International Journal of Network Dynamics and Intelligence Scilight

Article

Online Learning of Bayesian Classifiers with Nonstationary Data Streams

Peng Wu [1,2,*], and Ning Xiong [1]

[1] School of Innovation, Design and Engineering Mälardalen University, Västerås 721 23, Sweden
[2] Department of Computer Engineering, Taiyuan Institute of Technology, Taiyuan 030008, China
[*] Correspondence: 14112078@bjtu.edu.cn

Received: 31 March 2023
Accepted: 13 July 2023
Published: 26 September 2023

Abstract: The advancement in Internet of things and sensor technologies has enabled data to be continuously generated with a high rate, i.e., data streams. It is practically infeasible to store streaming data in a hard disk, and apply a traditional batch learning method to extract a relevant knowledge model from these data. This paper studies online incremental learning with data streams, in which one sample is processed at each time to update the existing model. For the learning target, the Bayesian classifier is adopted which is a computationally economical model of easy deployment for online processing in edges or devices. By using the individual new example, we first present an online learning algorithm to incrementally update classifier parameters in a way equivalent to the offline learning counterpart. In order to adapt to concept drifts in nonstationary environments, the proposed online learning algorithm is improved to enable recent examples to be more impactful during the sequential learning procedure. Preliminary simulation tests reveal that the improved online learning algorithm can lead to faster model adaption than the unimproved online algorithm when the data drift occurs. In case of presumed stationary data streams without drifts, the improved online algorithm is proved to be competent by performing at least as good as (sometimes, even better than) the unimproved algorithm.

Keywords: incremental learning; online learning; Bayesian classifier; concept drift

1. Introduction

The advancement in the Internet of things (IoT) and sensor technologies has enabled an exponential growth of data which is generated and collected from a wide range of sources. Understanding big volumes of data [1] requires cutting-edge tools and techniques to extract useful knowledge and information for decision support. Data being continuously generated at a high rate is referred to as the streaming data (or the data stream) and is of high interest [2]. Data streams are prevalent in many industrial domains such as manufacturing and automatic control systems. As streaming data contains a huge number or even an unbounded sequence of samples, it is practically infeasible to store the full volume of data on a hard disk, and access the data multiple times to extract relevant knowledge.

So far, the main stream of the machine learning methods has been focused on a batch-based and offline learning manner. A limitation of offline learning methods is that, with the arrival of a set of new examples, the learning algorithm has to be re-executed using all training data available. Hence, it can lead to an inefficient learning process without utilizing the knowledge learned previously. Particularly, offline learning is incapable of handling data streams in an online environment, in which fresh data is generated continuously with open ends.

Incremental learning [3] is a promising technology to deal with online data streams, and aims to update an existing knowledge model based on learning from new data rather than scratches. This basic idea shows analogy to the cognitive behavior (of humans) which is used to acquire knowledge in an incremental fashion over time. One of the earliest incremental learning algorithms was proposed in [4] to solve data clustering problems. According to the definition given in [5], an incremental algorithm should be computationally efficient when incorporating the experience (training data) into models, and should not use any unreasonable space to store the already used experience. This implies that training examples have to be discarded immediately after being processed, leading to rather low memory demands during the learning procedure.

Th every fast decision tree (VFDT) algorithm [6, 7] was developed for incremental learning of a decision tree

based on streaming data. The tree is learned by recursively replacing leaves with decision nodes. The Hoeffding bounds are used in the VFDT to decide when to install a split-test at a leave. Later Rutkowski et al. [8] proposed a technique to use the Gaussian approximation to select the best splitting attribute in a considered node. It is proved that, by using this technique, the best selected attribute has a high probability to be identical to the attribute derived from the whole data stream. However, no pruning strategies are addressed in the above mentioned papers to prune the underlying tree and avoid over-fitting during the learning procedure.

Wang, Minku and Yao developed two online stream mining methods, i.e. the oversampling based online bagging (OOB) and under-sampling based online bagging (UOB) [9], aiming to bias the learning to favor the minority class. Note that the adaptability of varying the imbalance status is supported by the incremental estimation of the occurrence probabilities of classes, and this provides basis to adjust the sampling rate in a dynamic process. A limitation of the OOB and UOB methods is that they can only handle data streams of two classes.

Yavtukhovskyi et al. proposed an incremental learning method to construct fuzzy classification rules from data streams [10]. This method not only generates new fuzzy rules to reflect information from new data, but also incrementally updates confidence in the old rules. When the existing knowledge is applied to classify a new instance, the low-confidence rules will not be used in the fuzzy rule based inference. Similar results were conducted using evolving fuzzy classifiers [11−15] that emerged as an own research line in recent years. However, both incremental and evolving fuzzy classifiers treat training samples equally in model construction, regardless of their arrival time. Hence, such classifiers are not well suitable for nonstationary environments, where recently generated data should weigh more than those data generated from the long past.

Incremental learning is also exploited to increase the computational efficiency when learning from big data sets. The work in [16, 17] split the large volume of training data into multiple subsets, with each corresponding to a learning episode. In each episode, the knowledge acquired from the preceding stage is updated and refined based on the current data subset. Duda et al. [18] proposed a boosting based mini-batch training approach to accelerating the training process of deep neural networks. This approach differs from traditional learning methods in that 1) learning examples are randomly taken from the original training set to form a continuous data stream; 2) and the mini-batch is constituted by a group of recent elements from the stream for model update.

Online learning [19], as a closely related concept to incremental learning, represents a family of machine learning methods attempting to learn from a sequence of samples one by one at a time. The goal of online learners is to optimize the performance for solving a sequence of prediction problems by utilizing known answers to previous tasks. According to [20], incremental learning can work in either an online or a batch mode. In the online mode, only one example is used to update the model each time, while in the batch mode, a batch of multiple examples is used for model update at each time.

This paper focuses on incremental online learning of Bayesian classifiers which represent a class of computational cheap and probabilistic models for classification. Bayesian classifiers are appealing for employment in an online setting due to the following reasons. 1) The Bayesian classifiers involve no specification and tuning of the model architecture, which is yet common in many other machine learning models. Changing model architecture will terminate the sequential update procedure and request model training anew. 2) Online learning is often executed in edges or devices where the space capacity is limited. Fortunately, Bayesian classifiers are economic models with a low number of parameters, making their deployment easy on the edge or device levels. Therefore, we adopt the Bayesian classifier as the target model for online learning in this paper. We present an online learning algorithm that uses one sample (along the stream) at a time to update the parameters of the existing Bayesian classifier. It is also clearly shown that this online algorithm works equivalently to the offline learning counterpart that uses the entire data stream as the training data for once.

Further, we consider nonstationary environments in which the data distribution and characteristics may evolve with time, i.e., concept drifts. Learning to adapt to concept drifts is one of the grand challenges and is crucial for many real-world applications [21]. To this end, we propose an improved online learning algorithm for Bayesian classifiers, which enables recent examples to be more impactful during the sequential learning procedure. Preliminary simulation tests reveal that the improved online learning algorithm can lead to faster model adaption than the unimproved online algorithm when such a drift occurs in data. In case of presumed stationary data streams without concept drifts, this improved online algorithm is still competent by performing at least as good as (sometimes, even better than) the unimproved algorithm.

In summary, the novel contribution of this paper is highlighted from the following aspects.

1) We investigate online learning with Bayesian classifiers that are economic to be implemented in edge devices.

2) We propose the online learning algorithm (Algorithm 1), which is shown equivalent to the offline learning counterpart that uses the whole data stream for once.

3) We further propose the improved online learning algorithm (Algorithm 2), which enables faster model adaptation in nonstationary environments.

The remainder of the paper is organized as follows. Section 2 presents the basic principle of the naive Bayesian classifier. The online learning algorithms for Bayesian classifiers are elaborated in Section 3. Section 4 gives the preliminary results of simulation tests to evaluate the proposed learning algorithm. Finally, concluding remarks are given in Section 5.

2. Naive Bayesian Classifier

Naive Bayesian classifiers are simple probabilistic models which can be used to solve many classification problems. The basic idea is to consider the posterior probability for each probable class H_k (k=1...K) when attempting to classify a new instance, which is represented by the feature vector (x_1, x_2, ⋯, x_n). According to the Bayes theorem, this posterior probability is expressed by

$$P(H_k|x_1,x_2,\cdots,x_n)=\frac{P(H_k)P(x_1,x_2,\cdots,x_n|H_k)}{P(x_1,x_2,\cdots,x_n)} \tag{1}$$

Assuming that all features are independent of each other, Equation (1) can be further written as:

$$P(H_k|x_1,x_2,\cdots,x_n)=\frac{P(H_k)P(x_1|H_k)P(x_2|H_k)\cdots P(x_n|H_k)}{P(x_1,x_2,\cdots,x_n)} \tag{2}$$

The rationale of this independence assumption of features can be found in [22].

Since the denominator in Equation (2) is a constant given the feature values, we can decide the class y* that maximizes the numerator, i.e.,

$$y^*=\underset{k}{arg\max}P(H_k)P(x_1|H_k)P(x_2|H_k)\cdots P(x_n|H_k) \tag{3}$$

The likelihood $P\left(x_j|H_k\right)$ in Equation (3) needs to be derived from a predefined probability density function provided that feature x_j takes continuous values. This paper uses the Gaussian function as the density function to express the distribution of a feature under a given class. It was explained in [23] that the Gaussian function is able to approximate a wide range of distributions such as Gamma, Binnominal and Poisson distributions. Thereby, the likelihood $P\left(x_j|C_k\right)$ required by Equation (3) is computed as

$$P\left(x_j|H_k\right)=\frac{1}{\sqrt{2\pi}\sigma_{jk}}exp\left[-\frac{1}{2}\left(\frac{x_j-m_{jk}}{\sigma_{jk}}\right)^2\right] \tag{4}$$

where m_{jk} and σ_{jk} stand for the mean and standard deviation of the Gaussian function, respectively.

Let the training data set be formulated as $TD=\bigcup_{k=1}^{K}\{(\boldsymbol{X}_k(i),H_k)|i=1,\cdots,n_k\}$ with $\boldsymbol{X}_k(i)$ representing the feature vector of the ith training example of class H_k. n_k denotes the number of training examples of class H_k. $x_{jk}(i)$ refers to the jth element (the value of feature x_j) in the feature vector $\boldsymbol{X}_k(i)$. The prior probability $P(H_k)$ of class H_k and the mean and standard deviation of feature x_j (j=1⋯n) under class H_k can be obtained from the training data set, which are formulated as follows:

$$P(H_k)=\frac{n_k}{\sum_{t=1}^{K}n_t} \tag{5}$$

$$m_{jk}=\frac{1}{n_k}\sum_{i=1}^{n_k}x_{jk}(i) \tag{6}$$

$$\sigma_{jk}^2=\frac{1}{n_k}\sum_{i=1}^{n_k}\left(x_{jk}(i)-m_{jk}\right)^2 \tag{7}$$

which are considered as the parameters of the Bayesian classifier.

3. Online Learning Algorithms

This section discusses incremental online learning for Bayesian classifiers. In Subsection 3.1, we present the online learning algorithm that weighs all the samples in the streams equally. This algorithm is then improved in Subsection 3.2 to deal with the concept drift of nonstationary data streams in time-varying environments.

3.1. Online Learning with Stationary Data Stream

Online learning is performed to update the Bayesian classifier with each new training example. First, let's con-

sider stationary data streams in which the data characteristics and distribution remain unchanged with time. Hence, it is required to realize incremental calculation of the classifier parameters defined by Equations (5)–(7). This means that, given a new training example, the old classifier parameters have to be updated to new values which are the same as those obtained using all the old and new training samples together.

Let $P(H_k, i)$ be the prior probability of class H_k after incorporating the first i training samples. With the arrival of the next sample E_{i+1}, the class probability is updated by

$$P(H_k, i+1) = \frac{i \cdot P(H_k, i) + s}{i+1} = P(H_k, i) + \frac{1}{i+1}\left[s - P(H_k, i)\right] \tag{8}$$

where $s = (class(E_{i+1}) = H_k)$ is a logic variable returning 1 when E_{i+1} is of class H_k, and 0 otherwise.

Likewise, the mean of a feature under a class can be incrementally updated using examples of that class. Let $m_{jk}(i)$ be the mean of feature x_j under class H_k, which is estimated based on the first i samples of that class. Now, suppose that the next example of class H_k is available and its value on the jth feature is denoted by $x_{jk}(i+1)$. We have the new estimate of the mean parameter by updating the old one as follows:

$$m_{jk}(i+1) = \frac{i \cdot m_{jk}(i) + x_{jk}(i+1)}{i+1} = m_{jk}(i) + \frac{1}{i+1}\left[x_{jk}(i+1) - m_{jk}(i)\right] \tag{9}$$

Relying on the relation in Equation (9), we further have Equation (10) concerning the update of the standard deviation of features under a class.

$$\sigma_{jk}^2(i+1) = \frac{i\sigma_{jk}^2(i)}{i+1} + \frac{i\left(x_{jk}(i+1) - m_{jk}(i)\right)^2}{(i+1)^2} = \sigma_{jk}^2(i) + \frac{1}{i+1}\left[\frac{i}{i+1}\left(x_{jk}(i+1) - m_{jk}(i)\right)^2 - \sigma_{jk}^2(i)\right] \tag{10}$$

The standard deviation feature under class is

$$\begin{aligned}
\sum_{t=1}^{i+1}\left(x_{jk}(t) - m_{jk}(i+1)\right)^2 &= \sum_{t=1}^{i+1}\left[\left(x_{jk}(t) - m_{jk}(i)\right) - \frac{1}{i+1}\left(x_{jk}(i+1) - m_{jk}(i)\right)\right]^2 \\
&= \sum_{t=1}^{i+1}\begin{pmatrix} x_{jk}(t) \\ -m_{jk}(i) \end{pmatrix}^2 - \frac{2}{i+1}\begin{pmatrix} x_{jk}(i+1) \\ -m_{jk}(i) \end{pmatrix}\sum_{t=1}^{i+1}\begin{pmatrix} x_{jk}(t) \\ -m_{jk}(i) \end{pmatrix} + \frac{1}{i+1}\begin{pmatrix} x_{jk}(i+1) \\ -m_{jk}(i) \end{pmatrix}^2 \\
&= \sum_{t=1}^{i}\begin{pmatrix} x_{jk}(t) \\ -m_{jk}(i) \end{pmatrix}^2 + \begin{pmatrix} x_{jk}(i+1) \\ -m_{jk}(i) \end{pmatrix}^2 - \frac{1}{i+1}\begin{pmatrix} x_{jk}(i+1) \\ -m_{jk}(i) \end{pmatrix}^2 \\
&= \sum_{t=1}^{i}\begin{pmatrix} x_{jk}(t) \\ -m_{jk}(i) \end{pmatrix}^2 + \frac{i}{i+1}\begin{pmatrix} x_{jk}(i+1) \\ -m_{jk}(i) \end{pmatrix}^2 = i\sigma_{jk}^2(i) + \frac{i}{i+1}\begin{pmatrix} x_{jk}(i+1) \\ -m_{jk}(i) \end{pmatrix}^2
\end{aligned}$$

Therefore, we have

$$\sigma_{jk}^2(i+1) = \frac{1}{i+1}\sum_{t=1}^{i+1}\left(x_{jk}(t) - m_{jk}(i+1)\right)^2 = \frac{i}{i+1}\sigma_{jk}^2(i) + \frac{i}{(i+1)^2}\left(x_{jk}(i+1) - m_{jk}(i)\right)^2$$

Based on Equations (8)–(10) and the recursive calculation of the Bayesian classifier parameters, the online BC learning algorithm is presented as follows:

Algorithm 1: Online BC learning

Inputs: Number of classes K; number of features n
Data stream of labelled examples, $stream(\cdot)$
Output: Set of Bayesian classifier parameters:
$P(H_k), m_{jk}, \sigma_{jk}^2$, $k = 1 \cdots K$, $j = 1 \cdots n$

for k =1 to K **do**
  $P(H_k) = 0$
  $Count(k) = 0$
  for j = 1 to n **do**
    $m_{jk} = 0; \sigma_{jk}^2 = 0$
  end for
end for
$t = 0$
while $(strteam(t+1) \neq \varnothing)$ **do**
  $E = stream(t+1)$
  for k = 1 to K **do**
    $s = (H_k = class(E))$
    $P(H_k) \leftarrow P(H_k) + \frac{1}{t+1}[s - P(H_k)]$
  end for
  $k = index(class(E))$
  for j=1 to n **do**
    $$\sigma_{jk}^2 \leftarrow \sigma_{jk}^2 + \frac{1}{Count(k)+1}\left[\frac{Count(k)}{Count(k)+1}\left(x_j - m_{jk}\right)^2 - \sigma_{jk}^2\right]$$
    $$m_{jk} \leftarrow m_{jk} + \frac{1}{Count(k)+1}\left(x_j - m_{jk}\right)$$
    //x_j: the jth feature value of sample E
  end for
  $Count(k) \leftarrow Count(k) + 1$
  $t \leftarrow t + 1$
end while

The outcomes of Algorithm 1 are the prior probabilities $P(H_k)$ of the class, the mean m_{jk} and the standard deviation σ_{jk}^2 of class features. These parameters of the Bayesian classifier can be used in Equations (3) and (4) for online classification of new instances, and are continuously updated during the execution of the algorithm.

3.2. Adapting to Concept Drifts

The online BC learning algorithm presented in Subsection 3.1 is based on the recursive calculation of the class probabilities, and the mean and standard deviation of features under classes. For the sake of preserving the knowledge learned in the past, all historical samples are weighed equally when the total learning process is completed. Nonetheless, in a dynamic environment where data distributions evolve with time, the data that arrives recently is of more importance than the data that arrives long before. In order to adapt to concept drifts, a challenging issue is how to learn from more recent data. This subsection tackles this issue by proposing the improved online BC learning algorithm.

Revisiting Equations (8)–(10), it can be seen that the inverse of (i+1) is an important coefficient to decide the step size, where the old parameters of the Bayesian classifier are updated by a new example and i+1 is the number of examples that have been received so far. Obviously, the step size diminishes quickly with the increase of i, which results in the declining influence of new samples along with time. This is, unfortunately, contradictory to our expectation of learning with drifting data streams that, more recent samples should gain more importance in the learning procedure in order to not only dilute outdated knowledge, but also adapt to the concept drift. Under this consideration, we replace $\frac{1}{i+1}$ by a constant coefficient $\alpha\,(0 < \alpha < 1)$, which is termed as the learning rate in Equations (8)–(10). This leads to the new online incremental learning rules for the Bayesian classifier parameters given below:

$$P(H_k, i+1) = P(H_k, i) + \alpha[s - P(H_k, i)] \tag{11}$$

$$m_{jk}(i+1) = m_{jk}(i) + \alpha\left[x_{jk}(i+1) - m_{jk}(i)\right] \tag{12}$$

$$\sigma_{jk}^2(i+1)=\sigma_{jk}^2(i)+\alpha\left[(1-\alpha)\left(x_{jk}(i+1)-m_{jk}(i)\right)^2-\sigma_{jk}^2(i)\right] \tag{13}$$

The advantage of this revision is that, with the time-independent learning rate, the influence of training examples (that arrive later) will not decrease with time. We have the opportunity of defining a suitable value for α to achieve timely reactions (of the model) to the streaming drift. In principle, the value of α is related to the speed of the data evolution. The faster the data evolves, the larger α has to be. On the other hand, the learning rate must not be too large in order to avoid quickly forgetting previously acquired useful knowledge, i.e. the risk of catastrophic forgetting [23]. Based upon the new learning rules of Equations (12)–(14) , the improved online BC learning algorithm is given in Algorithm 2.

Algorithm 2: Improved online BC learning

Inputs: Number of classes K; number of features n
Data stream of labelled examples, $stream(\cdot)$Learning rate α

Output: Set of Bayesian classifier parameters:
$P(H_k), m_{jk}, \sigma_{jk}^2,\ k=1\cdots K,\ j=1\cdots n$

for k = 1 to K **do**
  $P(H_k)=0$
  for j = 1 to n **do**
    $m_{jk}=0; \sigma_{jk}^2=0$
  end for
end for
$t=0$
while $(strteam(t+1)\neq\varnothing)$ **do**
  $E=stream(t+1)$
  for k =1 to K **do**
    $s=(H_k=class(E))$
    $P(H_k)\leftarrow P(H_k)+\alpha[s-P(H_k)]$
  end for
  $k=index(class(E))$
  for j =1 to n **do**
    $\sigma_{jk}^2\leftarrow\sigma_{jk}^2+\alpha\left[(1-\alpha)\left(x_j-m_{jk}\right)^2-\sigma_{jk}^2\right]$
    $m_{jk}\leftarrow m_{jk}+\alpha\left(x_j-m_{jk}\right)$
    //x_j: the jth feature value of sample E
  end for
  $t=\leftarrow t+1$
end while

4. Results and Evaluation

This section presents the simulation tests conducted on some benchmark data sets, and evaluates the efficacy of the online learning algorithms. Subsection 4.1 briefly introduces the experimental setup. Subsequently, the results of simulation tests are given and discussed in Subsection 4.2.

4.1. Experimental Setup

Eight benchmark datasets from the UCI machine learning repository [24] are used for simulation tests in this paper. All these data sets are used for classification, see Table 1. Each data set is treated as a data stream in the simulation tests. This means that each training example is processed once and then discarded during the execution of the online learning algorithms.

Table 1 Summary of the eight benchmark data sets.

Dataset name	Attribute number	Class number	Sample number
Monk-2	6	2	432
Sonar	60	2	208
Wine	13	3	178
Titanic	3	2	220
Heart	13	2	270
Pirma	8	2	768
Balance	4	3	625
Australian	14	2	690

The K-fold cross-validation method [25] is used in the evaluation to make a fair assessment of the generalization ability of the learning algorithms. The concrete value of K is set to be 10 in the experiments. Hence, each data set is divided into 10 equal parts with one part being used as test data, and the other nine parts together being used as the training data (in each of the 10 trials). The final classification accuracy is the mean of the accuracy based on the test data of the 10 trials. Additionally, the learning rate α for the improved online algorithm is set to be 0.01 when learning with all the data sets.

4.2. Results and Discussions

We apply the traditional Bayesian classifier (offline BC learning) and the two online learning algorithms (online BC learning and improved online BC learning) to the eight benchmark data sets. The mean and variance of accuracy obtained by the three algorithms in the 10-fold cross-validation are shown in Table 2. It can be seen from the table that the online BC learning algorithm always has the same results as the offline BC learning algorithm on all the eight data sets. This is fully consistent with our theoretic recognition that, the online BC learning algorithm is equivalent to the offline learning counterpart which uses the entire data stream for once. In the comparison of the two online algorithms, we find that the improved online BC learning algorithm performs similarly as well as the online BC learning algorithm and, on the Australian data set, performs even slightly better. This is an interesting observation given that the used data sets are static containing no drifts inside. This indicates that, although proposed for nonstationary streams, the improved online BC learning algorithm can work well in stationary environments. The reason is that, if the data stream is time invariant, the new samples will still follow the current data distribution that has been learned so far. Therefore, increasing the influence of recent samples, as implemented in the improved online BC algorithm, will not lead to significant changes in the learning outcomes.

Table 2 Test accuracy in 10-fold cross validations.

Datasets	*Offline BC learning*		*Online BC learning*		*Improved online BC learning*	
	Mean	Variance	Mean	Variance	Mean	Variance
Monk-2	0.6668	0.0056	0.6668	0.0056	0.6529	0.0065
Sonar	0.6824	0.0175	0.6824	0.0175	0.6343	0.013
Wine	0.9722	0.0014	0.9722	0.0014	0.9611	0.0013
Titanic	0.7733	0.0006	0.7733	0.0006	0.7733	0.0006
Heart	0.8333	0.0061	0.8333	0.0061	0.8222	0.0063
Pima	0.7579	0.0013	0.7579	0.0013	0.7488	0.0015
Balance	0.9072	0.0006	0.9072	0.0006	0.8912	0.0011
Australian	0.7928	0.0027	0.7928	0.0027	0.7986	0.0026

It is also worthy to note that the Bayesian classifier parameters are always available when our online learning algorithm is applied to a data stream. This implies that the classifier can be applied to classify new instances while being updated in the online learning process. Definitely, the model performance in classification will evolve with time when more and more training examples arrive. When the online BC learning algorithm is executed, Figure 1 shows how the performance (accuracy on test data) of the Bayesian classifier changes with the proportion of processed training samples. We can see from the figure that the model accuracy generally has a trend to increase when more examples are incorporated into the model.

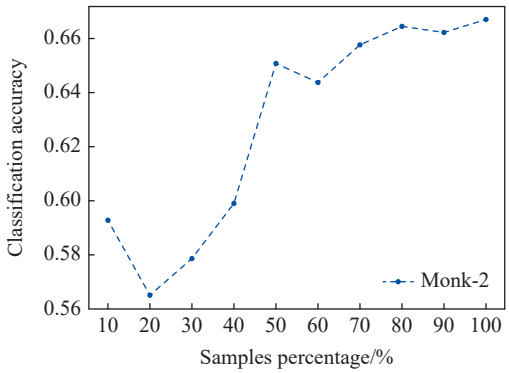

Figure 1. Evolution of test accuracy on Monk-2 data set with online BC learning

To evaluate the performance of the improved online learning algorithm in nonstationary environments, we create partially synthetic data by twisting data from the Pirma data set. More concretely, we reverse the labels of examples in the second half of the training streams to make a sharp concept drift. At the same time, we obtain the twisted test data by reversing the labels of examples of the original test data. The twisted test data is used to examine the accuracy of a model after the occurrence of the concept drift.

We apply both the online BC learning and improved online BC learning algorithms to the twisted training data streams, and also compare the resultant models on the test data. The original test data is used when learning is performed with the first half of the training streams, and the twisted test data is used when the learning has progressed with more than half of the training streams. The evolving performance of both models (from online BC learning and improved online BC learning algorithms) is shown in Figure 2 for comparison. We can see from the figure that both models have sharp degradation in test accuracy after the drift occurs in the data stream, and the model from the improved online BC learning manages to recover more quickly. This is attributed to the introduction of the learning rate to the improved online learning rules, and such rules put more effects on the recent samples in model update so that the model is able to adapt faster to the concept drift.

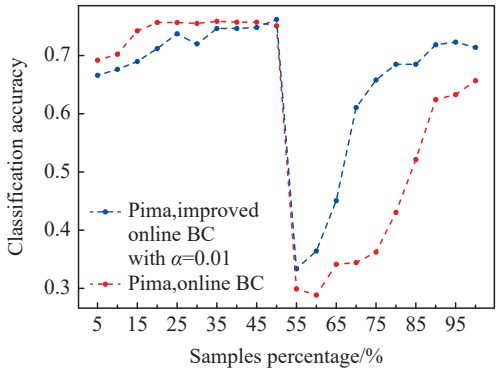

Figure 2. Comparing two online learning algorithms in concept drift

5. Conclusion

Data in continuous flow makes the traditional offline and batch learning modes infeasible. This paper has investigated new online learning methods to learn Bayesian classifiers with data streams in an online setting. The main contribution of our work has been highlighted in two-folds. First, we have presented the online learning algorithm for recursive Bayesian classifiers which have equivalent implementation procedures with the offline learning counterpart. This means that, after executing the online learning algorithm with full streams, we obtain the same model parameters as those of the offline learning counterpart. In an online incremental case, this is beneficial to offer the opportunity of solving big data learning problems to achieve higher computation and space efficiency. Second, we have proposed an improved online learning algorithm to tackle possible concept drifts in data streams. Interestingly, this improved algorithm not only contributes to faster model adaptation in nonstationary environments, but also works well in situations where no obvious drifts are observed from streams.

The proposed work has strong relevance to intelligent agents that interact with the environment of a "learning and prediction" cycle. Whenever a piece of new experience is obtained, our online learning algorithm can be applied to update the Bayesian classifier parameters by using the collected experience as the training example. Subsequently,

the updated classifier has been used to predict the class of the next new instance with augmented knowledge. As to the potential applications, we envision an autonomous vehicle that needs to learn to predict the intention of other traffic entities in the surroundings. When movement observations of traffic entities are continuously acquired with time, this paper has provided a new method to build and update the Bayesian classifier (as the prediction model) in a lifelong learning manner.

Author Contributions: **Peng Wu:** programming, writing—review and editing; **Ning Xiong:** conceptualization, methodology, writing—original draft preparation. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Swedish Research Council (ADAPTER, project number 2016-05431).

Data Availability Statement: The data used to support the findings of this study are included within the article.

Conflicts of Interest: The authors declare no conflict of interest.

Acknowledgments: This research was mainly supported by the Swedish Research Council (ADAPTER, project number 2016-05431), partly by the General Project of Key R&D Plan of Shanxi Province, high-technology field (No. 201903D121171), and the Project (No. 2022LJ041) especially for excellent doctor working in Shanxi.

References

1. Emani, C.K.; Cullot, N.; Nicolle, C. Understandable big data: A survey. *Comput. Sci. Rev.*, **2015**, *17*: 70−81.
2. Muthukrishnan, S. Data streams: Algorithms and applications. *Found. Trends Theor. Comput. Sci.*, **2005**, *1*: 117−236.
3. Gepperth, A.; Hammer, B. Incremental learning algorithms and applications. In *24th European Symposium on Artificial Neural Networks*, *Bruges*, *Belgium*, *27–29 April 2016*; ESANN, 2016.
4. Fisher, D.H. Knowledge acquisition via incremental conceptual clustering. *Mach. Learn.*, **1987**, *2*: 139−172.
5. Langley, P. Order effects in incremental learning. *Learning in Humans and Machines*: *Towards an Interdisciplinary Learning Science*. *Pergamon*, **1995**, *136*, 137.
6. Domingos, P.; Hulten, G. Mining high-speed data streams. In *Proceedings of the ACM Sixth International Conference on Knowledge Discovery and Data Mining*, *Boston*, *MA*, *USA*, *1 August 2000*; ACM: New York, 2000; pp. 71−80. doi:10.1145/347090.347107
7. Gama, J.; Medas, P. Learning decision trees from dynamic data streams. *J. Univ. Comput. Sci.*, **2005**, *11*: 1353−1366.
8. Rutkowski, L.; Jaworski, M.; Pietruczuk, L.; *et al*. The CART decision tree for mining data streams. *Inf. Sci.*, **2014**, *266*: 1−15.
9. Wang, S.; Minku, L.L.; Yao, X. Resampling-based ensemble methods for online class imbalance learning. *IEEE Trans. Knowl. Data Eng.*, **2015**, *27*: 1356−1368.
10. Yavtukhovskyi, V.; Abukhader, R.; Tillaeus, N.; et al. An incremental fuzzy learning approach for online classification of data streams. In *Proceedings of the 12th International Conference on Soft Computing and Pattern Recognition (SoCPaR 2020)*, *India*, *15–18 December 2020*; Springer: Cham, 2021; pp. 583–592. doi:10.1007/978-3-030-73689-7_56
11. Lemos, A.; Caminhas, W.; Gomide, F. Adaptive fault detection and diagnosis using an evolving fuzzy classifier. *Inf. Sci.*, **2013**, *220*: 64−85.
12. Lughofer, E.; Buchtala, O. Reliable all-pairs evolving fuzzy classifiers. *IEEE Trans. Fuzzy Syst.*, **2013**, *21*: 625−641.
13. Lughofer, E.; Weigl, E.; Heidl, W.; *et al*. Integrating new classes on the fly in evolving fuzzy classifier designs and their application in visual inspection. *Appl. Soft Comput.*, **2015**, *35*: 558−582.
14. Lughofer, E. Evolving multi-user fuzzy classifier systems integrating human uncertainty and expert knowledge. *Inf. Sci.*, **2022**, *596*: 30−52.
15. Lughofer, E. Evolving fuzzy and neuro-fuzzy systems: Fundamentals, stability, explainability, useability, and applications. In *Handbook on Computer Learning and Intelligence*; Angelov, P.P., Ed.; World Scientific: Singapore, 2022; pp. 133–234.
16. Gámez, J.C.; García, D.; González, A.; et al. On the use of an incremental approach to learn fuzzy classification rules for big data problems. In *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, *Vancouver*, *BC*, *Canada*, *24–29 July 2016*; IEEE: New York, 2016; pp. 1413–1420. doi:10.1109/FUZZ-IEEE.2016.7737855
17. Romero-Zaliz, R.; González, A.; Pérez, R. Incremental fuzzy learning algorithms in big data problems: A study on the size of learning subsets. In *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, *Naples*, *Italy*, *9–12 July 2017*; IEEE: New York, 2017; pp. 1–6. doi:10.1109/FUZZ-IEEE.2017.8015671
18. Duda, P.; Jaworski, M.; Cader, A.; *et al*. On training deep neural networks using a streaming approach. *J. Artif. Intell. Soft Comput. Res.*, **2020**, *10*: 15−26.
19. Hoi, S.C.H.; Sahoo, D.; Lu, J.; *et al*. Online learning: A comprehensive survey. *Neurocomputing*, **2021**, *459*: 249−289.
20. Read, J.; Bifet, A.; Pfahringer, B.; et al. Batch-incremental versus instance-incremental learning in dynamic and evolving data. In *11th International Symposium on Advances in Intelligent Data Analysis XI*, *Helsinki*, *Finland*, *25–27 October 2012*; Springer: Berlin, 2012; pp. 313–323. doi:10.1007/978-3-642-34156-4_29
21. Žliobaitė, I.; Pechenizkiy, M.; Gama, J. An overview of concept drift applications. In *Big Data Analysis*: *New Algorithms for A New Society*; Japkowicz, N.; Stefanowski, J., Eds.; Springer: Cham, 2016; pp. 91–114. doi:10.1007/978-3-319-26989-4_4
22. Zhang H. The optimality of naive Bayes. In *Proceedings of the 17th International FLAIRS Conference*, *American Association for Artificial Intelligence*, *Miami Beach*, *FL*, *USA*; AAAI Press: Palo Alto, 2004; pp. 562−567.
23. Ramasesh, V.V.; Lewkowycz, A.; Dyer, E. Effect of scale on catastrophic forgetting in neural networks. In *The 10th International Conference on Learning Representations*, *25–29 April 2022*; ICLR, 2022.
24. UC Irvine Machine Learning Repository. Available online: http://archive.ics.uci.edu/ml (accessed on 10 April 2022).

25. Refaeilzadeh, P.; Tang, L.; Liu, H. Cross-validation. In *Encyclopedia of Database Systems*; Liu, L.; Özsu, M.T., Eds.; Springer: New York, 2009; pp. 1–7. doi:10.1007/978-1-4899-7993-3_565-2

Citation: Wu, P.; Ning, X. Online learning of bayesian classifiers with nonstationary data streams. *International Journal of Network Dynamics and Intelligence*. 2023, 2(3), 100009. doi: 10.53941/ijndi.2023.100009