Classifying Potentially Unbounded Hierarchical Data Streams with Incremental Gaussian Naive Bayes[⋆]

Eduardo Tieppo[1,2]0000-0001-8271-0214, Jean Paul Barddal[1]0000-0001-9928-854X, and Júlio Cesar Nievola[1]0000-0002-2212-4499

[1] Pós-Graduação em Informática (PPGIa), Pontifícia Universidade Católica do Paraná (PUCPR), Curitiba, Brazil
{eduardo.tieppo,jean.barddal,nievola}@ppgia.pucpr.br
[2] Instituto Federal do Paraná (IFPR), Pinhais, Brazil
eduardo.tieppo@ifpr.edu.br

Abstract. Hierarchical data stream classification inherits the properties and constraints of hierarchical classification and data stream classification concomitantly. Therefore, it requires novel approaches that (i) can handle class hierarchies, (ii) can be updated over time, and (iii) are computationally light-weighted regarding processing time and memory usage. In this study, we propose the *Gaussian Naive Bayes for Hierarchical Data Streams* (GNB-hDS) method: an incremental Gaussian Naive Bayes for classifying potentially unbounded hierarchical data streams. GNB-hDS uses statistical summaries of the data stream instead of storing actual instances. These statistical summaries allow more efficient data storage, keep constant computational time and memory, and calculate the probability of an instance belonging to a specific class via the Bayes' Theorem. We compare our method against a technique that stores raw instances, and results show that our method obtains equivalent prediction rates while being significantly faster.

Keywords: Hierarchical Classification · Data Stream Classification · Gaussian Naive Bayes · Incremental Learning

1 Introduction

Hierarchical classification is required on problems where instances are labeled with classes that are related to one another in a hierarchy, such as in recognition of music genres and subgenres [12], computer-aided diagnosis where diseases are categorized by their etiology [43], recognition of animals, which are organized in a taxonomy [28,42], and, recently, even helping in COVID-19 identification using the hierarchical etiology of pneumonia [29].

However, classification techniques often assume that data samples of a particular problem are static and fully available to a learning model in a well-defined

[⋆] Supported by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

training step [26]. This assumption does not reflect many of the real-world scenarios in which classification is applied. The ever-increasing volume of data from diverse sources such as the Internet, wireless sensors, mobile devices, or social networks produces massive large-scale data streams [32,27,23].

Data streams are potentially unbounded over time and hence cannot be stored in memory. Also, as the time component is intrinsic in data streams, these are expected to be transient, i.e., the underlying data distribution is ever-changing, thus resulting in variations in the target concept, a phenomenon named concept drift [39,10,19,21].

When merged, hierarchical classification and data stream classification areas combine their properties and introduce new challenges in a roughly unexplored area: the hierarchical classification of data streams. Consequently, novel algorithms for hierarchical data stream classification must: (i) handle class hierarchies, (ii) be updatable over time, (iii) detect and adapt to changes in data behavior, and (iv) be computationally light-weighted regarding processing time and memory consumption [19,10,31].

In this study, we propose the *GNB-hDS* method: an Incremental Gaussian Naive Bayes for classifying potentially unbounded hierarchical data streams. *GNB-hDS* uses statistical summaries of the data stream instead of storing raw instances.

Despite the relevant application of Bayesian classifiers in hierarchical and data stream classification tasks separately, they have not been adapted yet to their intersection task. Therefore, to the best of our knowledge, this is the first method that combines incremental Bayesian learning with hierarchical classification. These statistical summaries allow a more efficient data storage, holding constant computational time and memory usage, and permit the calculation of the probability of a given instance belonging to a specific class via the Bayes' Theorem.

The novel contributions of this work are as follows:

- We qualify Gaussian Naive Bayes, a well-known classification technique [11], to work with potentially unbounded hierarchical data streams and in an incremental fashion by using updatable statistical summaries related to a class hierarchy.
- We propose *GNB-hDS*, a method for the hierarchical classification of data streams using summarization techniques. The model is incremental and handles potentially unbounded data streams with constant memory usage.

Furthermore, as a byproduct of this research, we make the source code for the proposed method, as well as the datasets used in the experiments, available for reproducibility.

The remainder of this paper is organized as follows. Section 2 describes the problem of hierarchical classification of data streams and Section 3 brings forward related works. Section 4 describes the proposed incremental Gaussian Naive Bayes for the hierarchical classification of data streams. Section 5 comprises the experimental protocol and the discussion of the results obtained. Finally, Section 6 concludes this paper and states envisioned future works.

2 Problem Statement

As mentioned above, in this paper, we are particularly interested in hierarchical data stream classification. This specific task combines characteristics and challenges from two different areas, and thus, it differs from classical classification in two key aspects.

First, concerning hierarchical classification, instances of a problem are assigned to a label path that belongs to a hierarchically structured set of classes instead of one single independent label [35]. Figure 1 compares a general approach of (a) flat (non-hierarchical) classification, and (b) hierarchical classification in an illustrative problem. In flat classification, the decision must be made while considering all the classes of the problem (all the possible song genres). Meanwhile, hierarchical classification concerns an existing class taxonomy, which can be used to make first smaller and generic decisions about the problem (in the example, decide first between Rock and R&B genres), and then more specific ones.

Second, concerning data stream classification, there is not the concept of a complete and fully available dataset; instead, instances of a problem are provided to the model sequentially over time [19]. Figure 2 compares (a) a traditional classification process and (b) a data stream classification process. In traditional (or batch) classification, the dataset is assumed to be static and completely available to the model at the training step. Next, the dataset is divided into training and test subsets; the training data is submitted to the learning model that reviews them as many times as necessary until obtaining a single satisfactory test model. This final model is then applied to the subset of test data and provides predictions and, consequently, accuracy estimates.

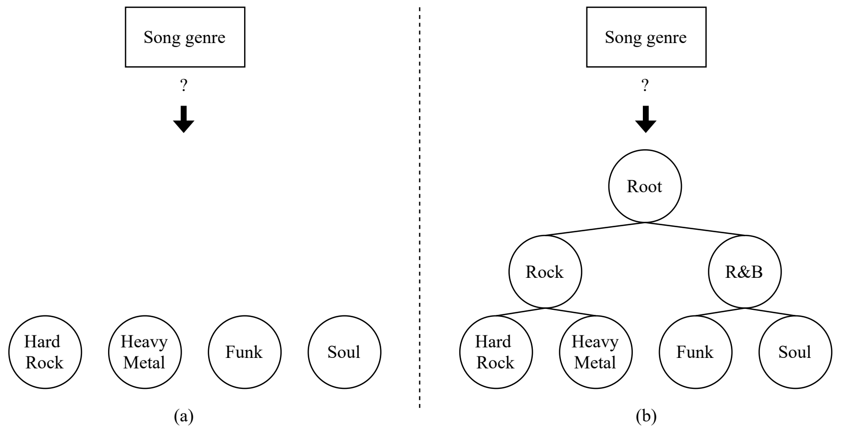

Fig. 1. Example of general approaches of (a) flat and (b) hierarchical classification in a hypothetical music genre problem. The class taxonomy can be used to lead smaller specific decisions about the classes by splitting the context complexity.

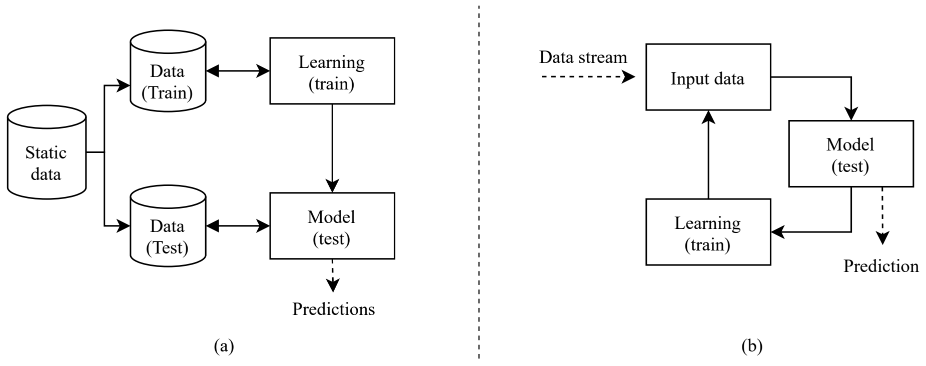

Fig. 2. Example of general approaches of (a) Traditional (batch) classification and (b) Data Stream Classification. An input data is obtained from the data stream, tested, incorporated into the model, and discarded; then, the cycle starts again.

In contrast, in streaming scenarios, data is made available sequentially over time, and even a single instance can be provided to the model at a time. Each instance is tested by the model, resulting in a prediction, and only after that it is incorporated into the model (being used as training data). This process, entitled 'test-then-train', is repeated for each instance, or chunk of instances, that is gathered from the stream. Any processed instance needs to be eventually discarded to maintain the model stable to process new instances since the data stream is potentially unbounded.

Thus, hierarchical classification of data streams regards learning models that use hierarchical data streams as input to their learning processes, not only as a source of data but effectively processing portions of the data over time, using the premise that there is no complete dataset and effectively using class taxonomy in their decision processes.

More formally, we let hDS define a hierarchical data stream in the $[(\vec{x}^t, \vec{y}^t)]_{t=0}^{\infty}$ format providing instances $(\vec{x}^t, \vec{y}^t)$ on a specific timestamp t, where $\vec{x}^t$ represents a d-dimensional features set and its values, and $\vec{y}^t$ represents the corresponding ground-truth label path (hierarchically structured classes).

These hierarchically structured classes compose a regular concept hierarchy arranged on a partially ordered set $(Y, \succ)$, where Y is a finite set containing all label paths and the relationship $\succ$ is defined as an asymmetric, anti-reflexive, and transitive subsumption (is-a) relation [35]. Finally, the classification of hierarchical data streams can be formally defined as $f^t : \vec{x}^t \mapsto \vec{y}^t$, where the function f^t is continuously updated by mapping features $\vec{x}$ to the corresponding label paths $\vec{y}^t$ accurately.

As the data streams are potentially infinite due to their time component, learning models are restrained by finite computational resources and must work with bounded memory and time, analyzing each instance only once according to their arrival and then discarding it. The processing time of an incoming instance from the data stream must not surpass the ratio in which new instances become available. Otherwise, the learning model will need to discard new instances without analyzing them [10,5].

3 Related Work

Machine learning models based on the Bayes' Theorem have been widely used in classification since their outputs are human-readable, they can naturally handle missing values, and are relatively easy to implement [36,22].

In hierarchical classification, Bayesian classifiers were used with different levels of adaptation. The authors in [14] used Bayesian probabilities attached to each node in the hierarchy using a Local Classifier per Node approach [35] and a top-down strategy to analyze the binary predictions along with the hierarchical structure. Similarly, the authors in [13] used binary classifiers for each class in the hierarchy considering both the parent and child classes of the current class.

In the works of [45,7], the authors also used Bayes-based classifiers within a Local Classifier per Node approach but to perform hierarchical multilabel classification. Finally, the authors in [36] proposed a Naive Bayes fitted to the hierarchical classification using a global approach [35].

A Bayesian classifier fitted to handle hierarchical classification needs to be adapted, at least, to consider the relationship between the hierarchically structured classes in the calculation of probabilities [36].

In data stream classification, incremental adaptations of Bayesian classifiers have been widely studied and are also widely applied in state-of-the-art algorithms. Data stream classification can be handled by a Naive Bayes classifier in a straightforward manner, since the learning model only needs to incrementally store summaries of data that allow the probabilities calculations as new instances are provided from the data stream [25].

The authors in [3] introduced the idea of recalculating probabilities for each instance provided to a model and this idea was later reinforced by the authors in [2,25]. In the work [33], the authors proposed an incremental Bayes Tree based on statistical summaries of data which are updated with each incoming instance. The authors in [9] used Naive Bayes classifiers ensembled with other tree-based classifiers to improve specific leaf node predictions. Finally, the authors in [4] also used incremental statistical summaries to restrain a Naive Bayes classifier and cope with limited computational resources.

It is noteworthy, nonetheless, to highlight the work of [28], where the authors proposed an incremental k-Nearest Neighbors (kNN) [1] approach for the hierarchical classification of data streams. This can be considered a seminal work of the area, yet, it does depict drawbacks such as kNN relies on distance computations, which are computationally intensive and can put in jeopardy time and memory usage constraints required by streaming scenarios [27,38]. In this sense, in Section 5, we compare our proposal (*GNB-hDS*) against the one proposed in [28] and show that *GNB-hDS* uses Bayes probabilities to obtain competitive prediction correctness with better computational performance.

4 Proposed Method

Our proposal, hereafter referred to as Gaussian Naive Bayes for Hierarchical Data Streams (*GNB-hDS*), is an incremental method for the hierarchical classification of data streams based on the Naive Bayes technique [18,11].

The main idea behind *GNB-hDS* is the use of incremental data summaries, specifically the mean, standard deviation, and the number of data instances, that allow the calculation of probabilities used in the Bayes' Theorem [11,22]. These incremental data summaries are attached to nodes of the hierarchy and are updated as new instances are gathered from the data stream. We implemented two key adaptations in the traditional Naive Bayes classifier to make it handle hierarchical data streams:

- Regarding the hierarchical data structure, the original algorithm was modified to consider not only one class but all related classes of a given instance. As the hierarchical data structure represents a subsumption relation, any new instance provided from the data stream also belongs to its ancestors. Thus, we traverse the hierarchy to update all data summaries of parent nodes recursively until the root node of the hierarchy.
- Regarding the streaming input data, the algorithm must store incremental statistical descriptors instead of the actual instances. Thus, we need to compute the mean, the standard deviation, and the count of data instances assigned to each class incrementally, discarding the instance after it is analyzed.

Regarding the stated problem approach, *GNB-hDS* represents the class taxonomy in a tree structure using local classifiers at each parent node and assigns leaf node classes as the last class of one predicted label path $\vec{y}^t$ (mandatory leaf-node and single path prediction) [35].

We point out that although the GNB-hDS method has been implemented here in a more specific way regarding the stated problem, GNB-hDS also supports direct acyclic graphs and non-mandatory leaf node prediction in its concept. To that, the data structure of a given node in the hierarchy should allow links with more than one parent node, and the top-down strategy used in the prediction step must consider some stopping criteria (e.g., a probability threshold) resulting in partial depth label paths.

Figure 3 illustrates the process performed by *GNB-hDS*. Circles represent classes, and dashed squares enclose classifiers. The method represents the class taxonomy in a tree structure, where R stands for the root node of the hierarchy and classes are related with each other (as described in Section 2).

When receiving an incoming instance for prediction, the method tackles the hierarchy using a Local Classifier per Parent Node (LCPN) approach [35], thus analyzing the current parent node and predicting between its child nodes by using probabilities obtained with the Bayes' Theorem. This process is repeated until a leaf node is reached.

Each node in the tree stores the count of instances (n), a d-dimensional incremental mean ($\bar{x}_n$) and a d-dimensional incremental standard deviation (σ_n)

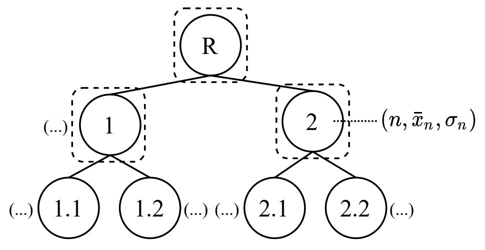

Fig. 3. Illustration of GNB-hDS method.

of the class represented (as shown in class 2). After the incoming instance processing, the statistical descriptors $(n, \bar{x}_n \sigma_n)$ are updated incrementally with the instance feature values on all the classes through the hierarchy regarding the ground-truth label path of that instance.

As before introduced, the instances are represented by data summaries comprising three statistical descriptors stored incrementally: (i) the count of class instances, (ii) the d-dimensional mean instance, and (iii) the d-dimensional standard deviation of the instances of a given class.

The number of instances assigned to a class C is stored in an attached counter. When an instance is retrieved from the stream, the C-th class counter is incremented alongside the counters of C's ancestors.

The incremental mean $(\bar{x}_n)$ and the incremental standard deviation (σ_n) considering each attribute from a $d-$dimensional x_n instance are obtained, respectively, from Equations 1 and 2, where n stands for the number of instances observed so far assigned to C [40,15].

Also, it is important to reinforce that the incremental mean and the incremental standard deviation are d-dimensional descriptors as the features set and its values from the d-dimensional x_n instance. Note that Equations 1 and 2 support only continuous feature sets and the current mean and the standard deviation of the previous observed instances assigned to C are represented by $\bar{x}_{n-1}$ and σ_{n-1}.

$$\bar{x}_n = \frac{(\bar{x}_{n-1}(n-1)) + x_n)}{n} \tag{1}$$

$$\sigma_n = \sqrt{\frac{(n-2)\,\sigma_{n-1}^2 + (n-1)\,(\bar{x}_{n-1} - \bar{x}_n)^2 + (x_n - \bar{x}_n)^2}{n-1}} \tag{2}$$

The prediction of the class to be assigned to an incoming instance provided from the data stream is performed in three steps: (i) computation of the *a priori* probabilities based on the count of class instances, (ii) computation of likelihood probabilities based on the Bayes' Theorem for each attribute of the incoming instance, and (iii) calculation of the maximum value of the *a posteriori* probability from the product of the independent feature probabilities given a class C.

The calculation of the likelihood probability is described in Equation 3, where i represents a feature index and j a class index [11].

$$p\left(x_i \middle| \, C_j\right) = \frac{1}{\sqrt{2\pi\sigma_{i,j}^2}} exp \left\{ -\frac{1}{2} \left(\frac{x_i - \bar{x}_{i,j}}{\sigma_{i,j}} \right)^2 \right\} \tag{3}$$

To perform the class assignment, the *GNB-hDS* obtains the class label with the maximum value of the *a posteriori* probability, as described in Equation 4, from the product of the independent feature probabilities given C [11].

$$p\left(C_j|x\right) \propto \left\{ \prod_i p\left(x_i \middle| \, C_j\right) \right\} p(C_j) \tag{4}$$

Moreover, these three steps are performed from the top of the hierarchy data structure and repeated until a leaf node is reached, resulting in the union of the class assignments made from Equation 4 and representing the final label path assigned to the incoming instance.

Algorithm 1 shows the proposed Gaussian Naive Bayes for Hierarchical Data Streams (*GNB-hDS*). It receives a hierarchical data stream hDS supplying instances $(\vec{x}, \vec{y})$ over time and, if required, outputs a set of predicted labels (a label path) $\widehat{\vec{y}_i}$ for each given instance $(\vec{x}, \vec{y})$, where $\vec{x}$ represents a d-dimensional features set and its values, and $\vec{y}$ represents the corresponding ground-truth label path of that instance.

The algorithm starts by understanding and representing the class taxonomy from the hierarchical data stream. The first loop (line 2 onwards) receives an

Algorithm 1:
GNB-hDS - Gaussian Naive Bayes for Hierarchical Data Streams

```
   input  : a hierarchical data stream $hDS$ providing instances $(\vec{x}, \vec{y})$
   output: a predicted label path $\widehat{\vec{y}_i}$ for the input instance
1  Tree ← classTaxonomy($hDS$);
2  foreach ($\vec{x} \in hDS$) do
3      predictedNode ← Tree.root;
4      while ¬(predictedNode.isLeaf) do
5          foreach (childNode ∈ predictedNode.children) do
6              priors ← priorProbability(childNode.Class);
7          end
8          likelihood ← likelihoodProbability($\vec{x}$,priors);
9          posterior ← posteriorProbability(likelihood,priors);
10         predictedNode ← argmax(posterior);
11         $\widehat{\vec{y}_i} \leftarrow \widehat{\vec{y}_i} \cup$ {predictedNode.label};
12     end
13     UpdateStatisticalDescriptors($\vec{y}_i$);
14 end
```

incoming instance from the hierarchical data stream. The following loop (lines 4 - 12) handles the hierarchy using the LCPN approach by predicting one of the children labels possible for that parent node.

The *a priori* probabilities are calculated in line 6 using the counts of class instances. The likelihood and posterior probabilities are calculated in lines 8 and 9 by the application of Equations 3 and 4, respectively. The predicted node for the evaluated parent is obtained in line 10, and the respective single label is appended to a partial label path $\widehat{\vec{y}_i}$ (line 11). This process is repeated until a leaf node is reached and the label path $\widehat{\vec{y}_i}$ is complete and ready to be output by the algorithm.

Finally, the algorithm updates the statistical descriptors (the count n of class instances, the incremental mean instance $\bar{x}_n$, and the incremental standard deviation σ_n) of all classes contained in $\vec{y}_i$, from the leaf to the root class.

5 Analysis

In this section, we report the experimental analysis conducted to compare our proposal against existing works in hierarchical data stream classification. First, we provide the experimental protocol adopted. Next, we discuss the results in terms of prediction and performance.

5.1 Experimental Protocol

Table 1 depicts the 14 hierarchically labeled datasets used in our testbed, listing their number of instances, features, and classes, the number of labels per level in the hierarchy (from top-level to leaf level), and references. These datasets contain different features, instances, and domains, thus assessing how our proposal behaves in different scenarios.

Table 1. Datasets used in the experiment.

Dataset	Instances	Features	Classes	Labels per level	Reference
Entomology	21,722	33	14	4, 6, 9, 14	[28]
Ichthyology	22,444	15	15	2, 6, 12, 15	[28]
Insects-a-b	52,848	33	6	1, 1, 2, 6	[37]
Insects-a-i	355,275	33	6	1, 1, 2, 6	[37]
Insects-i-a-r-b	79,986	33	6	1, 1, 2, 6	[37]
Insects-i-a-r-i	452,044	33	6	1, 1, 2, 6	[37]
Insects-i-b	57,018	33	6	1, 1, 2, 6	[37]
Insects-i-g-b	24,15	33	6	1, 1, 2, 6	[37]
Insects-i-g-i	143,323	33	6	1, 1, 2, 6	[37]
Insects-i-i	452,044	33	6	1, 1, 2, 6	[37]
Insects-i-r-b	79,986	33	6	1, 1, 2, 6	[37]
Insects-i-r-i	452,044	33	6	1, 1, 2, 6	[37]
Insects-o-o-c	905,145	33	24	4, 10, 14, 24	[37]
Instruments	9,419	30	31	5, 10, 31	[28]

During the experiments, classifiers were assessed in terms of hierarchical F-measure [24]. Like traditional classification metrics, the hierarchical F-Measure (hF) relies on hierarchical precision and recall components, but instances are associated with a path of labels, and the entire path is evaluated.

The hierarchical F-Measure is depicted in Equation 5, while its precision (hP) and recall (hR) components are described in Equations 6 and 7, respectively. In both precision and recall metrics, $\widehat{\vec{y_i}}$ is the set of labels predicted for the i-th instance, and $\vec{y_i}$ is its corresponding ground-truth label set.

$$hF = \frac{2 \times hP \times hR}{hP + hR} \tag{5}$$

$$hP = \frac{\sum_i \left|\widehat{\vec{y_i}} \bigcap \vec{y_i}\right|}{\sum_i \left|\widehat{\vec{y_i}}\right|} \tag{6}$$

$$hR = \frac{\sum_i \left|\widehat{\vec{y_i}} \bigcap \vec{y_i}\right|}{\sum_i \left|\vec{y_i}\right|} \tag{7}$$

We report the hF metric using the prequential test-then-train [10,20] validation method, where each instance is used to test the model before it is used for training and updating [10,21].

Furthermore, we measured the time performance by calculating the number of instances that a classifier can process per second.

We compared our proposed *GNB-hDS* to the hierarchical kNN described in Section 3 proposed in [28], hereafter referred to as *kNN-hDS*. We set up *kNN-hDS* with $k \in \{1, 3, 5\}$ and n (buffer size) $\in$ {5, 10, 15, 20}. The method *GNB-hDS* does not require setting parameters.

Finally, the results obtained by both methods were assessed using Wilcoxon hypothesis tests [41] with a 95% confidence level according to the protocol provided in [16] to verify significant differences in the hF and instances processed per second rates obtained by both methods.

The experiments in this paper were performed using Python 3.7. The proposed script containing the *GNB-hDS* method, as well as the datasets, are freely available for download[3].

5.2 Results

Table 2 shows the Hierarchical F-measure (hF) and Instances per second rates obtained by *kNN-hDS* and *GNB-hDS* in the datasets (greater values are highlighted in bold). In the *kNN-hDS method*, rates represent the best hF results obtained in the parameters configuration (as described in Section 5.1).

In terms of predictive performance assessment, the *GNB-hDS* method obtained better hF rates in 10 out of the 14 datasets. However, hF values are

[3] http://www.ppgia.pucpr.br/~jean.barddal/datasets/GNB-hDS.zip

Table 2. Hierarchical F-measure (*hF*) and Instances per second rates obtained during experiments.

	hF (%)		Instances per second	
Dataset	**kNN-hDS**	**GNB-hDS**	**kNN-hDS**	**GNB-hDS**
Entomology	**51.51**	48.64	127	**379**
Ichthyology	40.55	**46.82**	157	**395**
Insects-a-b	80.95	**81.11**	151	**489**
Insects-a-i	79.14	**80.88**	153	**494**
Insects-i-a-r-b	79.49	**81.42**	153	**495**
Insects-i-a-r-i	78.52	**81.57**	153	**491**
Insects-i-b	79.78	**80.55**	148	**500**
Insects-i-g-b	**83.29**	81.53	158	**483**
Insects-i-g-i	78.94	**80.40**	154	**495**
Insects-i-i	78.63	**80.90**	152	**497**
Insects-i-r-b	**80.14**	78.57	153	**491**
Insects-i-r-i	78.60	**81.61**	153	**494**
Insects-o-o-c	55.24	**64.14**	75	**282**
Instruments	**65.42**	48.31	79	**262**
Average	72.16	**72.60**	140.43	**446.21**

similar across both methods, such that the average difference between them is 0.44% while favoring *GNB-hDS*. Despite the improvements, the Wilcoxon test showed no statistical difference between *hF* rates obtained by the methods (p-$value = 0.2209$).

Concerning processing speed comparison, the *GNB-hDS* method was able to process more instances per second across all datasets, with an average rate of 446.21 instances against 140.43 of the *kNN-hDS* method. Thus, on average, our method was able to process 3.2 times more instances than the *kNN-hDS* method.

A one-tailed Wilcoxon test indicated a statistical difference between instances per second rates obtained by both methods (p-$value = 0.0005$) and confirmed that *GNB-hDS* is significantly faster when compared to *kNN-hDS* method.

Considering predictive performance and processing speed rates, *GNB-hDS* can obtain computational performance improvements without significant threats to the predictive performance by using statistical summaries of data combined with the class hierarchy information.

As aforementioned, the *GNB-hDS* method uses the premise of a Gaussian (normal) data distribution to deal with instance representation in the learning model [30]. In this sense, in addition to the previously described analysis, we investigated if *GNB-hDS* could use its premise to obtain better *hF* rates when data is normally distributed.

Thus, the *GNB-hDS* method, in addition to its speed, would present an additional advantage to the *kNN-hDS* method (or to any other method that does not use the premise of data normality) since it would be more adapted to classify normally distributed data. This advantage can be even more noticeable when we consider the data stream context, where data are potentially unbounded and statistical descriptors, such as mean and standard deviation, are more likely to obtain better representations of the population.

To examine this claim, we perform a Shapiro–Wilk test of normality [34] in all the datasets and applied the Yeo-Johnson power transformation technique [44] in all data to improve their adherence to a more normal distribution. The data normality was measured by Shapiro–Wilk test before and after the application of the Yeo-Johnson transformation.

Table 3 depicts the Shapiro-Wilk W statistic before (raw data) and after (transformed data) the Yeo-Johnson transformation. W statistic is bounded by 1, and closer values to this upper bound represent data more fitted to a normal distribution.

In addition, Table 3 depicts the Hierarchical F-measure (*hF*) obtained by both *GNB-hDS* and *kNN-hDS* methods when applied in both raw and transformed datasets. One can note that the predictive performance of *GNB-hDS* was improved when using transformed data in 13 out 14 datasets. Likewise, the average *hF* increased 2.35%, with noticeable increases in some datasets, such as Entomology (5.23%) and Insects-o-o-c (5.32%). Oppositely, *kNN-hDS* could not achieve the same improvements. In fact, *kNN-hDS* obtained lower *hF* rates with the transformed data resulting in a decrease of 1.6% in the average *hF* from 72.16% to 70.56%.

Finally, we performed one-tailed Wilcoxon tests to verify if the results obtained with the transformed datasets are significantly higher than with raw data for both *GNB-hDS* and *kNN-hDS* methods.

On *kNN-hDS*, the test indicated a statistical difference between performances with both data (*p-value* = 0.0009) favoring raw data, i.e., *kNN-hDS* does not benefit from a more normal distributed data. In contrast, on *GNB-hDS* the test indicated a statistical difference between performances with both data (*p*-

Table 3. Shapiro-Wilk W statistic of datasets and Hierarchical F-measure (hF) obtained by *GNB-hDS* with raw and transformed data.

	Shapiro-Wilk W statistic		**hF (%) obtained by GNB-hDS**		**hF (%) obtained by kNN-hDS**	
Dataset	**Raw Data**	**Transformed Data**	**Raw Data**	**Transformed Data**	**Raw Data**	**Transformed Data**
Entomology	0.7489	0.9517	48.64	**53.87**	**51.51**	51.07
Ichthyology	0.9028	0.9839	46.82	**50.27**	**40.55**	35.86
Insects-a-b	0.7236	0.9240	81.11	**81.90**	**80.95**	78.78
Insects-a-i	0.7248	0.9272	80.88	**84.05**	**79.14**	76.96
Insects-i-a-r-b	0.7268	0.9273	81.42	**83.48**	**79.49**	78.48
Insects-i-a-r-i	0.7234	0.9269	81.57	**83.40**	**78.52**	76.60
Insects-i-b	0.7239	0.9239	80.55	**82.28**	**79.78**	77.90
Insects-i-g-b	0.7280	0.9273	**81.53**	81.42	**83.29**	81.66
Insects-i-g-i	0.7227	0.9288	80.40	**83.16**	**78.94**	77.02
Insects-i-i	0.7234	0.9269	80.90	**83.05**	**78.63**	76.58
Insects-i-r-b	0.7250	0.9252	78.57	**79.58**	**80.14**	79.03
Insects-i-r-i	0.7234	0.9269	81.61	**83.45**	**78.60**	76.60
Insects-o-o-c	0.7416	0.9468	64.14	**69.46**	**55.24**	55.03
Instruments	0.9689	0.9868	48.31	**49.93**	65.42	**66.25**
Average	0.7577	0.9381	72.60	**74.95**	**72.16**	70.56

value = 0.0006) favoring the transformed data and has confirmed that *GNB-hDS* can take advantage of a more normal distribution-like data, thus corroborating our claims.

6 Conclusion

In this paper, we proposed *GNB-hDS*, an algorithm for hierarchical classification of data streams using data summaries to represent data. Our proposal is incremental and handles potentially unbounded data streams with constant memory consumption. Consequently, the proposed method processes more instances per second without dreadful impacts in prediction rates when compared to existing kNN-based techniques. To the best of our knowledge, our method extends the state-of-the-art being the first incremental method based on Bayes' Theorem tailored for hierarchical data streams classification.

The resulting source code and all the datasets used in the experiments are freely available for download to be used as a baseline to further research on the hierarchical classification of data streams, such as data preprocessing, computational resources analysis, and concept drift detection and adaptation.

In future works, we are interested in designing and applying other data summaries and different window types to maintain more than one *a priori* probabilities per class to allow *a posteriori* probabilities calculation weighted by data newness. Also, we are interested in applying existing drift detectors [8,17,6] to increase the responsiveness to changes in the data distribution.

References

1. Aha, D.W., Kibler, D., Albert, M.K.: Instance-based learning algorithms. Machine learning **6**(1), 37–66 (1991)
2. Alcobé, J.R.: Incremental learning of tree augmented naive bayes classifiers. In: Ibero-American Conference on Artificial Intelligence. pp. 32–41. Springer (2002)
3. Anderson, J.R., Matessa, M.: Explorations of an incremental, bayesian algorithm for categorization. Machine Learning **9**(4), 275–308 (1992)
4. Bahri, M., Maniu, S., Bifet, A.: A sketch-based naive bayes algorithms for evolving data streams. In: 2018 IEEE International Conference on Big Data (Big Data). pp. 604–613. IEEE (2018)
5. Barddal, J.P., Gomes, H.M., Enembreck, F., Pfahringer, B., Bifet, A.: On dynamic feature weighting for feature drifting data streams. In: Joint european conference on machine learning and knowledge discovery in databases. pp. 129–144. Springer (2016)
6. Barros, R.S., Cabral, D.R., Gonçalves Jr, P.M., Santos, S.G.: Rddm: Reactive drift detection method. Expert Systems with Applications **90**, 344–355 (2017)
7. Bi, W., Kwok, J.T.: Bayes-optimal hierarchical multilabel classification. IEEE Transactions on Knowledge and Data Engineering **27**(11), 2907–2918 (2015)
8. Bifet, A., Gavalda, R.: Learning from time-changing data with adaptive windowing. In: Proceedings of the 2007 SIAM international conference on data mining. pp. 443–448. SIAM (2007)

9. Bifet, A., Holmes, G., Pfahringer, B., Kirkby, R., Gavalda, R.: New ensemble methods for evolving data streams. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 139–148 (2009)
10. Bifet, A., Kirkby, R.: Data stream mining a practical approach (2009)
11. Bishop, C.M.: Pattern recognition and machine learning. springer (2006)
12. Burred, J.J., Lerch, A.: A hierarchical approach to automatic musical genre classification. In: Proceedings of the 6th international conference on digital audio effects. pp. 8–11. Citeseer (2003)
13. Campos Merschmann, L.H., Freitas, A.A.: An extended local hierarchical classifier for prediction of protein and gene functions. In: Proceedings of the 15th International Conference on Data Warehousing and Knowledge Discovery-Volume 8057. pp. 159–171 (2013)
14. Cesa-Bianchi, N., Gentile, C., Zaniboni, L.: Incremental algorithms for hierarchical classification. The Journal of Machine Learning Research **7**, 31–54 (2006)
15. Chan, T.F., Golub, G.H., LeVeque, R.J.: Algorithms for computing the sample variance: Analysis and recommendations. The American Statistician **37**(3), 242–247 (1983)
16. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research **7**(Jan), 1–30 (2006)
17. Frías-Blanco, I., del Campo-Ávila, J., Ramos-Jimenez, G., Morales-Bueno, R., Ortiz-Díaz, A., Caballero-Mota, Y.: Online and non-parametric drift detection methods based on hoeffding's bounds. IEEE Transactions on Knowledge and Data Engineering **27**(3), 810–823 (2014)
18. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian network classifiers. Machine learning **29**(2), 131–163 (1997)
19. Gama, J.: Knowledge discovery from data streams. Chapman and Hall/CRC (2010)
20. Gama, J., Sebastião, R., Rodrigues, P.P.: On evaluating stream learning algorithms. Machine learning **90**(3), 317–346 (2013)
21. Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. ACM computing surveys (CSUR) **46**(4), 44 (2014)
22. Han, J., Pei, J., Kamber, M.: Data mining: concepts and techniques. Elsevier (2011)
23. Hesabi, Z., Tari, Z., Goscinski, A., Fahad, A., Khalil, I., Queiroz, C.: Data summarization techniques for big data—a survey. In: Handbook on Data Centers, pp. 1109–1152. Springer (2015)
24. Kiritchenko, S., Famili, F.: Functional annotation of genes using hierarchical text categorization. Proceedings of BioLink SIG, ISMB (01 2005)
25. Klawonn, F., Angelov, P.: Evolving extended naive bayes classifiers. In: Sixth IEEE International Conference on Data Mining-Workshops (ICDMW'06). pp. 643–647. IEEE (2006)
26. Kotsiantis, S.B., Zaharakis, I., Pintelas, P.: Supervised machine learning: A review of classification techniques. Emerging artificial intelligence applications in computer engineering **160**, 3–24 (2007)
27. Nguyen, H.L., Woon, Y.K., Ng, W.K.: A survey on data stream clustering and classification. Knowledge and information systems **45**(3), 535–569 (2015)
28. Parmezan, A.R.S., Souza, V.M., Batista, G.E.: Towards hierarchical classification of data streams. In: Iberoamerican Congress on Pattern Recognition. pp. 314–322. Springer (2018)
29. Pereira, R.M., Bertolini, D., Teixeira, L.O., Silla Jr, C.N., Costa, Y.M.: Covid-19 identification in chest x-ray images on flat and hierarchical classification scenarios. Computer Methods and Programs in Biomedicine **194**, 105532 (2020)

30. Pontes, E.A.S.: A brief historical overview of the gaussian curve: From abraham de moivre to johann carl friedrich gauss. International Journal of Engineering Science Invention (IJESI) pp. 28–34 (2018)
31. Prasad, B.R., Agarwal, S.: Stream data mining: platforms, algorithms, performance evaluators and research trends. International journal of database theory and application **9**(9), 201–218 (2016)
32. Quiñonero-Candela, J., Sugiyama, M., Schwaighofer, A., Lawrence, N.D.: Dataset shift in machine learning. The MIT Press (2009)
33. Seidl, T., Assent, I., Kranen, P., Krieger, R., Herrmann, J.: 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. pp. 311–322 (2009)
34. Shapiro, S.S., Wilk, M.B.: An analysis of variance test for normality (complete samples). Biometrika **52**(3/4), 591–611 (1965)
35. Silla, C.N., Freitas, A.A.: A survey of hierarchical classification across different application domains. Data Mining and Knowledge Discovery **22**(1-2), 31–72 (2011)
36. Silla Jr, C.N., Freitas, A.A.: A global-model naive bayes approach to the hierarchical prediction of protein functions. In: 2009 Ninth IEEE International Conference on Data Mining. pp. 992–997. IEEE (2009)
37. Souza, V.M.A., Reis, D.M., Maletzke, A.G., Batista, G.E.A.P.A.: Challenges in benchmarking stream learning algorithms with real-world data. Data Mining and Knowledge Discovery pp. 1–54 (2020). https://doi.org/10.1007/s10618-020-00698-5
38. Steinbach, M., Ertöz, L., Kumar, V.: The challenges of clustering high dimensional data. In: New directions in statistical physics, pp. 273–309. Springer (2004)
39. Tsymbal, A.: The problem of concept drift: definitions and related work. Computer Science Department, Trinity College Dublin **106**(2), 58 (2004)
40. West, D.: Updating mean and variance estimates: An improved method. Communications of the ACM **22**(9), 532–535 (1979)
41. Wilcoxon, F.: Individual comparisons by ranking methods. In: Breakthroughs in statistics, pp. 196–202. Springer (1992)
42. Wu, F., Zhang, J., Honavar, V.: Learning classifiers using hierarchically structured class taxonomies. In: International Symposium on Abstraction, Reformulation, and Approximation. pp. 313–320. Springer (2005)
43. Yassin, N.I., Omran, S., El Houby, E.M., Allam, H.: Machine learning techniques for breast cancer computer aided diagnosis using different image modalities: A systematic review. Computer methods and programs in biomedicine **156**, 25–45 (2018)
44. Yeo, I.K., Johnson, R.A.: A new family of power transformations to improve normality or symmetry. Biometrika **87**(4), 954–959 (2000)
45. Zaragoza, J.C., Sucar, E., Morales, E., Bielza, C., Larranaga, P.: Bayesian chain classifiers for multidimensional classification. In: Twenty-second international joint conference on artificial intelligence. Citeseer (2011)