ADWIN method's performance, Bifet [2] replaced the linear estimator by an adaptive Kalman filter, where the covariances of $w(n)$ and $v(n)$ were set to $n^2/50$ and $200/n$ respectively, where $n$ is the length of the window maintained by ADWIN.

### 2.2.4 Miscellaneous Approaches

To perform classification in dynamic systems, a common strategy is to transfer acclaimed classification schemes for static data to a dynamic environment by training the classifier on a sliding window, or an exponential weighting of historic data. A challenge is that for most classifiers, like the Support Vector Machine (SVM), it is challenging to recursively update the parameters of the classifiers when new data arrive. Several attempts have been suggested for different classifiers to overcome this challenge.

The Hoeffding tree is a decision tree classifier for data streams [8]. Traditional decision trees need to scan the training data many times to select the splitting attribute. However, this requirement is infeasible in the data stream environment. To overcome this limitation, the Hoeffding bound is used to choose an optimal splitting attribute within receiving a sufficient amount of data objects.

Seidl *et al.* proposed a novel index-based classifier called the Bayes tree [26]. Adapted from the

R*-tree [1], the Bayes tree generates a hierarchical Gaussian-mixture tree to represent the entire data set. Each tree node contains statistics of the data objects including a minimum bounding rectangle, the number of data objects, the linear sum and the quadratic sum of all the data objects.

The SVM has demonstrated its prominent performance in many ML problems with static data sets. However, it is very expensive to use SMVs in large-scale applications due to its time and memory complexity. Tsang *et al.* proposed the Core Vector Machine (CVM) algorithm that uses the Minimum Enclosing Ball (MEB) to reduce its complexity [34].

For other classifiers based on linear regression (ordinary least squares), the parameters of the model can be updated recursively, [14, 13]. Even the least squares approach that uses Tikhonov regularization (ridge regression) can be updated recursively like the traditional least squares approach.

For dynamically changing data streams, one may expect a strong temporal dependence. One may expect a similar temporal dependence in the class labels. Mayo and Bifet [17] take advantage of this imporant observation when constructing online classification algorithms. They show how simple classifiers such as Naive Bayes can boost their performance.

Recently several papers focus on classification in dynamically varying data streams were the number of labels received online are minimal or not present at all.

Plumpton et al. [24] consider the problem of real-time classification of fMRI data where the labeling is very limited. The authours update the classifier using so called *naive labelling*. Naive labelling is a protocol where in the absence of ground truth, updates are carried out using the label assigned by the classifier.

Lowne et al. [15] look at situations where decision boundaries between classes are potentially non-linear and subject to "concept drift". The inherent non-stationarity in the data is tracked using a non-linear dynamic classifier, the parameters of which evolve under an extended Kalman filter framework, derived using a sequential Bayesian-learning paradigm. The method is extended to take into account missing and incorrectly labeled targets and to actively request target labels.

Souza et al. [27] suggest an approach where no labeled data become available and refer to this as extreme verification latency. The authors sugest a methods which consists of a clustering step followed by a classification step applied repeatedly in a closed loop fashion. Saki and Kehtarnavaz [25] also apply clusting to perform classification where no labeled data become available. The algorithm consists of a number of steps including density-based outlier removal, decision on the number of clusters, new cluster generation, and cluster update.

## 3 Basic Notation

In this section we present the basic notation used in the rest of the paper. Let $X(t)$ be a stochastic variable representing the outcome from a dynamic data stream at time $t$. We assume that $X(t)$ is from one of $K$ classes $C(t) \in \{1, 2, \ldots, K\}$ and that the probability that $X(t)$ is from class $C(t) = k$ is $p_k(t)$. The conditional distribution of $X(t)$ given class $C(t) = k$, has the probability distribution $f_t(x|k)$, i.e., $X(t)|C(t) = k \sim f_t(x|k)$. Using the law of total probability, we deduce that the marginal distribution of $X(t)$ is given by $f_t(x) = \sum_{k=1}^{K} f_t(x|k)p_k(t)$. Finally, let $(x(t), c(t))$ denote an outcome of the pair, $(X(t), C(t))$, which is the data point examined and its corresponding class label.

## 4 Bayesian and AB Classification in a Static System

In this section we describe the Bayesian and AB classification methodologies. To make the presentation easier, we consider, in this section, a static data stream, i.e. $X(t) = X$, $C(t) = C$, $p_k(t) = p_k$ and so on. We also assume that we have a training set of $n$ samples with class labels, $(x_1, c_1), (x_2, c_2), \ldots, (x_n, c_n)$.

### 4.1 Bayesian Classification

Let $\widehat{\mu}_k$ denote the mean value of the samples from class $k$, i.e.

$$\widehat{\mu}_k = \frac{1}{\sum_{i=1}^{n} I(c_i = k)} \sum_{i=1}^{n} I(c_i = k)x_i,$$

where $I(A)$ denote the indicator function that returns the value of unity if $A$ is true, and the value of zero if $A$ is false.

We now receive a new sample $x_0$ whose class is unknown, and the intention is to classify it to one of the classes. We assume that the distributions $f(x|k)$ and $p_k$ are unknown, and the classification must be based on the training samples. Explained in a rather informal manner, the optimal Bayesian classification rule is to assign $x_0$ to the class whose class mean is is closest to $x_0$, i.e., assign $x_0$ to class $k$ if

$$\|x_0 - \widehat{\mu}_k\| < \|x_0 - \widehat{\mu}_j\| \quad \forall j \neq k.$$

Of course, one must also consider the actual metric used to measure the distance from the means. Indeed, this need not necessarily be the simple Euclidean metric, but could rather be one based on the covariance matrices, for example, the Mahalonabis distance.

### 4.2 The Anti-Bayesian (AB) Classification

The AB paradigm is based on a radically different approach from its Bayesian counterpart, where the classification is based on quantiles *distant* from the mean, rather than the mean. The methodology is described in [29], [31] and [22], where its properties have also been proven. Let $Q_{kp}$ denote the quantile related to a given probability value, $p$, for a class whose index is $k$, i.e. $P(X \leq Q_{kp} | C = k) = p$. Further, let $\widehat{Q}_{kp}$ denote an estimate of $Q_{kp}$ based on the sample $(x_1, c_1), (x_2, c_2), \ldots, (x_n, c_n)$ using some quantile estimation procedure. We now define $q = 1 - p$ and assume that $p < 1/2$. Consequently, we clearly see that $Q_{kp} < Q_{kq}$.

To explain the AB approach, assume for the present that we have only two classes denoted by $k = 1$ and $k = 2$. A generalization to $K$ classes will then be explained in the next step. In such a case, the AB classification method operates as follows:

1. Determine which of the distributions $f(x|k = 1)$ or $f(x|k = 2)$ is to the left by using the quantiles of the distributions. We have three possible cases:

   **Case 1:** If $\widehat{Q}_{1p} < \widehat{Q}_{2p}$ and $\widehat{Q}_{1q} < \widehat{Q}_{2q} \implies f(x|k = 1)$ is to the left of $f(x|k = 2)$.

   **Case 2:** If $\widehat{Q}_{1p} > \widehat{Q}_{2p}$ and $\widehat{Q}_{1q} > \widehat{Q}_{2q} \implies f(x|k = 2)$ is to the left of $f(x|k = 1)$.

   **Case 3:** Else[1], we determine their relative positions by comparing the averages of the quantiles as follows:

   If $\frac{\widehat{Q}_{1p}+\widehat{Q}_{1q}}{2} < \frac{\widehat{Q}_{2p}+\widehat{Q}_{2q}}{2} \implies f(x|k = 1)$ is to the left of $f(x|k = 2)$.
   Else $f(x|k = 2)$ is to the left of $f(x|k = 1)$.

   Figure 1 depicts the above three cases. We see that for Cases 1 and 2, $f(x|k = 1)$ and $f(x|k = 2)$ are the distributions to the left, respectively. In the bottom figure (Case 3), the decision is not that obvious because the classes are highly overlapping.

2. Once the relative positions of the distributions are determined, the classification rule must now be specified. For simplicity, we describe this merely for Case 1 since the rules for the "mirrored" cases are analogous. The AB rule classifies using the *right* quantile of the left distribution and the *left* quantile of the right distribution. If $B = \frac{\widehat{Q}_{1q}+\widehat{Q}_{2p}}{2}$, we classify as follows:

   If $x_0 < B$, classify $x_0$ to class $k = 1$.

   Else, classify $x_0$ to class $k = 2$.

   This approach works even when the distributions overlap such that $\widehat{Q}_{2p}$ is to the left of $\widehat{Q}_{1q}$ as shown in Figure 2.

If we need to classify $x_0$ to one of $K > 2$ classes, we simply repeat the procedure described above $K - 1$ times in a "winner-takes-all" sequential, pairwise manner. First, we compute if $x_0$ is more

[1]This case occurs rarely in practice except when the classes are highly overlapping, in which case the classification problem is often meaningless. While Cases 1 and 2 select one quantile from each distribution in accordance with the AB paramdigm, it is not obvious how to do this for Case 3. This motivates to classify based on the average of the quantiles.

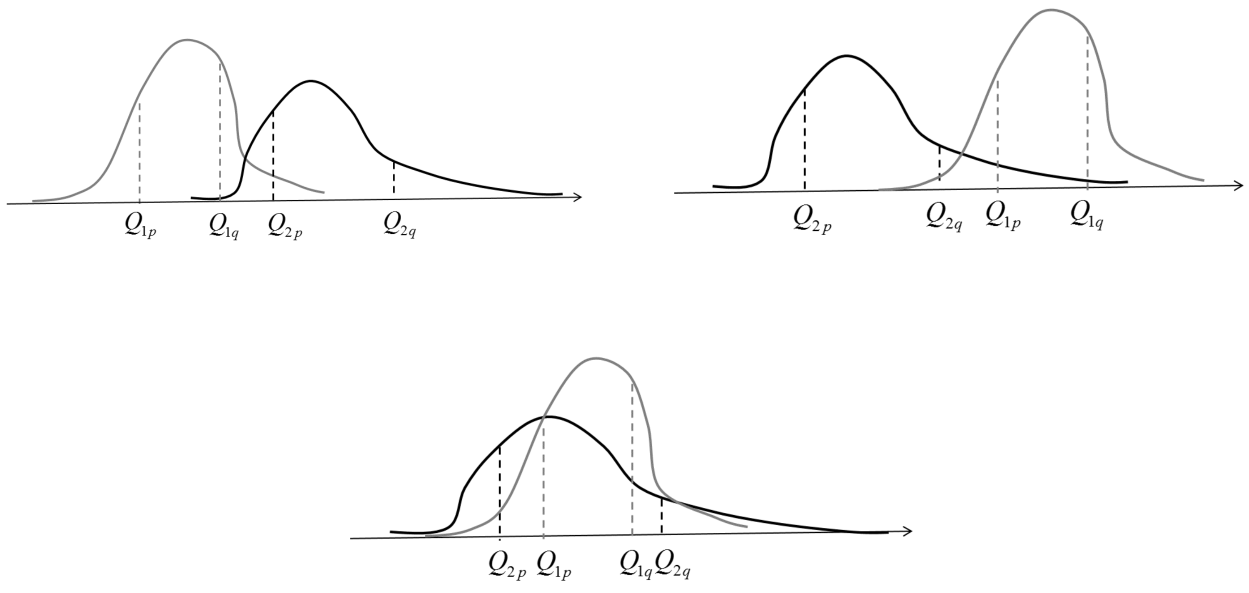

Figure 1: This figure depicts Cases 1, 2 and 3 – arranged from left to right and from top to bottom respectively.

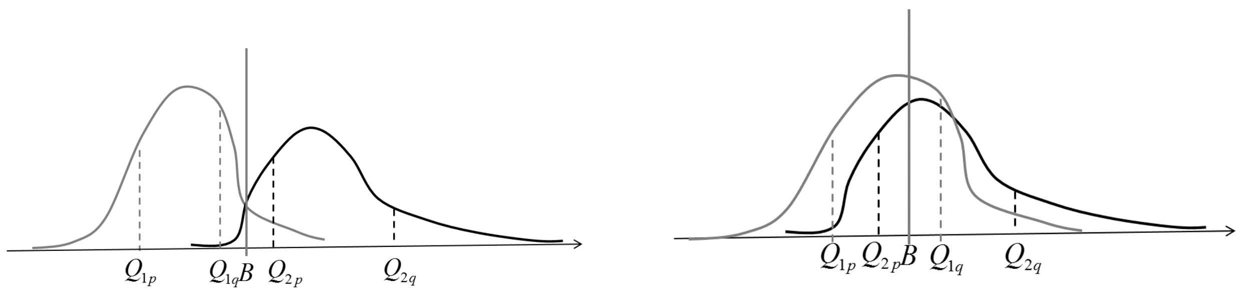

Figure 2: The left panel shows the standard situation under Case 1, while the right panel shows a situation when $\widehat{Q}_{2p}$ is to the left of $\widehat{Q}_{1q}$.

likely to belong to class $k = 1$ or $k = 2$. Assume that class $k = 2$ is the most likely one. We thereafter do an evaluation between classes $k = 2$ and $k = 3$, and repeat this for all the remaining classes $4, \ldots, K$. Finally, we classify $x_0$ to the class that is most likely to be the assigned class, after going through all the $K - 1$ evaluations.

## 5 Tracking Quantiles and the Mean Value of Dynamic Data Streams

We now present algorithms to track the quantiles and the mean values of a data stream. Here, we assume that samples arrive at equidistant time steps[2], i.e., $x(1), x(2), \ldots, x(t), x(t+1), \ldots$.

[2]The methodology in this section and in Section 6 can easily be extended to cases when when samples are received at arbitrary time points $x(t_1), x(t_2), x(t_3), \ldots$. We will look at such scenarios in Section 8.2, where the problem involves

## 5.1 Tracking Quantiles

We initiate discussions by presenting methods for tracking the quantiles of dynamic data streams. Let $Q_{kp}(t)$ denote the quantile of $X(t)|C(t) = k$ for some probability value $p$, i.e.,

$P(X(t) \leq Q_{kp}(t)|C(t) = k) = p.$

Further, let $\widehat{Q}_{kp}(t)$ be an estimator of $Q_{kp}(t)$. The standard way of estimating a quantile related to some probability value $p$ in a static system is to sort the quantiles, and to then select the data point in position $\lfloor pn \rfloor$ or $\lceil pn \rceil$ (or using an appropriate weighting factor). Unfortunately, such an approach would not work well for dynamic data streams as the computation time and memory requirement increases linearly with the number of samples, $n$, arriving from the data stream.

Incremental quantile estimators are estimators that do small updates of the quantile estimates every time a new sample is received from the data stream. Incremental quantile estimators have been documented to yield a good performance for dynamical systems, as reported in [33, 6, 5]. More recently, Yazidi and Hammer suggested multiplicative incremental quantile estimators that are not only more efficient then the current state-of-the-art quantile estimators for data streams, but are also far simpler to implement [12, 37]. The AB classifications presented here are, therefore, based on these algorithms used to estimate the quantiles[3]. We now give a short description of the algorithms by Yazidi and Hammer [12, 37].

Suppose that we need to track only a single quantile of the distribution related to some probability $p$. The method reported in [12, 37] is as follows. We start with some initial quantile estimate $\widehat{Q}_{kp}(0)$, and update the quantile estimate for this class every time we receive a new sample $x(t)$ from class $k$ as per Eq. (3):

$$\begin{aligned} \widehat{Q}_{kp}(t+1) &\leftarrow \widehat{Q}_{kp}(t) + \lambda p \widehat{Q}_{kp}(t), && \text{if } x(t) > \widehat{Q}_{kp}(t) \\ \widehat{Q}_{kp}(t+1) &\leftarrow \widehat{Q}_{kp}(t) - \lambda(1-p) \widehat{Q}_{kp}(t), && \text{if } x(t) \leq \widehat{Q}_{kp}(t). \end{aligned} \tag{3}$$

The idea in the above updating rule is quite simply the following: If the sample $x(t)$ is above (below) our current estimate, we should respectively increase (decrease) the corresponding quantile estimates. The variable $\lambda$ is a parameter that controls the step size, and the weighting with $p$ and $1 - p$ is included to ensure that the estimator converges to the true quantile. A potential challenge with these simple rules is that if we start with $\widehat{Q}_{kp}(0) > 0$, every estimate will be above zero whenever $0 < \lambda < 1$. One solution that works well in practice is to run the update rules on a right shifted quantile estimate that is known to be above zero. The estimate of $Q_{kp}(t)$ is then determined by a left shift of the right shifted estimate. For more details about this scheme, referred to as the Deterministic Update Based Multiplicative Incremental Quantile Estimator (DUMIQE), we refer

the online classification of tweets.

[3]This approach can be seen as the AB counterpart of the Bayesian classification approach where the means of the classes are tracked by the exponential moving average.

the reader to [12, 37].

To now specifically apply the DUMIQE to AB classification in a dynamic environment, we observe that we need to track two quantiles for the distribution of each class, namely, for the probabilities $p < 1/2$ and $q = (1-p)$. One approach is to simply use the above DUMIQE scheme to estimate both of these quantiles. A challenge with this approach is that we may end up with unrealistic estimates in the sense that the monotone property of quantiles gets violated. This means that $\widehat{Q}_{kp}(t)$ gets a higher value than $\widehat{Q}_{kq}(t)$ even though $p$ is less than $q$.

In [12], Hammer et al. suggested a modification of the DUMIQE scheme to ensure that the monotone property of the quantiles are satisfied in every iteration. Suppose that at time $t$ that the monotone property is satisfied, i.e. $\widehat{Q}_{kp}(t) < \widehat{Q}_{kq}(t)$. We may get a violation if $x(t)$ gets a value between $\widehat{Q}_{kp}(t)$ and $\widehat{Q}_{kq}(t)$. According to Eq. (3), we will obtain the following updates:

$\widehat{Q}_{kp}(t+1) \leftarrow \widehat{Q}_{kp}(t) + \lambda p \widehat{Q}_{kp}(t)$, which is an increased value, and

$\widehat{Q}_{kq}(t+1) \leftarrow \widehat{Q}_{kq}(t) - \lambda(1-q)\widehat{Q}_{kq}(t)$, which is a decreased value.

Since the lower quantile estimate gets an increased value while the upper quantile receives a reduced value, we observe that we could obtain an overlap that violates the monotone property of the quantiles. The idea suggested in [12] is to adjust the update size, $\lambda$, to ensure that this quantile monotone property is satisfied. One such value of $\lambda$ (denoted $\widetilde{\lambda}$ below) can be determined by ensuring that the distance between $\widehat{Q}_{kp}(t+1)$ and $\widehat{Q}_{kq}(t+1)$ is some portion, $\alpha \in (0,1)$, of the distance from the previous iteration, i.e.,

$$\begin{aligned} \widehat{Q}_{kq}(t+1) - \widehat{Q}_{kp}(t+1) &= \alpha\left(\widehat{Q}_{kq}(t) - \widehat{Q}_{kp}(t)\right) \\ (1 - \widetilde{\lambda}(\underbrace{1-q}_{=p}))\widehat{Q}_{kq}(t) - (1+\widetilde{\lambda}p)\widehat{Q}_{kp}(t) &= \alpha\left(\widehat{Q}_{kq}(t) - \widehat{Q}_{kp}(t)\right). \end{aligned} \tag{4}$$

Solving Eq. (4) with respect to $\widetilde{\lambda}$ yields:

$$\widetilde{\lambda} = \beta \frac{\widehat{Q}_{kq}(t) - \widehat{Q}_{kp}(t)}{p\left(\widehat{Q}_{kq}(t) + \widehat{Q}_{kp}(t)\right)}, \tag{5}$$

where $\beta = 1 - \alpha$. By utilizing the quantity $\widetilde{\lambda}$ for the parameter $\lambda$ in Eq. (3) whenever the updating of both $\widehat{Q}_{kp}(t)$ and $\widehat{Q}_{kq}(t)$ are done, we can ensure that the monotone property is satisfied at every iteration. The parameter $\beta$, however, controls the size of the update. Using a value of $\beta$ close to zero, results in small updates, while setting $\beta$ close to unity, performs maximal updates without violating the monotone property. For the rest of this paper, we refer to this scheme as the Multiple DUMIQE (MDUMIQE), and mention in passing that the proof of convergence for this scheme and various other computational details are found in [12]. They are omitted here in the interest of brevity.

## 5.2 Tracking the Mean Value

The above schemes, DUMIQE and MDUMIQE, are computationally extremely light-weighted since the quantiles are tracked by only a single operation in every iteration, by resorting to Eq. (3). The natural analog when tracking the mean value, is the exponential moving average (EMA). To be more specific, let $\widehat{\mu}_k(t)$ denote the estimate of the mean value of class $k$ at time $t$. In the EMA scheme we update the estimate as follows:

$$\widehat{\mu}_k(t+1) \leftarrow (1-\gamma)\widehat{\mu}_k(t) + \gamma\, x(t) \tag{6}$$

for some $\gamma \in [0,1]$.

The part that remains is that of knowing how to choose reasonable values for the tuning parameters $\lambda, \beta$ and $\gamma$. If the dynamics of the data stream change rapidly (slowly) one should use large (small) values of the tuning parameters. One can of course use the history of the data stream to learn reasonable values of the tuning parameters based on the estimation/classification performance. Unfortunately, this will not work well if the dynamics of the data stream changes with time, e.g., if the stream changes rapidly in some time periods and slowly in others. The good news is that the performance of the tracking methods usually is quite robust on the choice of the tuning parameters. Choosing $\lambda$ and $\gamma$ around 0.05 and a value of $\beta$ around 0.2 perform satisfactory for most applications.

## 6 Bayesian and AB Classification in Dynamical Data Streams

We now have the tools to perform classification in dynamic data streams. We first explain the methodology for the Bayesian case and then proceed to the Anti-Bayesian paradigm.

### 6.1 Bayesian Classification

Bayesian classification is done in the manner explained earlier, and this has, indeed, been the basis for classification for decades. Here, in every iteration, the classification is based on the approach detailed in Section 4.1. Every time we receive a new sample, we update that estimate of the mean value based on the class label, i.e., update $\widehat{\mu}_{c(s)}(s)$ as per Eq. (6):

$$\widehat{\mu}_{c(s)}(s+1) \leftarrow (1-\gamma)\widehat{\mu}_{c(s)}(s) + \gamma\, x(s),$$

for $s \leq t$. The estimates of the mean values for the other classes remain unchanged.

The reader should note that as per our model, we receive a sample $x(t+1)$ whose class is unknown. We then classify $x(t+1)$ to one of the $K$ classes by using the Bayesian classification method described in Section 4.1 using the estimates of the mean values for each class at time $t$, namely $\widehat{\mu}_k(t)$, $k = 1, \ldots, K$. Whenever we receive the true class labels after a subsequent delay of $h$ time steps, we follow the same procedure as described above, except that we also include the known class of *this* sample in the updated training step.

- Classify $x(t+1)$ to one of the $K$ classes as per the Bayesian rule, and denote the result as $\widehat{c}(t+1)$.
- Update the estimate of the mean value for class $\widehat{c}(t+1)$, $\widehat{\mu}_{\widehat{c}(t+1)}(t)$ using Eq. (6).
- Classify $x(t+2)$ to one of the $K$ classes using the estimates of the mean values at time $t+1$, namely $\widehat{\mu}_k(t+1)$, $k = 1, \ldots, K$.
- Update the estimate of the mean value of class $\widehat{c}(t+2)$ using Eq. (6).
- Repeat the above steps till time $t+h$.

### 6.2 AB Classification

To explain the AB classification, we assume that we have received samples with their respective class labels up to time $t$, $(x(1), c(1)), (x(2), c(2)), \ldots, (x(t), c(t))$. Every time we receive a new sample, we update that quantile estimates based on the class label $\widehat{Q}_{c(s)\,p}(s)$ and $\widehat{Q}_{c(s)\,q}(s)$ as per Eq. (3):

$$\begin{aligned}
\widehat{Q}_{c(s)\,p}(s+1) &\leftarrow \widehat{Q}_{c(s)\,p}(s) + \lambda p \widehat{Q}_{c(s)\,p}(s), && \text{if } x(s) > \widehat{Q}_{c(s)\,p}(s)\\
\widehat{Q}_{c(s)\,p}(s+1) &\leftarrow \widehat{Q}_{c(s)\,p}(s) - \lambda(1-p) \widehat{Q}_{c(s)\,p}(s), && \text{if } x(s) \leq \widehat{Q}_{c(s)\,p}(s),\\
\widehat{Q}_{c(s)\,q}(s+1) &\leftarrow \widehat{Q}_{c(s)\,q}(s) + \lambda q \widehat{Q}_{c(s)\,q}(s), && \text{if } x(s) > \widehat{Q}_{c(s)\,q}(s)\\
\widehat{Q}_{c(s)\,q}(s+1) &\leftarrow \widehat{Q}_{c(s)\,q}(s) - \lambda(1-q) \widehat{Q}_{c(s)\,q}(s), && \text{if } x(s) \leq \widehat{Q}_{c(s)\,q}(s),
\end{aligned}$$

for $s \leq t$. For the MDUIQE scheme, we use $\widetilde{\lambda}$ from Eq. (5) in place of $\lambda$ in the above updates. The quantile estimates for the other classes, $\widehat{Q}_{kp}(s)$ and $\widehat{Q}_{kq}(s)$ for $k \neq c(s)$, now remain unchanged.

Now suppose that we receive a new sample $x(t+1)$, whose class identity is unknown. We classify $x(t+1)$ to one of the $K$ classes by using the AB classification method described in Section 4.2 using the quantile estimates for each class at time $t$, namely $\widehat{Q}_{kr}(t), r = p, q,\ k = 1, \ldots, K$.

We may also consider the case where we receive class labels with a delay of $h$ time steps. In this sense, at time instant $t$ we have samples $x(t+1), \ldots, x(t+h)$ with unknown class labels. To classify these samples, we use the following iterative procedure:

- Classify $x(t+1)$ to one of the $K$ classes as described above, and denote the result $\widehat{c}(t+1)$.
- Update the quantile estimates of class $\widehat{c}(t+1)$, $\widehat{Q}_{\widehat{c}(t+1)\,r}(t)$, $r = p, q$ using DUMIQE/MDUMIQE.

- Classify $x(t+2)$ to one of the $K$ classes using the quantile estimates from time $t+1$, namely $\widehat{Q}_{kr}(t+1), r = p, q,\ k = 1, \ldots, K$.
- Update the quantile estimates of class $\widehat{c}(t+2)$ using DUMIQE/MDUMIQE.
- Repeat the above steps till time $t+h$.

### 6.3 Pros and Cons of the AB and Bayesian Dynamical Classification Methods

There are a few pros and cons of the suggested methods in this paper.

A major strength is that they are computationally extremely efficient. The tracking procedures are potentially able to track the mean and quantiles of almost any data stream. However, the tracking methods are very simple and not able to learn systematic trends, cycles or seasonalities from history like traditional time series models (e.g., ARIMA models) [7]. The performance of traditional time series models are on the other hand more sensitive to changes in the dynamics of the data streams, e.g., if the stream goes from a period with rapid changes to a period with slow changes.

Comparing the AB and Bayesian dynamical classification methods we now emphasize an important property of AB classification which renders it to be superior to the Bayesian classification in dynamic environments. By virtue of the design of the quantile estimator, the AB approach is robust against outliers. This is a phenomenon that is absent in the Bayesian approach.

To clarify why this is true, we explain how the DUMIQE handles outliers. Although the magnitude of the observation is fed to the algorithm, only the fact whether the new observation is larger or smaller than the current quantile estimate is of significance. In other words, DUMIQE updates are based on the *sign* of the difference between the estimate and observation, while the EMA relies directly on the magnitude of the observations, to estimate the mean. It is thus clear that outliers might corrupt the mean estimate, while they will not have such a significant effect on the quantile estimates. In Section 7.4, we present some experiments that illustrate this specific phenomenon by corrupting the data with some outliers.

In Section 7 and Section 8, we shall demonstrate the power of schemes for synthetic and real-life data sets.

## 7 Experiments Results: Synthetic Data

We first compared the performance of the Bayesian and AB algorithms using synthetic data sets. The details of these sets and the results obtained are explained below.

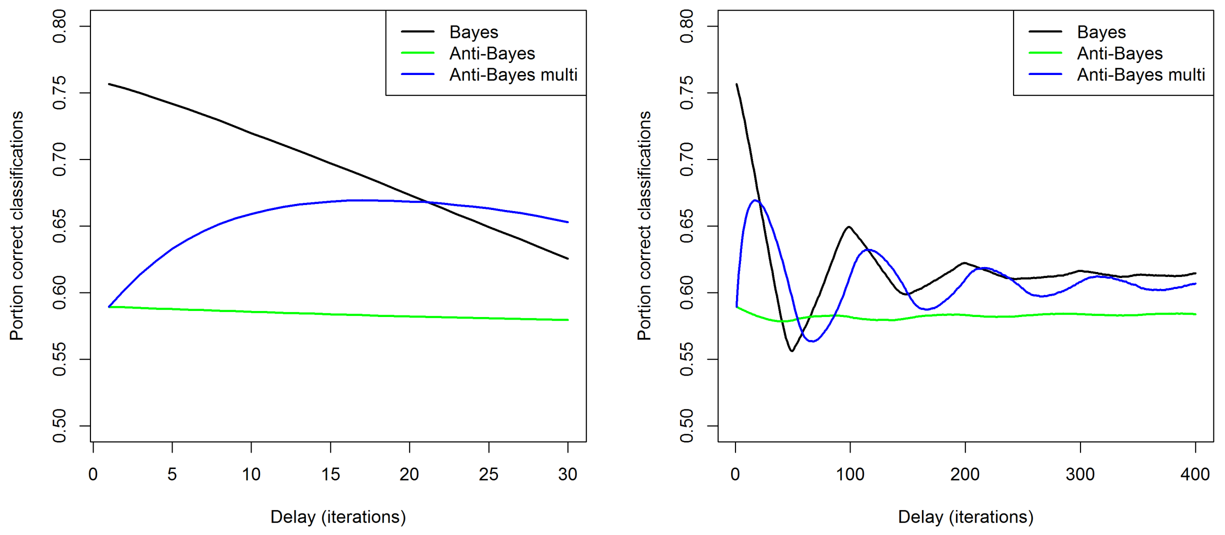

Figure 3: Jump process: Portion of samples correctly classified for the Bayesian and AB algorithms as a function of the delay on the class label information. The right panel shows portions of correct classifications for all delays up to 400 time steps, while the left figure zooms-in on delays up to 30 time steps. 'Anti-Bayes' and 'Anti-Bayes multi' refer to the AB approaches that use the DUMIQE and MDUMIQE schemes respectively, to track the quantiles.

## 7.1 Jump Processes

In this set of experiments, we assumed that the distribution for class $k$ was normally distributed with expectation $\mu_k$ and standard deviation $\sigma$. We assumed a jump process with period $T$ such that the expectations "jumped" by a value of $b$ every half period. Formally, this is defined as:

$$\mu_k(t) = \begin{cases} ak & \text{if} \quad (t \mod T) < T/2 \\ ak + b & \text{if} \quad (t \mod T) > T/2, \end{cases}$$

for $k = 1, 2, \ldots, K$. Thus, the expectations for the different classes were separated by a difference $a$. In essence, we had the situation where $X(t)|C(t) = k \sim f_t(x|k) = N(\mu_k(t), \sigma)$ for $k = 1, 2, \ldots, K$. Finally, we assumed that $p_k(t) = \frac{1}{K}$ for $k = 1, 2, \ldots, K$ and all time steps.

In the first set of experiments, we fixed $\sigma = 1, a = 2, \sigma = 2, b = 4$ and $T = 100$. We also considered two cases where $K = 2$ and $K = 10$ classes. We then evaluated the classification performance of the AB approach using both DUMIQE and MDUMIQE to track the quantiles, and the Bayesian scheme using the EMA to track the distributions' mean values, as described in Section 6. Figure 3 shows the portion of samples that were correctly classified when we were dealing with $K = 2$ classes using the three algorithms with $\lambda$ being set to 0.01, and $\beta$ and $\gamma$ being set to 0.2.

From these results, we see that the classification performance varied periodically with a period

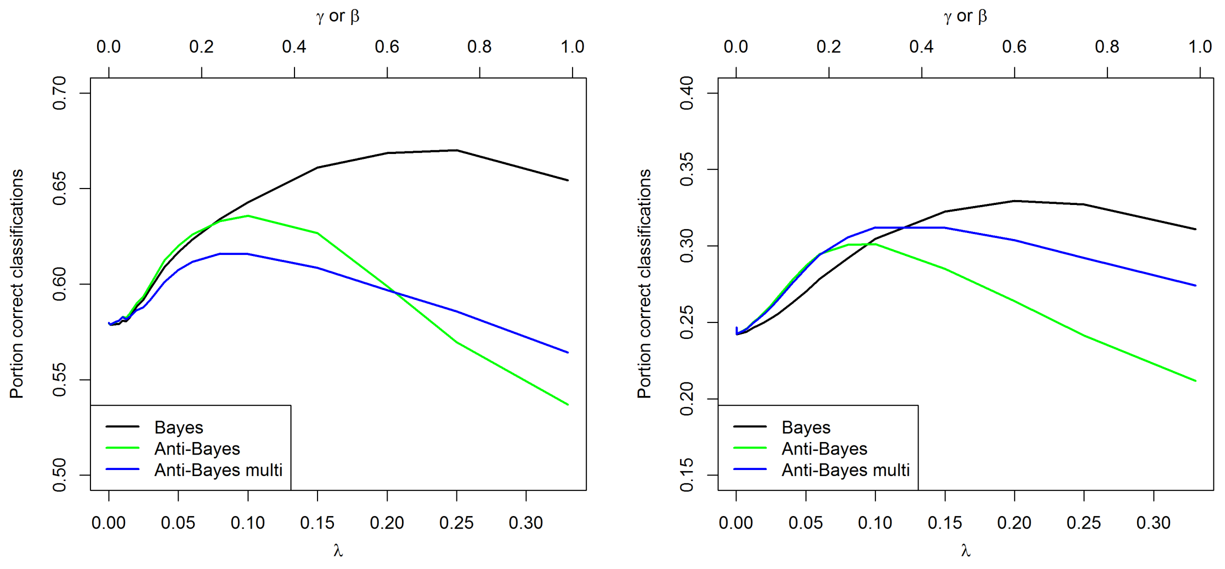

Figure 4: Jump process: Portion of samples correctly classified for the Bayesian and AB algorithms for different choices of the tuning parameters. The left and the right panels display the cases with $K = 2$ and $K = 10$ classes, respectively. The 'Anti-Bayes' and 'Anti-Bayes multi' curves refer to the AB approaches using the DUMIQE and MDUMIQE respectively, to track the quantiles.

equal to the period in the sample process, i.e., $T = 100$. The explanation is that it is easier for the schemes to predict when the delay is about one period compared to the scenario when it is only half a period. Another interesting observation is that for the Anti-Bayes multi, the classification performance was better for a delay of about 30 time steps compared to when the delay was smaller, which may seem surprising. We also observe the same phenomenon for the other two algorithms and for other choices of the tuning parameters. The explanation is that after a jump, the algorithms had to track the means and the quantiles, and it was still possible to do this in such a way that the classification improved even though we did not know the class labels of the received samples.

To demonstrate the performance of the three algorithms, we computed the portion of samples that were correctly classified when we averaged over all delays up to one period $T$, and used a large set of different choices for the three tuning parameters $\lambda, \beta$ and $\gamma$. Figure 4 shows the results. For $K = 2$, the Bayesian approach performed a little better than the AB approach. The best classification was achieved using a value of $\gamma$ around 0.7. For $K = 10$, the three algorithms performed about equally well, which is quite a fascinating results since the AB works in completely counter-intuitive manner.

## 7.2 Shrink/Expand Processes

In the second example, we investigated a process where the differences between the mean values shrank and expanded. We used the same setup and choice of parameters as above, but we used the following shrink/expand model for the variation of the mean values, described formally as:

$$\mu_k(t) = \begin{cases} a(k - K/2) & \text{if} \quad (t \mod T) < T/2 \\ 2a(k - K/2) & \text{if} \quad (t \mod T) > T/2, \end{cases}$$

for $k = 1, 2, \ldots, K$. The classification performances for the three algorithms for $K = 10$ classes are shown in left panel of Figure 5.

We see that the Bayesian approach performed slightly better than the AB approach. Further, we see that the AB approach using MDUMIQE for tracking the quantiles performed better than when it used the DUMIQE.

## 7.3 Switch Processes

In the third example, we considered a process where the mean values were modified by switching values with each other. More specifically the mean values were changed as follows:

$$\mu_k(t) = \begin{cases} ak & \text{if} \quad (t \mod T) < T/2 \\ a(K - k + 1) & \text{if} \quad (t \mod T) > T/2. \end{cases}$$

It was quite challenging to track the mean values and quantiles of the different classes since they entailed large changes of the mean values. Thus, for this process, we set $T = 1,000$ instead of $T = 100$ as we used for the two processes described above. The classification performance for the three algorithms for $K = 10$ classes are shown in right panel of Figure 5. Again we see that the Bayesian approach performed slightly better than the AB approach, and that the AB approach using MDUMIQE for tracking the quantiles performed better than when it used DUMIQE.

## 7.4 Jump Processes with outliers

Finally, to demonstrate the power of the AB paradigm when it concerns outliers, we investigated the performance of the algorithms for the scenario when a few of the received observations were distinctly outliers. We assumed a jump process identical to the one used Section 7.1, but where random observations, in every period $T$, were distorted. More specifically, for the distorted observations, instead of observing the stochastic variable $X(t)|C(t)$, we rather observe $X(t)|C(t) + \Delta$, where $\Delta$ is a stochastic variable taking the values $\delta$ and $-\delta$ with probability 0.5. Figure 6 shows the results for $K = 2$ classes and $\delta = 10$ and $10,000$. By comparing the left panel of Figure 6 with the left panel of Figure 4, we see that the performance of the Bayesian approach is *significantly* reduced only with minor outliers of size $\delta = 10$. By comparing the right panel of Figure 6 with the left panel of Figure 4, we see that when the outliers are increased to $\delta = 10,000$ the performance of the Bayesian approach is further significantly reduced. Finally, we see that the AB methods have no problems

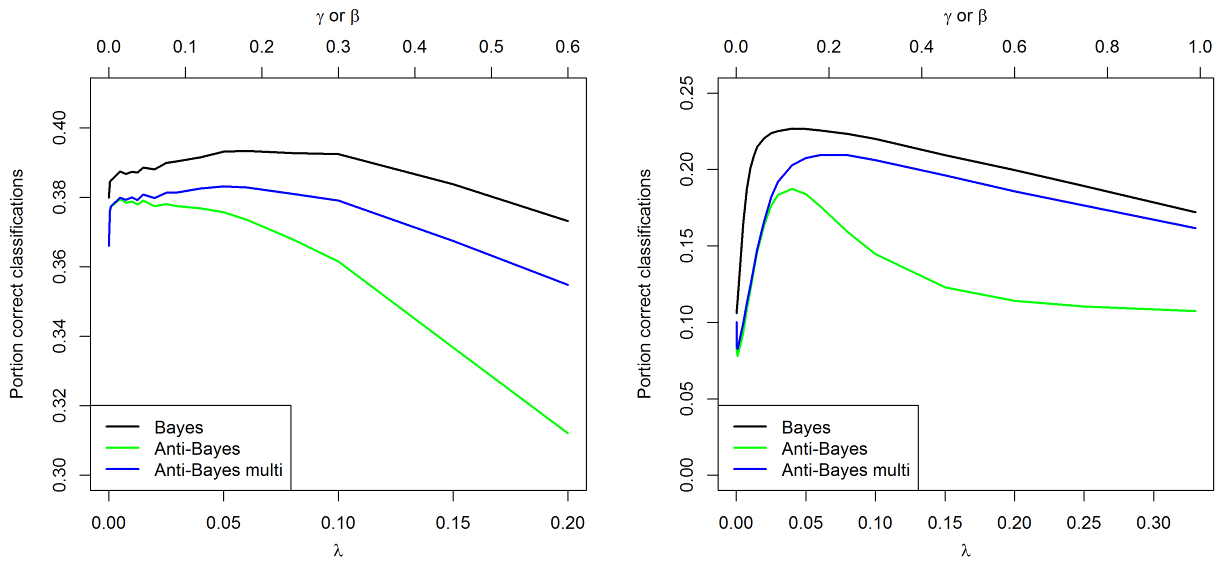

Figure 5: Classification performance for $K = 10$ classes. The panels show the portion of samples correctly classified for the Bayesian and AB algorithms for different choices of the tuning parameters. The left and the right panels display the cases for the shrink/expand and the switch processes, respectively. 'Anti-Bayes' and 'Anti-Bayes multi' refer to the AB approaches using DUMIQE and MDUMIQE respectively, to track the quantiles.

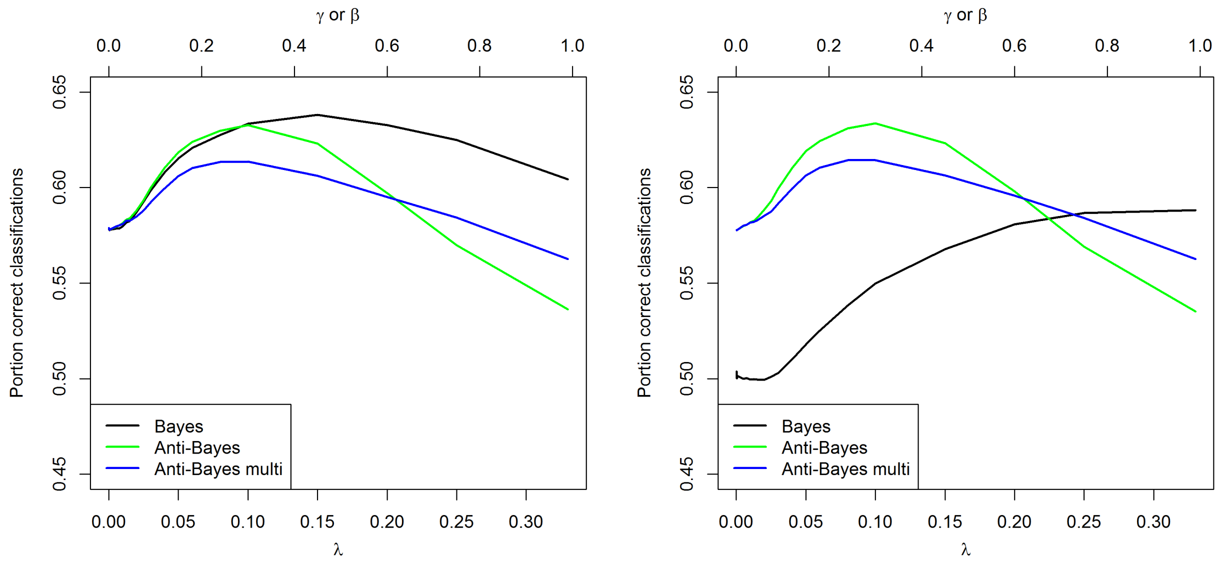

Figure 6: Jump process with outliers. The figure shows a portion of samples correctly classified for the Bayesian and AB algorithms for different choices of the tuning parameters for $K = 2$ classes. The left and the right panels show cases with outliers of size $\delta = 10$ and $10\,000$, respectively. The 'Anti-Bayes' and 'Anti-Bayes multi' curves refer to the AB approaches using the DUMIQE and MDUMIQE respectively, to track the quantiles.

with outliers as the performance is identical in the left panel of Figure 4 and in both of the panels in Figure 6.

## 8 Experimental Results: Real-life Data

We now compare the performance of the Bayesian and the AB algorithms for two real-life data sets.

### 8.1 Meteorological Data

The data used in this example were the observed outdoor air temperatures from two different locations in Norway, namely Alta and Rena. These were recorded every day at 1 PM from December 1, 1963 to January 31, 2013. Alta is in the far north in Norway and has a cold coastal climate, while Rena is further south in Norway and far inland. The yearly average temperature is about the same for the two locations. Figure 7 shows a plot of the temperatures for the two locations for three arbitrary selected years. One easily observes that the temperature varies a little more during the year in Rena compared to Alta, and as expected, temperature variations are, typically, less along the coast (due to the effect of the sea) compared to inland (inland climate).

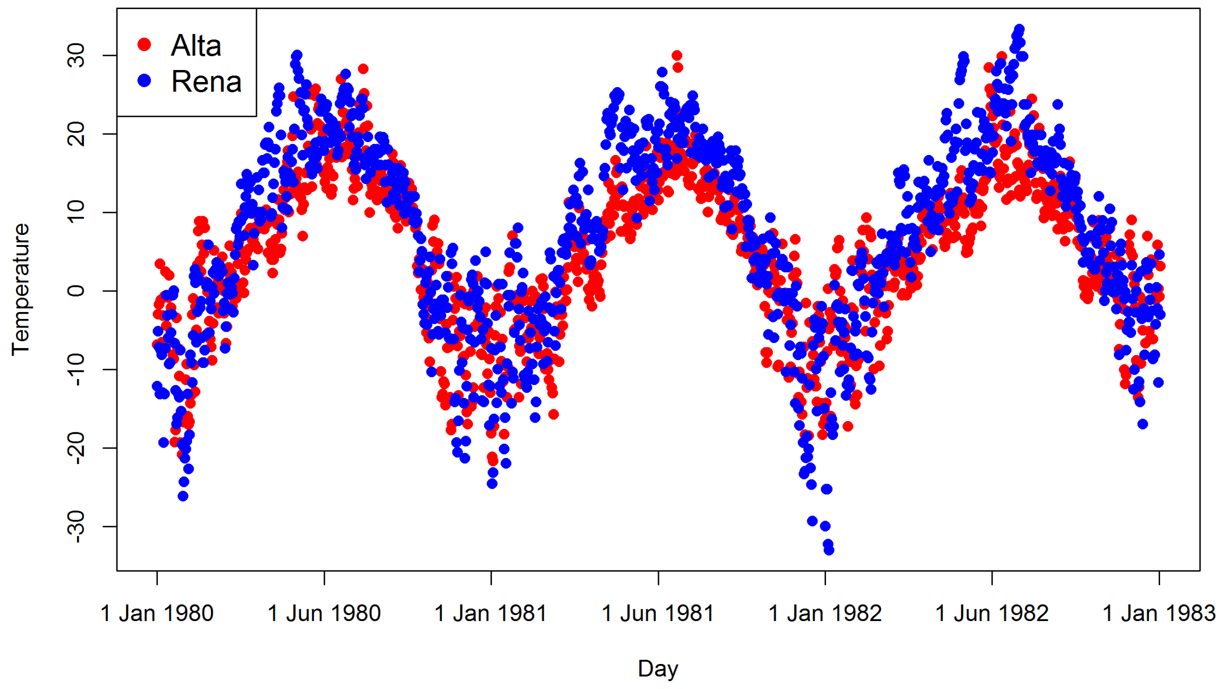

Figure 7: Outdoor air temperatures at the locations Rena and Alta from 1 January 1980 to 31 December 1982.

Assume now that we received observations of the outdoor air temperature in an online manner, except that the information about the location (Alta or Rena) from where the observations are received, appeared with a delay. The classification problem, therefore, was to classify the location of the temperature observations with unknown location. From Figure 7, we see that the temperature time series for the two locations are highly overlapping, making this a hard classification problem.

Our task was to evaluate the classification performance of the AB approach using both DUMIQE and MDUMIQE to track the quantiles, and the Bayesian approach, and for these we used different values of the tuning parameters $\lambda$ (DUMIQE), $\beta$ (MDUMIQE) and $\gamma$ (EMA). We assumed a delay of up to ten days to receive the true locations of the observations. The classification results are shown in Figure 8.

We see that the Bayesian approach with a high value of $\gamma$ (rapid updates), classified very well when the information about the location was received within a single day's delay. However, when the delay became larger, the performance dropped dramatically. With a delay of more than a single day, the three approaches classified about equally well. It is amazing that overall, the AB approach that tracks the quantiles using MDUMIQE (Anti-Bayes multi) was the best alternative. The approach performed well for all choices of $\beta$ which is important for dynamical systems. Of course, one may estimate a suitable choice of the tuning parameter from historical data, but for many dynamical systems this works poorly since the properties of the dynamical system may change dramatically and a choice of the tuning parameter performing well a few time steps ago, may perform poorly at the current time step. Consequently, an algorithm that is not too sensitive on the choice of the tuning parameter is important, rendering the Anti-Bayes MDUMIQE the preferable choice among the three algorithms.

## 8.2 Dynamical Classification of Tweets

On July 22, 2011, Norway was hit by a terrible terrorist act. In the aftermath of the terrorist act, a large number of the tweets in Norway were related to the terror attack. In this example, our goal was to filter out tweets related to the terror from the other tweets that were posted. We assumed that every tweet was labeled as being related to the concept of terror or about something else, but with some delay. One could, for example, imagine a panel that manually annotated the tweets, but that the annotation was, realistically, delayed when compared to when the tweets themselves were posted. One can clearly see this as a real-life classification problem over a data stream with natural delays, as described in Section 6.

The data set analyzed in this example was extracted from a large data set consisting of every single tweet posted in Norway in the aftermath of the above-mentioned incident (all the way to August 28, 2011). To perform the classification experiment we required tweets with their associated labels (i.e., whether the tweet was 'related to terror' or 'about something else'). These where constructed as follows:

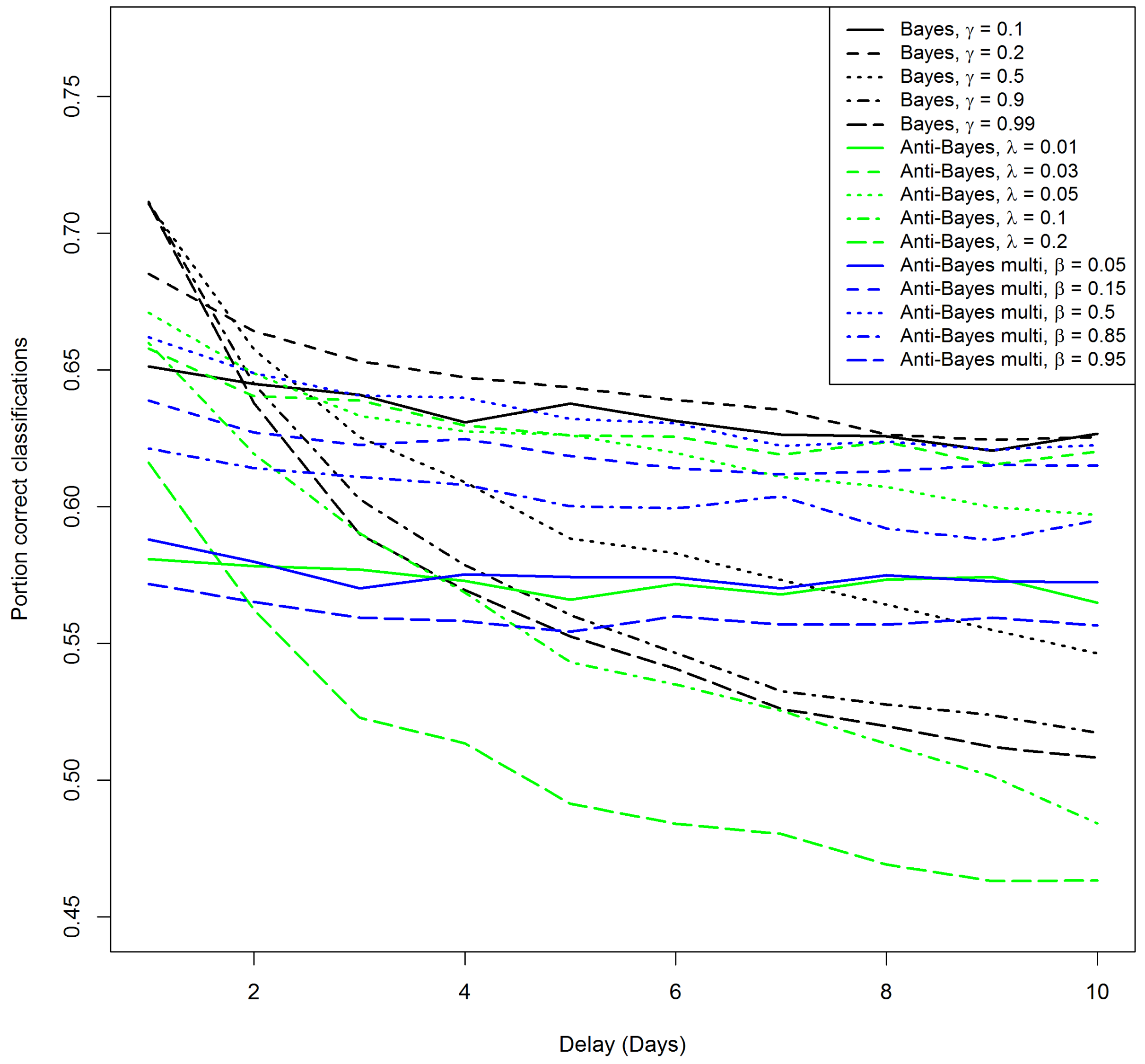

Figure 8: Portion of samples correctly classified for the Bayesian and AB algorithms. ‘Anti-Bayes’ and ‘Anti-Bayes multi’ refer to the AB approaches that use DUMIQE and MDUMIQE respectively, to track the quantiles.

1. Every tweet with at least one hash tag was filtered out.

2. Every hashtag occurring at least ten times in the data set was manually examined. Every hashtag was classified as being either 'related to terror' or 'about something else'. For a few hashtags, it was difficult to decide what the true 'class' was, and they were not involved in the classification process.

3. Every tweet with at least one of the annotated hashtags was selected, and this resulted in a set of annotated tweets. Of course, in the experiments conducted, the hashtags were removed from the tweets and the rest of the text in the tweets were used to achieve the classifications.

4. Finally, to render the tweets unique, all the 'retweets' were discarded, and we only included the original tweets.

After the above "data cleaning" phase, the data set consisted of 64,461 tweets.

The next challenge we faced was to design an appropriate classification procedure for tweets. Oommen *et al.* [21] presented an AB approach for classifying text documents. The approach was based on the distribution of the number of occurrences of words over documents within a class. They computed the quantiles of these word distributions over documents, and this formed the basis for their AB classification scheme. Unfortunately, the approach of [21] would not work well for tweets since, unlike documents, they consisted of very few words, and consequently, the number of occurrences of a specific word in a given tweet was almost always either zero or unity. Computing the quantiles for such distributions was thus a meaningless proposition.

A natural modification of the approach reported in [21] involved using a weighted sum of the words in a tweet and applying the distribution of this sum over the documents within a class as the basis for Bayesian and AB classification schemes. More specifically, let $\psi_k(t_i)$ denote the number of occurrences of a word $w_k, k \in \{1, 2, \ldots, W_i\}$ for a tweet posted at time $t_i$, where $W_i$ denotes the number of unique words observed in the tweets up to time $t_i$, i.e. the vocabulary. Further, let $c(t_i)$ denote the class label (i.e., tweet 'related to terror' or 'about something else'). Based on the reasoning above, we performed the classifications in this example based on a weighed sum of the word frequencies of a tweet given as:

$$x(t_i) = \alpha_0(t_i) + \alpha_1(t_i)\psi_1(t_i) + \alpha_2(t_i)\psi_2(t_i) + \cdots + \alpha_{W_i}(t_i)\psi_{W_i}(t_i). \tag{7}$$

Instead of using the word frequencies, $\{\psi_i(t_i)\}$ directly, as given by Eq. (7), we could, of course, have used a statistic of the word frequencies like the popular Term Frequency-Inverse Document Frequency (TF-IDF)[4] [16]. In our investigations, we explored three different approaches to determine the suitable values for the weights $\alpha_k(t_i)$, $k \in \{1, 2, \ldots, W_i\}$.

[4]This quantity was also considered in [21].

1. **Keywords:** In this approach, we selected a set of important words for the two classes, namely, 'terror' and 'other tweets'. The keywords were found by computing the Information Gain[5] (IG) for each of the words in the vocabulary up to time $t$. We may expect that the most-recently posted tweets are the most relevant, and we thus weighted the occurrences of the words in a tweet with respect to the index, time. More specifically, when computing the IG of different words at time $t$, the number of occurrences of a word at time $s$ were weighted with an exponential decay, i.e., instead of using $\psi_i(s)$ directly, we rather used $\exp\{-\rho(t-s)\}\psi_i(s)$ when computing the IG. In this expression, we used a value of $\rho$ such that the exponential decay was reduced from 1 to 0.01 in five days. Such a choice performed well in our experiments. We chose a threshold in the IG such that we retained 1,000 words. For these words, we set the weights, $\alpha_k(t_i)$, $k \in \{1, 2, \ldots, W_i\}$, equal to unity, and for the other words in the vocabulary, we set the weights to be zero. It should be mentioned that the IG could be computed recursively when new tweets were received, and this was a valuable phenomenon for the online classification problem considered in this example.

2. **Weighted Keywords:** Instead of setting the weight to unity for the 1,000 words, we set the weights equal to the computed IG. Let $IG_i(t)$ denote the IG of word $w_i(t_i)$. Assume that $w_i(t_i)$ is one of the 1,000 words with the largest IG at time $t_i$. If the word occurred in a larger portion of the tweets related to the class 'related to terror' than the other class, we set $\alpha_i(t_i) = IG_i(t_i)$. Alternatively, if $w_i(t_i)$ occurred in a larger portion of the tweets related to the class 'other tweets', we set $\alpha_i(t_i) = -IG_i(t_i)$. All the other weights were set to zero.

3. **Ridge regression:** In this setting, we interpreted the classification problem as a linear regression problem. Here, we assumed that we had tweets with known class labels at time points $t_1, t_2, \ldots, t_i$. Formulated as a regression problem, we obtained the following equations at time $t_i$ for the weights:

$$
\begin{aligned}
c(t_1) &= \alpha_0(t_1) + \alpha_1(t_i)\psi_1(t_1) + \ldots \alpha_2(t_i)\psi_2(t_1) + \cdots + \alpha_{W_i}(t_1)\psi_{W_i}(t_1), \\
c(t_2) &= \alpha_0(t_2) + \alpha_1(t_i)\psi_1(t_2) + \ldots \alpha_2(t_i)\psi_2(t_2) + \cdots + \alpha_{W_i}(t_2)\psi_{W_i}(t_2), \\
&\vdots \qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
c(t_i) &= \alpha_0(t_i) + \alpha_1(t_i)\psi_1(t_i) + \ldots \alpha_2(t_i)\psi_2(t_i) + \cdots + \alpha_{W_i}(t_i)\psi_{W_i}(t_i).
\end{aligned}
$$

Similar to the keyword and weighted keyword approaches described above, in this case, we set

[5] In a comparative study of different feature classification methods, Yang and Pedersen [36] showed that the Information Gain was effective in aggressive term removal without resulting in a loss of the corresponding classification accuracy.

all the weights equal to zero except for the 1,000 words with the highest IG. In spite of this, the problem required the estimation of a large number of weights, and we observed that it was not possible to estimate all the parameters using a traditional least squares approach. We, therefore, had to resort to a ridge regression methodology, which is, in principle, equivalent to an ordinary least squares approach augmented with a Tikhonov penalty, which implies that we sought for the weights that minimized the following expression:

$$\underset{\alpha_0(t_i),\ldots,\alpha_{W_i}(t_i)}{\arg\min} \sum_{j=1}^{i} e^{-\rho(t_i-t_j)} \left( c(t_j) - \sum_{k=1}^{W_i} \alpha_k(t_i)\psi_k(t_j) \right)^2 + \sum_{k=0}^{W_i} \alpha_k(t_i)^2,$$

where $e^{-\rho(t_i-t_j)}$ reduced the importance, with respect to time, since a tweet was posted. Further, $\sum_{k=0}^{W_i} \alpha_k(t_i)^2$ were the Tikhonov penalties ensuring unique and robust estimates of the weights $\alpha_0(t_i),\ldots,\alpha_{W_i}(t_i)$. A beautiful property of the ridge regression was that the estimates could be updated recursively as new data was received[6], and this characteristic was very useful for the online problem considered in this example.

To now achieve the desired classification, we assumed that we received the class labels with a delay, i.e., at time $t_i + h$ we had only received class labels up to time $t_i$. By using Eq. (7), we encountered the precise framework of Section 6, except that in this example, the difference between the time points of observation were not equidistant. One way to resolve this was to use small values of $\lambda, \beta$ or $\gamma$ in the update schemes in Section 5 if the time differences in posting times were small, and to increase these values of $\lambda, \beta$ or $\gamma$ if the differences in posting times increased[7].

In the few hours right after the terrorist act, almost very tweet in Norway was about the terror, and at the end of the test period (towards August 28, 2011), almost every tweet was about something else. It is evident that both of this extreme cases are not suitable when evaluating classification performance of an algorithm since a simplistic strategy to merely classify every tweet to the same appropriate class would yield an exceptional accuracy. Thus, while we ran the algorithm over the whole time span (July 22 to August 28, 2011), the classification performance was only measured for the time interval after the terrorist act when the portion of tweets related to the terror was between 30% and 70%. The resulting time interval was from 00:51 Central European Summer Time, July 23, 2011, to 00:59 Central European Summer Time, 30 July, 2011. This resulted in about a week of observations starting from about 11 hours after the bomb hit Oslo.

We now ran the algorithms in Section 6 by tracking the mean values and the quantiles of the

[6]Since the response, $c(t_1),\ldots,c(t_i)$, were dichotomous variables, a natural alternative to the linear ridge regression, as suggested above, would have been to use logistic ridge regression. A disadvantage with logistic regression is that the estimates could not have been updated in a recursive way as we were able to do for linear ridge regression. In addition, for text classification, the literature records that linear regression can achieve a similar classification performance as the logistic regression does [18].

[7]As it stands now, we have not looked into this. Rather, we used the same values of $\lambda, \beta$ or $\gamma$ over time.

distribution of $x(t_1), x(t_2), \ldots, x(t_i)$ from Eq. (7) for the two classes 'tweets about terror' and 'other tweets'. We chose a large set of different values for the tuning parameters $\lambda, \beta$ or $\gamma$ to evaluate the algorithms' peak performance and robustness. When running the experiments, we updated the values of the IG and the corresponding weights based on the ridge regression after every 30 minutes. Further, we performed a classification of the tweets from time $t_i$ up to time $t_i + h$ after every 10 minutes.

The results for the three approaches 'Keywords', 'Weighted keywords' and 'Ridge regression' are shown in Figures 9, 10 and 11, respectively. From the figures, we see that we assumed a delay in receiving the class labels of up to 24 hours.

For the 'Keywords' approach (Figure 9), we see that the best performance was achieved for the Bayesian approach using $\gamma = 5 \cdot 10^{-5}$ and the AB MDUMIQE with $\beta = 10^{-5}$. Similar to the previous example in Section 8.1, the MDUMIQE seemed to estimate well for a large range of values of the tuning parameter, and was more robust than the other two estimation strategies.

For the 'Weighted Keywords' approach (Figure 10), we see that the AB MDUMIQE strategy clearly outperformed the two other methods both when it concerned the peak performance and the robustness for the choice of the tuning parameter. The fact that an AB scheme clearly outperformed a Bayesian mechanism is a phenomenal discovery. Further, we see that the different estimation strategies that used the 'Weighted Keywords' approach did not perform as well as those that used the 'Keywords' approach. It was also surprising that the prediction performance increased with time from 20 to 24 hours into the future. This was probably because there were daily seasonal patterns in the twitter data, making it easier to predict 24 hours into the future than, say 15 hours.

For the 'Ridge Regression' approach (Figure 11), we again claim the fascinating result that AB MDUMIQE recorded the best peak performance. In addition, astonishingly, it was again the most robust method when it concerned the choice of the tuning parameter.

## 9 Open and Unresolved Issues

One will easily appreciate that since these are the first reported results that deal with the AB methods for evaluating quantiles and their application in classifying dynamic data streams, it leads to many open and unresolved problems[8] that beg attention. Some of these problems are listed below:

1. The first question that remains unresolved is that of making the parameters of DUMIQE and MDUMIQE ($\lambda$ and $\beta$) to be adaptive. Since the distances between the different samples from the respective quantile being considered may be different, it should be possible to conceive of a scheme that possesses different step sizes to update the quantiles. In other words, one could utilize a step size $\lambda$ that is proportional to the distance between the current sample and the

[8] We are extremely grateful to the anonymous Referees who brought these problems to our attention.

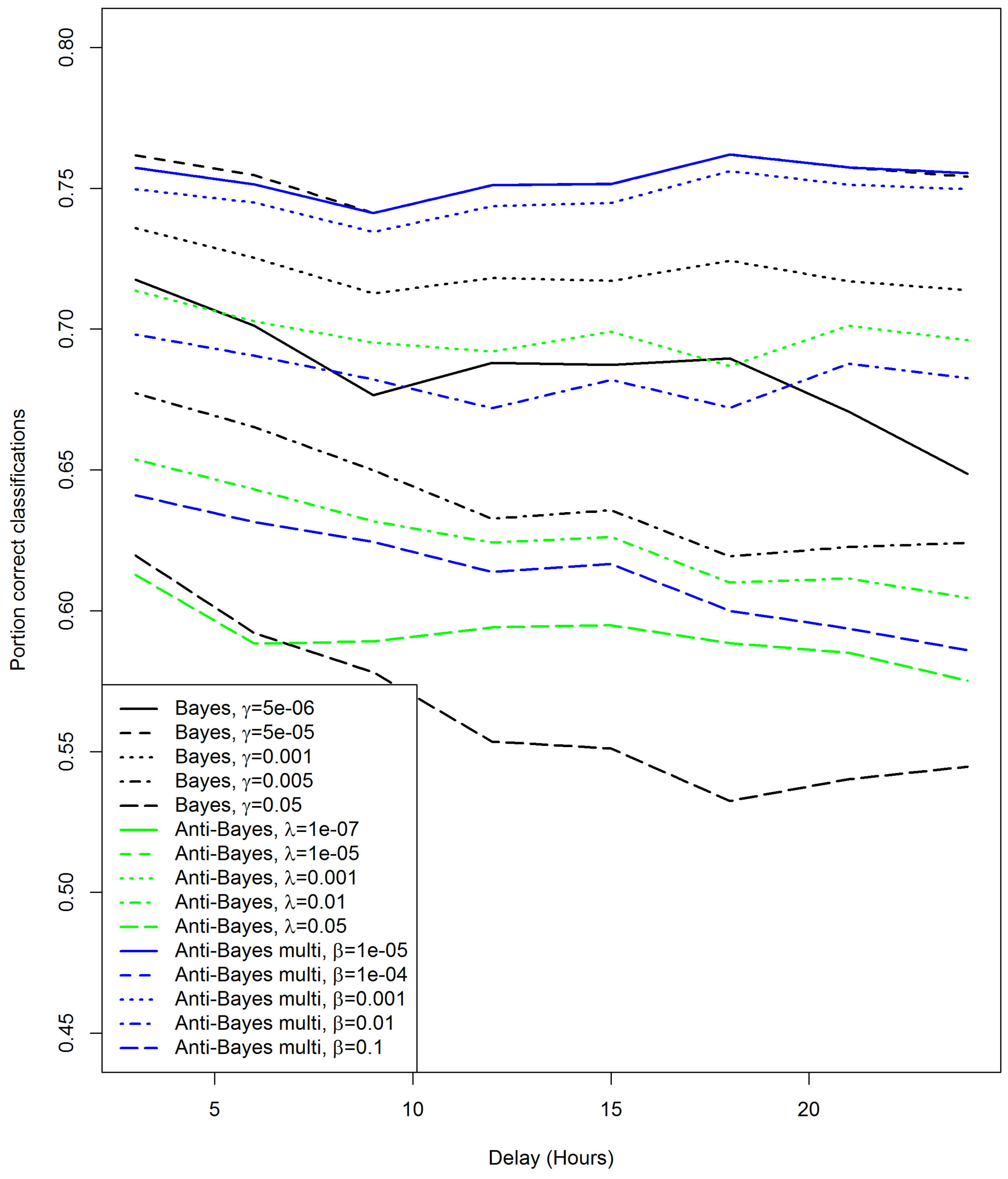

Figure 9: Twitter data example: Portion of samples correctly classified for the Bayesian and AB algorithms using the 'Keywords' approach. 'Anti-Bayes' and 'Anti-Bayes multi' refer to the AB approached using DUMIQE and MDUMIQE respectively, to track the quantiles.

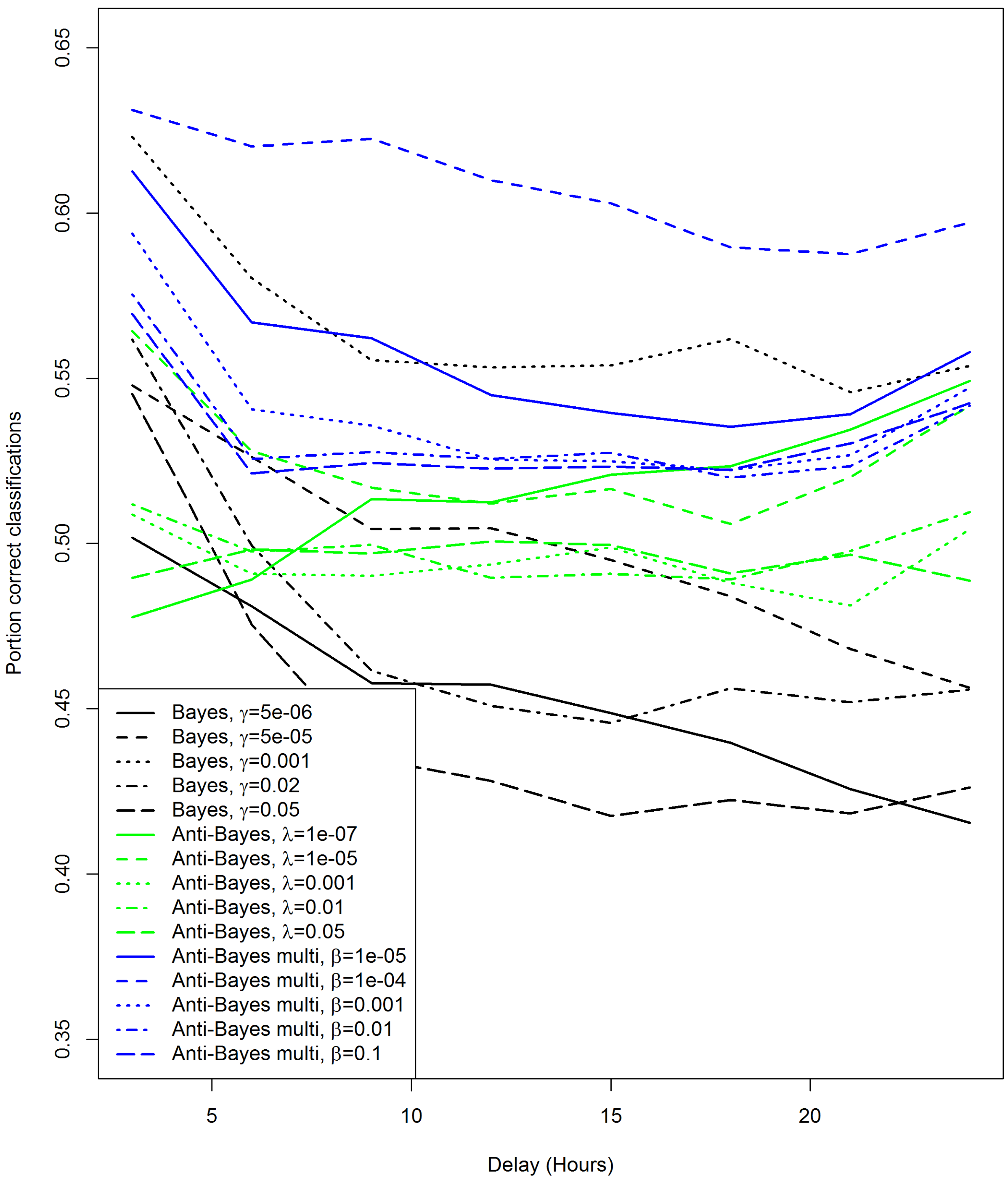

Figure 10: Twitter data example: Portion of samples correctly classified for the Bayesian and AB algorithms using the 'Weighted Keywords' approach. 'Anti-Bayes' and 'Anti-Bayes multi' refer to the AB approached using DUMIQE and MDUMIQE respectively, to track the quantiles.

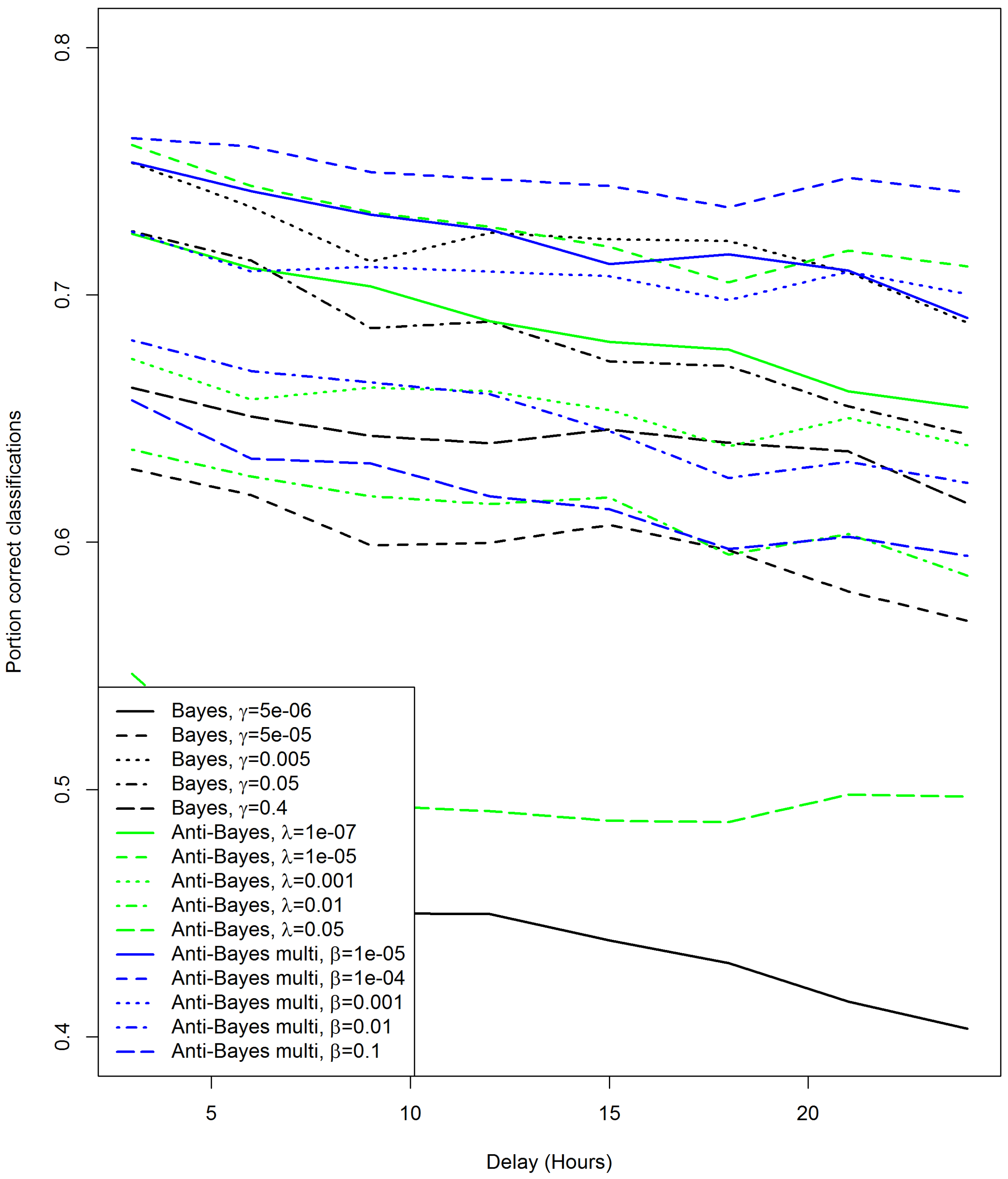

Figure 11: Twitter data example: Portion of samples correctly classified for the Bayesian and AB algorithms using the 'Ridge Regression' approach. 'Anti-Bayes' and 'Anti-Bayes multi' refer to the AB approached using DUMIQE and MDUMIQE respectively, to track the quantiles.

quantile being examined. While this is, in fact, a very reasonable option, the way by which these parameters are changed, so as to ensure convergence, is non-trivial. If the step size is too large, the scheme may not converge, and on the other hand, if it is too small, the convergence could be very sluggish. This issue should be considered open. Secondly, using the values of the samples directly to update the quantile estimates, makes the method more vulnerable to outliers then the DUMIQE and MDUMIQE introduced in this paper.

2. In Section 6.2, the algorithm that we had proposed indicated that the model would be updated $h$ times until the true class label was revealed. Since there would be some samples misclassified during these $h$ iterations (which would result in an incorrect update of the model/distributions), one can again conceive of a so-called "delayed" supervised mode of operation that adaptively corrects the model/distributions. Such a "delayed" supervised strategy would be a completely new mechanism – distinct from what we have proposed in this paper. However, the question of knowing how we could use these delayed responses in formulating the quantile updates and also of achieving the classification strategy remains unknown. While one could easily design naive solutions, the difficult part would be to confirm their performances, and so this will also serve as a rich avenue for future research.

3. The DUMIQE/MDUMQE have no problems tracking quantiles of multi-peak distributions, and thus the methods suggested in this paper should have no problems with multi-peak distributions.

4. The AB strategy can also be extended to high dimensional data. Once can see a proof of this, for example, in our most-recent clustering paper [10].

## 10 Conclusions

In this paper we have developed methods that apply the Bayesian and the recently-proposed Anti-Bayesian (AB) classification framework to perform online classifications for dynamic data streams. The classification of such dynamical data streams is among the most complex problems encountered in classification. This is, firstly, because the distribution of the data streams is non-stationary, and it changes without any prior "warning". Secondly, the manner in which it changes is also unknown. Thirdly, and more interestingly, we invoked the model with the assumption that the correct classes of previously-classified patterns become available at a juncture after their appearance. Apart from Bayesian methods, this paper pioneered the use of unreported novel schemes using AB techniques. Contrary to the Bayesian paradigm that compare the testing sample with the distribution's central points, AB techniques are based on the information in the distant-from-the-mean samples.

In this paper, the AB classification framework was based on estimating the time-varying quantiles of the distributions for the different classes. In this context, when performing AB classification for dynamic data streams, we tracked the quantiles using the DUMIQE and MDUMIQE methods developed in [12, 37]. By virtue of the design of the quantile estimator, the AB approach was shown to be more robust against outliers, which is a property absent in the Bayesian approach that tracks the mean. Both approaches were tested using both synthetic and real-life data. In the real-life examples, the AB approaches performed very well, and in most cases outperformed the Bayesian analog both with respect to peak performance and the robustness with respect to the tuning parameters.

## References

[1] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. The r*-tree: an efficient and robust access method for points and rectangles. In *ACM SIGMOD Record*, volume 19, pages 322–331. Acm, 1990.

[2] A. Bifet. *Adaptive Learning and Mining for Data Streams and Frequent Patterns*. PhD thesis, Departament de Llenguatges i Sistemes Informatics, Universitat PolitÃÍcnica de Catalunya, Barcelona Area, Spain, 2009.

[3] A. Bifet and R. Gavalda. Kalman Filters and Adaptive Windows for Learning in Data Streams. In L. Todorovski and N. Lavrac, editors, *Proceedings of the 9th Discovery Science*, volume 4265, pages 29–40, 2006.

[4] A. Bifet and R. Gavalda. Learning From Time-changing Data With Adaptive Windowing. In *Proceedings SIAM International Conference on Data Mining*, volume 8, pages 443–448, 2007.

[5] J. Cao, L. E. Li, A. Chen, and T. Bu. Incremental tracking of multiple quantiles for network monitoring in cellular networks. In *Proceedings of the 1st ACM workshop on Mobile internet through cellular networks*, pages 7–12. ACM, 2009.

[6] F. Chen, D. Lambert, and J. C. Pinheiro. Incremental quantile estimation for massive tracking. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 516–522. ACM, 2000.

[7] J. G. De Gooijer and R. J. Hyndman. 25 years of time series forecasting. *International journal of forecasting*, 22(3):443–473, 2006.

[8] P. Domingos and G. Hulten. Mining high-speed data streams. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 71–80. ACM, 2000.

[9] J. Gama, P. Medas, G. Castillo, and P. Rodrigues. Learning with Drift Detection. In A. Bazzan and S. Labidi, editors, *Advances in Artificial Intelligence âĂŞ SBIA 2004*, volume 3171 of *Lecture Notes in Computer Science*, pages 286–295. Springer Berlin Heidelberg, 2004.

[10] H. L. Hammer, A. Yazidi, and B. J. Oommen. “Anti-Bayesian” flat and hierarchical clustering using symmetric quantiloids. *Information Sciences*, 418:495–512, 2017.

[11] H. L. Hammer, A. Yazidi, and B. J. Oommen. On Using Novel “Anti-Bayesian” Techniques for the Classification of Dynamical Data Streams. In *IEEE Congress on Evolutionary Computation (CEC)*, 2017.

[12] H. L. Hammer, A. Yazidi, and H. Rue. Estimation of multiple quantiles in dynamically varying data streams. *arXiv preprint arXiv:1702.00046*, 2017.

[13] S. S. Haykin. *Adaptive filter theory.* Pearson Education India, 2008.

[14] T. Kailath, A. H. Sayed, and B. Hassibi. *Linear estimation*, volume 1. Prentice Hall Upper Saddle River, NJ, 2000.

[15] D. Lowne, S. J. Roberts, and R. Garnett. Sequential non-stationary dynamic classification with sparse feedback. *Pattern Recognition*, 43(3):897–905, 2010.

[16] C. D. Manning and H. Schütze. *Foundations of statistical natural language processing*, volume 999. MIT Press, 1999.

[17] M. Mayo and A. Bifet. Deferral classification of evolving temporal dependent data streams. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, pages 952–954. ACM, 2016.

[18] L. Miratrix, J. Jia, B. Gawalt, B. Yu, and L. El Ghaoui. Summarizing large-scale, multiple-document news data: sparse methods and human validation. *UC Berkeley Dept. of Statistics Technical Report: 801*, 2011.

[19] H.-L. Nguyen, Y.-K. Woon, and W.-K. Ng. A survey on data stream clustering and classification. *Knowledge and information systems*, 45(3):535–569, 2015.

[20] B. J. Oommen, R. Khoury, and A. Schmidt. Text classification using novel "anti-bayesian" techniques. In *Computational Collective Intelligence*, pages 1–15. Springer, 2015.

[21] B. J. Oommen, R. Khoury, and A. Schmidt. Text classification using novel "anti-bayesian" techniques. In *Computational Collective Intelligence*, pages 1–15. Springer, 2015.

[22] B. J. Oommen and A. Thomas. "Anti–Bayesian" parametric pattern classification using order statistics criteria for some members of the exponential family. *Pattern Recognition*, 47(1):40–55, 2014.

[23] R. Pears, S. Sakthithasan, and Y. S. Koh. Detecting Concept Change in Dynamic Data Streams. *Machine Learning*, 97(3):259–293, 2014.

[24] C. O. Plumpton, L. I. Kuncheva, N. N. Oosterhof, and S. J. Johnston. Naive random subspace ensemble with linear classifiers for real-time classification of fmri data. *Pattern Recognition*, 45(6):2101–2108, 2012.

[25] F. Saki and N. Kehtarnavaz. Online frame-based clustering with unknown number of clusters. *Pattern Recognition*, 57:70–83, 2016.

[26] T. Seidl, I. Assent, P. Kranen, R. Krieger, and J. Herrmann. Indexing density models for incremental learning and anytime classification on data streams. In *Proceedings of the 12th international conference on extending database technology: advances in database technology*, pages 311–322. ACM, 2009.

[27] V. M. Souza, D. F. Silva, J. Gama, and G. E. Batista. Data stream classification guided by clustering on nonstationary environments and extreme verification latency. In *Proceedings of the 2015 SIAM International Conference on Data Mining*, pages 873–881. SIAM, 2015.

[28] H. Tavasoli, B. J. Oommen, and A. Yazidi. On the online classification of data streams using weak estimators. In *Trends in Applied Knowledge-Based Systems and Data Science - 29th International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2016, Morioka, Japan, August 2-4, 2016, Proceedings*, pages 68–79, 2016.

[29] A. Thomas and B. J. Oommen. The fundamental theory of optimal "anti-bayesian" parametric pattern classification using order statistics criteria. *Pattern Recognition*, 46(1):376–388, 2013.

[30] A. Thomas and B. J. Oommen. A novel border identification algorithm based on an "anti-bayesian" paradigm. In *Computer Analysis of Images and Patterns*, pages 196–203. Springer, 2013.

[31] A. Thomas and B. J. Oommen. Order statistics-based parametric classification for multi-dimensional distributions. *Pattern Recognition*, 46(12):3472–3482, 2013.

[32] A. Thomas and B. J. Oommen. Ultimate order statistics-based prototype reduction schemes. In *Proceedings of AI'13, the 2013 Australasian Joint Conference on Artificial Intelligence*, pages 421–433. Springer, December 2013.

[33] L. Tierney. A space-efficient recursive procedure for estimating a quantile of an unknown distribution. *SIAM Journal on Scientific and Statistical Computing*, 4(4):706–711, 1983.

[34] I. W. Tsang, A. Kocsor, and J. T. Kwok. Simpler core vector machines with enclosing balls. In *Proceedings of the 24th international conference on Machine learning*, pages 911–918. ACM, 2007.

[35] G. Widmer and M. Kubat. Learning in the Presence of Concept Drift and Hidden Contexts. In *Machine Learning*, volume 23, pages 69–101, 1996.

[36] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *ICML*, volume 97, pages 412–420, 1997.

[37] A. Yazidi and H. L. Hammer. Multiplicative Update Methods for Incremental Quantile Estimation. *IEEE Transactions on Cybernetics (submitted)*, 2017.