Evaluation of Summarization Schemes for Learning in Streams

Alec Pawling, Nitesh V. Chawla, and Amitabh Chaudhary

University of Notre Dame
{apawling,nchawla,achaudha}@cse.nd.edu

Abstract. Traditional discretization techniques for machine learning, from examples with continuous feature spaces, are not efficient when the data is in the form of a stream from an unknown, possibly changing, distribution. We present a time-and-memory-efficient discretization technique based on computing ε-approximate exponential frequency quantiles, and prove bounds on the worst-case error introduced in computing information entropy in data streams compared to an offline algorithm that has no efficiency constraints. We compare the empirical performance of the technique, using it for feature selection, with (streaming adaptations of) two popular methods of discretization, equal width binning and equal frequency binning, under a variety of streaming scenarios for real and artificial datasets. Our experiments show that ε-approximate exponential frequency quantiles are remarkably consistent in their performance, in contrast to the simple and efficient equal width binning that perform quite well when the streams are from stationary distributions, and quite poorly otherwise.

1 Introduction

Increasing number of real world applications now involve *data streams*; *e.g.*, applications in telecommunications, e-commerce, stock market tickers, fraud and intrusion detection, sensor networks, astronomy, biology, geography, and other sciences [1]. These data streams, whether commercial or scientific, spatial or temporal, almost always contain valuable knowledge, but are simply too fast and too voluminous for it to be discovered by traditional techniques. The objective of modern practitioners is to find time-and-memory-efficient ways of modeling and learning from the streaming data, possibly at the cost of some loss in accuracy. The generally accepted definition of time-and-space efficiency in streams is this: if n elements of the stream have been seen thus far, the algorithm should take time and space at most polylogarithmic in n [1].

There are special cases in which it is relatively easy to learn efficiently in data streams. One such scenario is when the stream is assumed to be a random sample drawn from a stationary or a slowly shifting distribution, often called the "i.i.d. assumption" (for independent and identically distributed). In such situations, a reasonably sized sample, typically not larger than $\mathsf{polylog}(n)$, of the data stream can be assumed to describe the overall distribution (and hence,

the entire stream) quite accurately, and the problem reduces to learning from such a sample. Another such case is when the underlying domain(s) of the data values is *discrete*, either nominal or consisting of a few numerical values (again, at most $\mathsf{polylog}(n)$). The trick in discrete domains is to simply count the number of instances of each value and use that as a representation of the data. Often this simple representation is sufficient and we get by with memory usage much less than the actual size of the stream.

Many real-world applications, however, involve *continuous* feature values from unknown, and possibly changing, distributions. In these, learning algorithms designed for nominal feature spaces are inapplicable. A simple workaround is to discretize the continuous feature space, and then use techniques similar to those for nominal feature spaces. Discretization of continuous feature spaces is, in fact, quite common in machine learning, primarily because many machine learning algorithms either require a discrete feature space or perform better with discretized features. Naive-Bayes, for example, often uses discretized features to improve performance. Accordingly many supervised and unsupervised discretization methods have been proposed; e.g., Dougherty et al. [2] identify defining characteristics and present an empirical evaluation of several methods. Examples of these methods are equal interval width binning, equal frequency width binning, monothetic contrast criterions (supervised and unsupervised), and k-means clustering. Most of these methods are unsuitable for discretizing data that comes in a stream as they, essentially, require the dataset to be *sorted*, and that cannot be done efficiently in a stream.

In this paper, we study discretization techniques that allow us to learn efficiently in data streams with continuous feature spaces even when generated by unknown, changing (and therefore non-i.i.d.) distributions. As motivating applications we use feature selection and Naive-Bayes classification. Our objective is to find a discretization technique(s) for efficiently performing these two learning tasks on streams, and evaluate the technique(s) both analytically and empirically in "worst-case" scenarios. To differentiate from traditional discretization techniques that are not suited for streams, we shall call those we are interested in *summarization techniques*, emphasizing that their job is to summarize the vast amount of information in a stream. For our applications, as we shall see later (in Section 2), a summarization technique needs to be able to answer the following question: For any feature X_i of the examples in the data stream, any value v_i in the domain of X_i, and any class label y associated with the examples, what is the number of examples seen thus far with a class label y and with values for the feature X_i at most v_i? Naturally no scheme that is time-and-memory-efficient in data streams can answer the question exactly. Thus the quality of a scheme depends on the amount of error in the answer, in particular, on the relationships between the nature of the error, the application, and the type of streaming scenario.

The main summarization technique we study is, what we call, the *ε-approximate exponential frequency quantiles* (denoted by exp in the figures) technique. It is based on maintaining quantiles for each feature-class pair. The ϕ quantile of n

elements, for $\phi \in (0, 1]$ is the $\lceil \phi n \rceil$th smallest element in the set (we assume each element is a number, as in our case). An ε-approximate ϕ quantile is an element that lies in a position between $\lceil \phi n(1-\varepsilon) \rceil$ and $\lceil \phi n(1+\varepsilon) \rceil$ in the sorted data set. In our technique we maintain quantiles for values of $\phi = \alpha^{-i}$ for some constant $\alpha > 1$ and $0 \leq i \leq \lg_\alpha n$; hence the term "exponential frequency" in the name. Gupta and Zane [3] developed efficient techniques for maintaining such quantiles and used it in the context of counting inversions in data streams. We extend the technique to perform feature selection. Towards an analytical evaluation of this technique, we show that its performance for feature selection is exceptionally good by proving worst-case bounds on the error introduced in the computation of information entropy (as required in feature selection) (Section 4). To the best of our knowledge these are the first known error bounds related to (continuous) feature selection in streams. Our empirical evaluation of the technique is quite extensive: first of all we compare its performance with two other popular summarization techniques: the *equal width* summary and the *ε-approximate equal frequency quantiles*. The former is just the equal width binning technique constrained to perform in a stream. The latter is an adaptation of the quantile technique to implement equal frequency binning. We compare these three summarization techniques, and an offline optimal technique that is not constrained by time or memory limitations. We examine their performance and memory usage on large-sized real and artificial data sets, under various streaming scenarios: a stationary distribution (i.i.d), various non-stationary (non-i.i.d.) distributions, and even a distribution that has missing feature values (Section 5).

Related Work

Here, constrained by space, we give only a brief survey of the related research. There has been some research in classification in data streams: either on learning a single classifier [4, 5] or a set of classifiers [6]. Hulten et al. [5, 7] build decision trees in data streams from nominal domains under the i.i.d. assumption on a stationary or a slowly shifting distribution. They make a clever use of the Hoeffding bounds to decide when a sufficiently large sample has been observed and use that to create their tree. If they observe that the distribution is shifting, they change to a different tree that represents the new data. Gehrke et al. developed BOAT [4], an incremental decision tree algorithm that handles continuous feature spaces, and scales to large datasets, often requiring just 2 scans of the data (we limit ourselves to one).

Gama and Pinto [8] present a method for incremental discretization on a data stream that is based on a histogram summary of the data. They use these histograms to further discretize the data using a variety of the standard discretization techniques (equal width, equal frequency, entropy-based *etc.*) and test the accuracy of Naive-Bayes classifiers built from these discretizations on relatively small datasets (≤ 58000 examples). Guha et al. [9] give a number of methods for computing the entropy of a data stream: a two-pass algorithm that computes a $(1+\varepsilon)$ approximation in $\tilde{O}(1/(\varepsilon^2 H))$ space, where H is the entropy, a single-pass $e/(e-1)+\varepsilon$ approximation algorithm that takes $\mathsf{polylog}(n)$ space,

as well as a single pass algorithm that achieves a multiplicative approximation even when H is very small, but uses significantly more space. It is not clear, however, if these algorithms can be adapted to compute information gain in a data stream.

2 Motivating Applications

We use feature selection and Naive-Bayes classification as our applications. Due to space constraints, we omit the discussion and results of Naive-Bayes; however, these are available in [10]. Both these methods have commonly been studied in the context of discretization techniques even in non-streaming scenarios, as they are both typically implemented using discretized data.

Feature selection is a common process in machine learning, in which a subset of the features available from the data are selected for application of a learning algorithm, such as for classification. Selecting features based on information gain is a popular method; in this a feature is selected based on the maximum decrease in information entropy possible from a partition based on that feature.

Specifically, let there be d features denoted by $\mathbf{X} = \{X_1, X_2, \ldots, X_d\}$ and c class labels denoted by the set $\mathbf{Y}$. Let S_n denote the stream after n examples. In particular, $S_n = (\mathbf{x}_{(1)}, y_{(1)}), (\mathbf{x}_{(2)}, y_{(2)}), \ldots, (\mathbf{x}_{(n)}, y_{(n)})$ is the stream of n ordered pairs, each consisting of a feature vector $\mathbf{x}_{(i)}$ and a class label $y_{(i)}$. Suppose we wanted to determine the information gain resulting from partitioning the examples in S_n at $X_i = v_i$. Let S_n^L be the subset of examples such that $X_i \leq v_i$ and S_n^R be the remaining examples in S_n. Additionally, let $S_{n,y}$ denote the examples in S_n with the class label y. The information entropy in S_n is defined as

$$I(S_n) = \sum_{y \in \mathbf{Y}} \left(-\frac{|S_{n,y}|}{|S_n|} \lg \frac{|S_{n,y}|}{|S_n|} \right). \tag{1}$$

The information gain by partitioning at $X_i = v_i$ is defined as the decrease in entropy due to the partitioning, i.e.,

$$\text{gain}(S_n, X_i, v_i) = I(S_n) - \left[\frac{|S_n^L|}{|S_n|} I(S_n^L) + \frac{|S_n^R|}{|S_n|} I(S_n^R) \right] \tag{2}$$

Observe that to compute information gain, we need to be able to compute $|S_{n,y}^L|$, i.e., the number of examples with class y such that $X_i \leq v_i$.

3 Summarization Methods

We now describe the three methods of data streams summarizations that we study. We do not explicitly describe the process, but it will be clear from the description how each method can be used to estimate the answer to the basic question our applications need answered: How many examples with class label y are there in the stream thus far with the X_i feature value at most v_i, for any class y, feature X_i and value v_i?

3.1 Equal Width Summary

Equal width binning is a simple, yet popular, discretization method that divides the the range of each feature into equally sized intervals. It can be accomplished in one pass only if the minimum and maximum values for each feature are known ahead of time. Equal width summary is an adaptation of equal width binning for stream that uses the first k examples of the stream to estimate the minimum and maximum values of each feature. This information is used to create a specified number of bins (which is $\mathsf{polylog}(n)$). The first k examples are placed in their respective bins, and all subsequent examples are used to update counts of items in the bins without storing the example.

3.2 Approximate Exponential Frequency Quantile Summary

The approximate exponential frequency quantile summary is based on the approximate quantile structure developed by Gupta and Zane [3]. In it, we maintain ε-approximate ϕ quantiles for $\phi = \alpha^0, \alpha^{-1}, \alpha^{-2}, \ldots$, in which $\alpha = (1+\varepsilon)$. The quantile values are denoted $Q(\alpha^0), Q(\alpha^{-1}), Q(\alpha^{-2}), \ldots$ respectively. They are maintained using a collection of random samplers. Here we give a very brief, intuitive description of these samplers; please see the paper by Gupta and Zane [3] for details. Suppose we want to maintain the value at rank $n\alpha^{-i}$ (the α^{-i} quantile). We sample each element in the stream with a probability $T\alpha^i/n$, where $T = O(\log n/\varepsilon^2)$ is a parameter, and keep the T smallest values. We *expect* the largest value in the sampler to have rank $n\alpha^{-i}$, as desired. To ensure we have the correct value *with high probability* we just to need to maintain samplers at finer intervals: we can get ε-approximate quantiles for $\phi = \alpha^{-i}$ with probability $1 - 1/n^2$ if we maintain random samplers corresponding to ranks at β^j, $0 \le j \le \log_\beta n$, in which $\beta = 1 + \varepsilon/10$ and $T = 8\,\varepsilon^{-2} \ln n$ [3].

We use the samplers above, albeit with the following modification: we maintain α^{-i} quantiles for only values such that $0 \le \alpha^{-i} \le 1/2$ — which we call the *left* quantiles; for the rest, we maintain $1 - \alpha^{-i}$ quantiles, for $0 \le \alpha^{-i} \le 1/2$ — which we call the *right* quantiles. Suppose we want to find the number of elements with value at most a given v, we first determine whether v "falls" in the left quantiles or in the right quantiles. Then we determine the largest quantile α^{-i} such that $Q(\alpha^{-i}) \le v$ (or the smallest quantile $1-\alpha^{-i}$ such that $Q(1-\alpha^{-i}) > v$) and use $n\alpha^{-i+0.5}$ (or $n(1-\alpha^{-i+0.5})$) as a $(1+\varepsilon)^{1.5}$-approximation of the required number of elements. On the whole we keep $O(\log_\beta n)$ samplers, so the overall space is $O(\log^2 n/\varepsilon^{-3})$. For learning in a stream, one set of left and right quantiles is maintained for each feature-class pair. This allows us to calculate $(1+\varepsilon)^{1.5}$-approximation of the number of elements of class y with values $X_i \le v_i$ for any class y, any feature X_i, and any value v_i with high probability.

3.3 Approximate Equal Frequency Quantile Summary

The approximate equal frequency quantile summary is like the approximate exponential frequency quantile summary, except instead of maintaining β^i quantiles for different values of i, we keep the quantiles equally spaced — with equal

number of examples between any consecutive pair. For a fair comparison with the other methods, the number of examples between two consecutive quantiles is chosen so that the same number of bins are created. These quantiles too are maintained independently for each feature-class pair.

4 Analytical Evaluation of Exponential Frequency Quantiles

In this section we present our theoretical evaluation of the performance of the exponential-frequency quantiles summarization when used to compute the information entropy at any point in a stream. In particular, suppose that we have seen n examples in the stream, and we are given a feature X_i and a value v_i, and are asked the following question: What would be the information entropy if the set of examples seen thus far were partitioned into two subsets based on the question "Is $X_i \leq v_i$?" We show that we can bound the error introduced if this information entropy was computed using the exponential-frequency quantile summary. As far as we know, this is the first such bound known for *computing entropy in a data stream*, i.e., by an algorithm that makes one pass over the data stream which contains examples that have continuous features and are generated by some evolving, possibly changing, distribution, and such that the algorithm takes at most polylogarithmic space in the number of examples seen thus far. The error bound has a relative (multiplicative) as well as an absolute (additive) component, and is described precisely in the following theorem.

Theorem 1. *Let the information entropy, at any given point in the stream, resulting from partitioning at $X_i \leq v_i$, for a given X_i and v_i be I^*. Let I be the corresponding information entropy computed using exponential-frequency approximate quantile summarization. Then $(1-9\varepsilon)I^* - 9\varepsilon \leq I \leq (1+9\varepsilon)I^* + 9\varepsilon$.*

Proof. Let the given point in the stream be after n examples and, as before, let S_n be the set of examples seen thus far. Let S_n^L be the subset of examples such that $X_i \leq v_i$ and S_n^R be the remaining examples in S_n. Additionally, let $S_{n,y}$ denote the examples in S_n with the class label y, for every $y \in Y$. Finally, let $S_{n,y}^L$ be the subset of $S_{n,y}$ such that $X_i \leq v_i$ and $S_{n,y}^R$ be the remaining examples in $S_{n,y}$. We saw earlier that the exponential-frequency approximate quantile summarization can be used to compute a $(1+\varepsilon)^{1.5} = 1 + 1.5\varepsilon + \Theta(\varepsilon^2)$ approximation of $|S_{n,y}^L|$ and $|S_{n,y}^R|$. We shall show how this ensures a bounded approximation for the information entropy. In the rest of the proof we ignore terms that are $O(\varepsilon^2)$ in the interest of a simpler presentation. The expression for the entropy I^* is

$$\frac{|S_n^L|}{|S_n|} I(S_n^L) + \frac{|S_n^R|}{|S_n|} I(S_n^R).$$

We first develop the upper bound; a lower bound will follow in a similar fashion. For the upper bound, consider $I(S_n^L)$. Observe that this is essentially a sum of

terms of the following form, one for each (non-empty) class $y \in Y$:

$$-\frac{|S^L_{n,y}|}{|S^L_n|} \lg \frac{|S^L_{n,y}|}{|S^L_n|}.$$

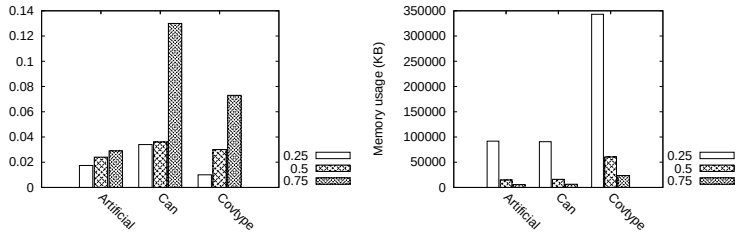
Now $|S^L_n| = \sum_y |S^L_{n,y}|$. Since we can compute an $(1 + 1.5\varepsilon)$-approximation of $|S^L_{n,y}|$, for each y, it follows that by adding each such approximation we can compute a $(1+1.5\varepsilon)$-approximation of the sum $|S^L_n|$. Thus our approximation of $|S^L_{n,y}|/|S^L_n|$ is upper bounded by $|S^L_{n,y}|(1 + 1.5\varepsilon)/(|S^L_n|(1 - 1.5\varepsilon)) = |S^L_{n,y}|/|S^L_n| \cdot (1 + 3\varepsilon)$, and lower bounded by $|S^L_{n,y}|/|S^L_n| \cdot (1 - 3\varepsilon)$. Let $|S^L_{n,y}|/|S^L_n|$ be denoted by p, and let the approximate value we obtain be denoted by $\tilde{p}$. Thus we have

$$-\tilde{p} \lg \tilde{p} \leq -p(1 + 3\varepsilon) \lg(p(1 - 3\varepsilon)).$$

(Recall that $-\lg \tilde{p}$ is a positive number since $\tilde{p}$ is always at most one, and that the right hand side has the factor $-\lg(p(1 - 3\varepsilon))$ instead of $-\lg(p(1 + 3\varepsilon))$ since it is the larger of the two.) We consider two cases: the first will result in both relative and absolute error terms, and the second in just a relative error term. In Case 1, assume $p > 1/2$. If we use the approximation $\ln(1 - x) = -x$ and ignore terms that are $O(\varepsilon^2)$, we get the following bound: $-\lg(p(1 - 3\varepsilon)) = -\lg p - \ln(1 - 3\varepsilon)/\ln 2 = -\lg p + 3\varepsilon/\ln 2 \leq -\lg p + 4.33\varepsilon$. Thus $-\tilde{p} \lg \tilde{p} \leq -(1 + 3\varepsilon)p \lg p + (1 + 3\varepsilon)p \cdot 4.33\varepsilon \leq -(1 + 3\varepsilon)p \lg p + 4.33\varepsilon$, since $p \leq 1$. For Case 2, assume $p \leq 1/2$. Thus $-\lg(p(1 - 3\varepsilon)) \leq -\lg p + 4.33\varepsilon$ as before, which in this case is bounded by $-\lg p(1 + 4.33\varepsilon)$ since $-\lg p \geq 1$. This results in $-\tilde{p} \lg \tilde{p} \leq -(1 + 3\varepsilon)p(1 + 4.33\varepsilon) \lg p = -(1 + 7.33\varepsilon)p \lg p$. Note, in $I(S^L_n)$, at most one such term can have $p > 1/2$, i.e., be in Case 1. Thus, by adding the approximations for each term, we have an approximation for $I(S^L_n)$ that is upper bounded by $(1+7.33\varepsilon)I(S^L_n)+4.33\varepsilon$. To bound the term $|S^L_n|/|S_n| \cdot I(S^L_n)$ in our expression for entropy, note that $|S_n| = n$ is known exactly, and we can obtain an $(1+1.5\varepsilon)$-approximation for $|S^L_n|$. We can similarly bound the approximation we get for $|S^R_n|/|S_n| \cdot I(S^R_n)$. Combining both, we get the following upper bound on the computed value of the entropy I: $(1+1.5\varepsilon)(1+7.33\varepsilon)I^* + 2 \cdot (1+1.5\varepsilon) \cdot 4.33\varepsilon \leq (1 + 8.53\varepsilon)I^* + 8.66\varepsilon$.

5 Empirical Evaluation

In this section we evaluate the accuracy and memory usage of the three summarizations empirically for feature selection and compare them with an offline algorithm that has no time or memory constraints. We present results only for experiments in which ε (as required by the approximate quantiles) is set to 0.5. This is because, based on experiments presented in [11] (e.g., see Figure 1), at this value we have, informally, a good compromise between low relative error and reasonable usage of memory. In experiments using equal width summaries we use $k = 100$ examples to establish the bin boundaries. In addition, for a fair comparison, we maintain as many bins as there are quantiles in the summaries for $\varepsilon = 0.5$.



(a) Relative Error (b) Memory Usage

Fig. 1: Computing information gain for different ε: 0.25, 0.50, and 0.75 [11].

Datasets

Large data sets are essential for our experiments, but we were limited to those available in the public domain. Table 1 summarizes the datasets we used. Among the large real data sets we found, we chose four to get a set with different characteristics. Since our method is for continuous features and nominal features will, in practice, be handled separately, we pruned the nominal features (if any) from each dataset. We also generated a large artificial dataset to stress-test the summarizations. Brief description of the data sets: (1) We generated the feature values of the *Artificial* dataset using normal distributions, and added random noise using different normal distributions. The class labels were determined by partitioning the space of the two "important" features into two using a random vector. (2) The *Can* dataset is from a scientific simulation of a can being crushed [12]. The data portion in which the can is being crushed (8,360 out of 443,872 examples) is labeled "very interesting" and the rest labeled "unknown." (3) The *Covtype* dataset is from the UCI repository [13] and is about forest cover types for different geographical features. It has 7 classes with a highly imbalanced distribution; we selected the 10 continuous features out of a total 54 features. (4) The *Census-Income* dataset is also from UCI and highly unbalanced. We selected the seven continuous features out of a total 45 features. (5) The *UCSD* dataset [14] is a large classification dataset used at the UCSD student data mining competition. We pruned its 42 features to 8 in our experiments.

Streaming Scenarios

We tested the summarization techniques on four different types of streaming scenarios: (1) The *Stationary distribution* which models an i.i.d scenario. For this the examples are randomly shuffled to create the stream. (2) The *Non-Stationary–Feature distribution*, in which the examples are sorted in increasing order of the most significant feature to create a "worst-case" non-i.i.d. distribution. (3) The *Non-Stationary–Class distribution*, in which the stream starts with a single class, followed by all classes occurring with the same frequency until all

Table 1: Datasets.

Dataset	Size	Classes	Features
Artificial	1,000,000	2	6
Can	443,872	2	9
Covtype	581,012	7	10
Census-Income	199,523	2	7
UCSD	1,837,583	2	8

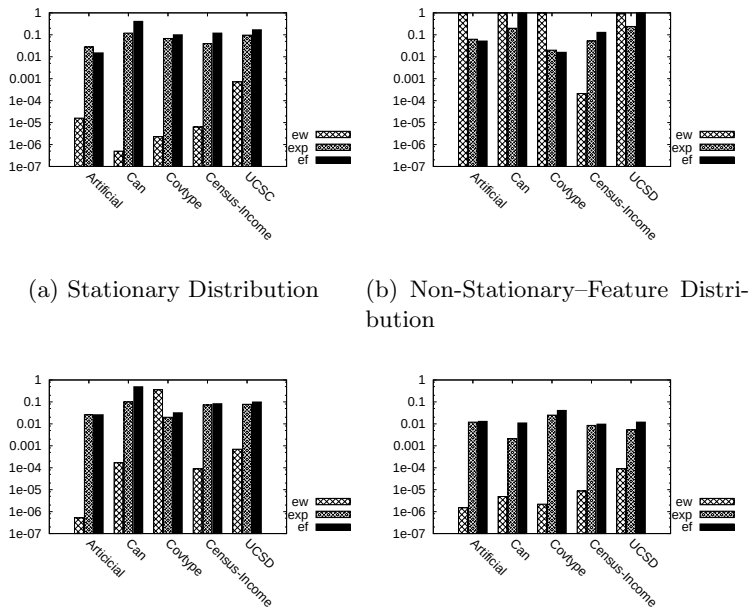
the examples of one (or more) class(s) are exhausted, to create another non-i.i.d. distribution. (4) The *Missing Values distribution*, which is like the stationary distribution, except 30% of the feature values have been removed at random. The non-stationary distributions may seem contrived, but our objective is to develop "bullet-proof" methods for unknown, rapidly evolving, streams.

5.1 Performance of Feature Selection

For each dataset and streaming scenario, we calculate the information gain at every tenth point of the total stream. Figure 2 shows the average relative error computing the gain for the best feature (according to gain), plotted for each combination of summarization, dataset, and streaming scenario. The approximate exponential frequency quantile technique shows remarkable consistency: its relative error is almost always between 0.01 and 0.1. When some features have missing values, it performs even better: relative error between 0.001 and 0.01. The likely reason for this is that we let the technique take memory based on the original size of the stream, without accounting for the smaller number of examples for each class-feature pair. In contrast, the equal width summary performs much better than the exponential frequency quantile summary for the stationary, non-stationary–class, and missing values distributions: with error better 10^{-7} and 0.01 most of the time. But for non-stationary–feature distribution, and the Covtype dataset on non-stationary–class distribution, its error shoots up to nearly 1. Thus the exponential frequency quantile summary is better suited for unknown streams. The equal frequency quantile summary rarely performs better than the exponential frequency quantile summary, and performs poorly on the non-stationary–feature distribution. Thus the strength in the exponential frequency quantile summary is in the exponential frequency part of the technique; as was also demonstrated by the analysis in Section 4.

Since we are considering information gain as a tool for feature selection, we examine how the feature rankings for each approximation correlates with the precise feature ranking. Figures 3 shows the *Spearman rank* correlations of the features sorted in descending order of importance, based on the information gain, for the Covtype dataset on the stationary, non-stationary–feature, and non-stationary–class distributions (we only show one dataset due to space constraints). Notice that the equal width summary produces a perfect ranking

for the stationary distribution, but degrades on the non-stationary–feature distribution. For the non-stationary–feature distribution, the feature on which the data stream was sorted (in this case, the most important feature) is likely to be misranked since the bins for that feature are established on a small subset of the range of the feature.



(a) Stationary Distribution

(b) Non-Stationary–Feature Distribution

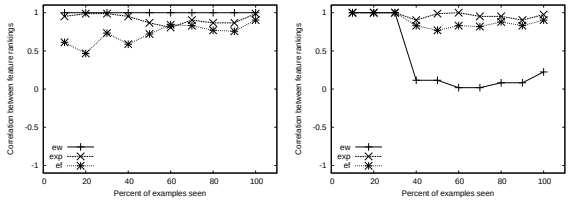
(c) Non-Stationary–Class Distribution

(d) Missing Values Distribution

Fig. 2: Average relative error between the information gain of the best feature for each dataset and streaming scenario; Y-axes are in log-scale.

5.2 Memory Usage

Figures 4 shows the memory usage for the offline and three approximations. In all cases, the equal width method needs the least amount of memory. The quantile based summarizations always require less memory than offline algorithm. The anomalous behavior of equal frequency quantiles can be explained. Note that the number of samplers that are actually useful depends upon the number of values that have been inserted into the quantile structure. If the structure isn't full (which it won't be unless all examples belong to a single class) there will be samplers that have been partially populated but aren't used. This is wasted



(a) Stationary Distribution(b) Non-Stationary–Feature Distribution

Fig. 3: Correlation between the feature ranking of offline and each summarization on Covtype for stationary and non-stationary distributions.

space. This is particularly problematic for the Covtype dataset since there are a large number of classes compared to the number of examples, and some classes have only a small number of examples. If we measure the memory usage where each example in the Covtype dataset is inserted into the quantile summary structures four times — essentially simulating a longer stream — the equal frequency quantile summary requires less memory than offline.

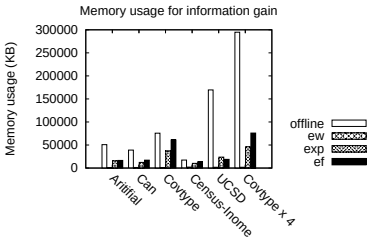


Fig. 4: Memory usage form information gain

6 Conclusion

In this paper we developed the ε-approximate exponential frequency quantiles based technique for summarizing data streams. Although this technique takes time and memory that are at most polylogarithmic in size of the stream, we showed that its performance in computing information entropy is very good by proving worst-case bounds on the error introduced in the computation compared to an offline optimal algorithm that has no time and memory constraints. In addition we empirically evaluated this technique and two other popular techniques — equal width summary and ε-approximate equal frequency quantiles — for feature selection under a variety of streaming scenarios for both real and artificial

datasets. The ε-approximate exponential frequency quantiles based technique is remarkably consistent, even in "worst-case" scenarios (its relative error is almost always around 0.1), and should be the technique of choice for streams with continuous feature spaces that are from unknown or non-stationary distributions (when the i.i.d. assumption does not hold). If, however, we know that the i.i.d. assumption is valid for a given stream, the technique to use (not surprisingly) is equal width summary, for its simplicity, performance, and memory usage.

References

1. Babcock, B., Babu, S., Datar, M., Motwani, R., Widom, J.: Models and issues in data stream systems. In: Proc. 21st ACM Symposium on Principles of Database Systems (PODS). (2002) 1–16
2. Dougherty, J., Kohavi, R., Sahami, M.: Supervised and unsupervised discretization of continuous features. In Prieditis, A., Russell, S., eds.: Proc. 12th International Conference on Machine Learning (ICML), Morgan Kaufmann (1995) 194–202
3. Gupta, A., Zane, F.X.: Counting inversions in lists. In: Proc. 14th ACM-SIAM Symposium on Discrete algorithms (SODA). (2003) 253–254
4. Gehrke, J., Ganti, V., Ramakrishnan, R., Loh, W.Y.: BOAT — optimistic decision tree construction. In: Proc. ACM SIGMOD International Conference on Management of Data. (1999) 169–180
5. Domingos, P., Hulten, G.: Mining high-speed data streams. In: Proc. 6th International Conference on Knowledge Discovery and Data Mining (KDD), ACM Press (2000) 71–80
6. Street, W.N., Kim, Y.: A streaming ensemble algorithm (sea) for large-scale classification. In: Proc. 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), New York, NY, USA, ACM Press (2001) 377–382
7. Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: Proc. 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), San Francisco, CA, ACM Press (2001) 97–106
8. Gama, J., Pinto, C.: Discretization from data streams: Applications to histograms and data mining. In: 2nd International Workshop on Knowledge Discovery from Data Streams. (2005)
9. Guha, S., McGregor, A., Venkatasubramanian, S.: Streaming and sublinear approximation of entropy and information distances. In: Proc. ACM SIAM Symposium on Discrete Algorithms (SODA). (2006) 733–742
10. Pawling, A., Chawla, N.V., Chaudhary, A.: Evaluation of summarization schemes for learning in streams. Technical Report 2006-08, University of Notre Dame (2006) `www.cse.nd.edu/research/tech_reports`.
11. Pawling, A., Chawla, N.V., Chaudhary, A.: Computing information gain in data streams. In: ICDM Workshop on Temporal Data Mining: Algorithms, Theory, and Applications. (2005)
12. Chawla, N.V., Hall, L.O.: Modifying MUSTAFA to capture salient data. Technical report, University of South Florida, Computer Science and Engineering Department (1999)
13. Newman, D.J., Hettich, S., Blake, C.L., Merz, C.J.: UCI repository of machine learning databases. http://www.ics.uci.edu/˜mlearn/MLRepository.html (1998)
14. UCSD: UCSD student data mining competition. `http://mill.ucsd.edu/ (2006)`