STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **x**, Month 202x, pp 0–13.
Published online in International Academic Press (www.IAPress.org)

An Evaluation of Discretization Techniques for HMM-Based Classifiers

Boutaina Ouriarhli [1], Badreddine Benyacoub [2], Hafida Benazza [1]

[1] *Faculty of Sciences, Mohammed V University, Rabat*
[2] *National Institute of Statistics and Applied Economics, Rabat*

Abstract Discretization of continuous features is an important task to handle problems with real values in machine learning. Many supervised classification algorithms perform well using a discrete space and the discretization process of the continuous features is suitable for more traditional algorithms the process. In this paper, we present the classification model based on the Hidden Markov Model (HMM) developed recently by Benyacoub and al using several discretization methods existing in the literature to construct the classifier. We conduct an experimental study using 9 benchmarking data sets to evaluate the performance and examine the effect of discretization methods on the assessment of the proposed learning algorithm. We conduct an experimental study using 9 benchmarking data sets to evaluate the performance and examine the effect of discretization methods on the assessment of the proposed learning algorithm. We report Accuracy (ACC) and Area Under the Curve (AUC), and we validate the global and pairwise differences across methods using the Friedman test followed by the Nemenyi post-hoc procedure.

Keywords Discretization, Hidden Markov Model, Classification, Continuous Attributes, Data Mining.

DOI: 10.19139/soic-2310-5070-2764

1. Introduction

ata pre-processing is an essential phase in data mining that aims to ensure the quality of the data and consists of several steps such as data transformation, data reduction, and other strategies. Its main objective is to clean the collected data and made its suitable during mining steps, which can improve the overall quality of the mining results [1]. One of the basic data transformation methods is the discretization process, and has attracted increasing research interest lately.

The main aim of discretization process is to convert a set of continuous attributes into discrete ones by assigning categorical values to intervals. This transformation allows to present numerical data into categorical data. Morover, the discretization process enables these algorithms to effectively handle continuous attribute, and thereby enhancing their performance in various applications [2] and [3]. While many discretization methods have been employed by many supervised learning models, including HMM models and applied in many fields.

HMMs have been successfully and extensively used in a diverse range of applications, citing as an example : economics and finance [5, 6], gesture recognition [7]and [8],speech recognition [9], computational biology [10],etc. In practice, a discrete space is required for many supervised classification tasks using the HMM framework. The data sets that are commonly used for classification tasks often have numeric attributes, This requires preprocessing using discretization methods to make them compatible with HMM classifiers during the training process [11].
Recent studies have advanced supervised and semi-supervised discretization. For instance, a semi-supervised adaptive discriminative discretization (SADD) improves class-separability under label scarcity [12]. Comprehensive supervised discretization evaluations update best practices and benchmarks [13]. In applied healthcare settings, combining discretization with missing-value imputation improves downstream classification

ISSN 2310-5070 (online) ISSN 2311-004X (print)
Copyright © 202x International Academic Press

robustness [14]. Moreover, the interaction between discretization and class-imbalance resampling has been quantified, informing pipeline design for imbalanced, multiclass data [15]. These works situate our comparative HMM study within modern discretization research that blends statistical criteria with learning-based optimization.

This paper explores the use of different discretization methods in a supervised classificarion model based on a Hidden Markov model(HMM). HMMs are powerful statistical tools for modeling generative processes that are described by an underlying process that generates an observed process [4].

In this comparative study, we analyze severl discretization techniques specifically designed for Hidden Markov Models (HMMs). The main objective is to evaluate the impact of different discretization methods on the performance of an HMM-based classifier when applied to real data sets. This paper introduces the discrete version of the Hidden Markov Model (HMM), focusing on its probabilistic structure for classification. We then discuss the preprocessing steps, focusing on the discretization of continuous features to match the expectations of the model.

Among the main contributions of this work, we perform an empirical evaluation of different discretization techniques for the HMM classifier, using real-world training datasets. We also investigate how the number of features and observations affects the classifier performance, focusing on key indicators such as accuracy and area under the ROC curve (AUC).

1.1. Related work

It is worth noting that a well-designed discretization algorithm can provide a succinct summary of continuous attributes, which can aid in comprehending the data. Furthermore, it can enhance the accuracy and efficiency of learning [16]. In the literature, the existing discretization methods can be categorized into five distinct groups based on specific criteria [17], [2, 18].

- Unsupervised Versus Supervised: Unsupervised algorithms (or class-blind) discretize characteristics without considering the associated dependent attribute [19]. In contrast, supervised algorithms consider the dependent attribute and discretize the attributes in a manner that ensures most cases during an interval are classified into the same category. The goal is to optimize the correspondence between attribute values and the desired result, using relevant data provided by the dependent attribute [20, 21].
- Local Versus Global: Local methods create divisions that are executed within specific areas of the instance space. This technique partitions the space based on the distinctive attributes of the data in each localized region of a subset of instances[18].While global method applies discretization across all the entire instance space [22, 2].
- Dynamic Versus Static: Dynamic method discretizers work during the learning phase, while in static discretizers finish their work before the learning phase starts and it operates independently of the learning algorithm [2, 23]. Many discretizers currently in practice are static, mainly because dynamic discretizers are often used as components or stages inside Data Mining (DM) techniques, especially for processing numerical data [24].
- Splitting Versus Merging: This pertains to the method employed to yield new intervals. Splitting methods are used to determine a specific place where a cut may be made among all the possible boundary points. This allows the domain to be divided into two separate parts. On the other hand, merging procedures start with a predetermined split and remove a selected point of separation, thereby merging the two adjacent intervals [24] and [25].
- Univariate vs. Multivariate: Univariate algorithms discretize each characteristic independently, whereas multivariate algorithms consider a combination among all attributes (i.e simultaneously considers multiple features) [23].

The research mentioned in this section of this article is essential to clarify that the studies reviewed are foundational but do not encompass the entire scope of this research. The main objective of this paper is to examine and compare several discretization strategies across different categories, and to assess their impact on

the effectiveness of HMM classifiers. This comprehensive analysis aims to demonstrate the capabilities of HMM classifiers when applied to discretized data and to illustrate their potential in handling real-value problems in machine learning [24].

2. Methodology

This section reveals our methodology using Hidden Markov Models (HMM) as a framework. We explore how HMMs can serve as classifiers to predict posterior probabilities, starting with an overview of supervised learning issues and the associated predictive probabilistic models. In addition, we examine the discretization strategy for the HMM classifier.

2.1. Hidden Markov Model

An HMM (Hidden Markov Model) is a bivariate process $\{(\mathbf{X}_t, \mathbf{Y}_t)\}_{t\geq 0}$ operates as a discrete-time Markov Chain with unobservable states, defined on a given probability space $(\Omega, \mathcal{F}, \mathbb{P})$ [30]. It is employed to model generative sequences characterized by an underlying stochastic process that generates an observable sequence [26, 27]. The Hidden Markov Model (HMM) belongs to the category of statistical models that incorporate various stochastic techniques into two primary processes. The starting process is dispicted by a Markov chain, which delineates a series of hidden states. whereas the second process is defined by a set of random variables which determine the observation sequence. HMMs are often differentiated from observable Markov models by their use of an indirect representation of states. In practice, Hidden Markov Models(HMMs)provide outputs that are weighted according to probabilities [28].The schematic representation of Hidden Markov Models with external inputs is illustrated in Fig. 1. [29].

The structure of a discrete HMM are:

$\{X_t\}_{t=1}^{n}$: A sequence of hidden state variables with a discrete and finite state space $S = \{e_1, e_2, \ldots, e_N\}$ [30].

$\{Y_t\}_{t=1}^{n}$: A set of discrete observations emitted from each hidden state within the finite set $S_Y = \{f_1, f_2, \ldots, f_m\}$ [31].

$\{\pi_i\}_{i=1,2,\ldots,N}$: the prior probability distribution; that is, $P(X_i = e_i) = \pi_i$ is the probability that the system starts in state X_i [32].

$A = \{a_{ij}\}_{i,j=1}^{N}$: the transition probabilities matrix, it is given by $a_{ij} = P(X_{t+1} = e_j \mid X_t = e_i), \quad \forall t \quad 1 \leq i, j \leq N$ defines the probability of transitioning from state e_i to state e_j. Here, X_t indicates the state at time t [31].

$C = \{c_{ij}\}_{i,j=1}^{N}$: the emission probabilities matrix, where $c_{ji} = P(Y_t = f_j \mid X_t = e_i), 1 \leq i \leq N; 1 \leq j \leq m.$ specifies the probability of observing $f_j$given the system is in state e_i at time t. Here, Y_t represents the observation at time t [31].

For a probability distribution to be considered valid within the context of Hidden Markov Models ,the prior probabilities, transition probabilities, and emission probabilities must satisfy certain conditions. These are formally defined as follows: the sum of the transition probabilities from any state must sum to one, $\sum_{i=1}^{N} a_{ij} = 1$; similarly, the sum of the emission probabilities for any state must also sum to one, $\sum_{j=1}^{M} c_{ij} = 1$; and the sum of the prior probabilities must total one, $\sum_{i=1}^{N} \pi_i = 1$. Additionally, all these probabilities must be non-negative: $a_{ij} \geq 0$, $c_{ij} \geq 0$, and $\pi_i \geq 0$. An HMM is typically represented succinctly by the triplet notation $\lambda = (A, C, \pi)$ [31].
In this study, the number of hidden states N was defined based on the number of classes in each dataset: $N = 2$ for binary problems and $N > 2$ for multiclass classification. To analyze the robustness of the model, additional tests were conducted with $N \in \{2, 3, 5, 7, 10\}$, and the results confirmed stable performance for $N \geq 5$. The initial

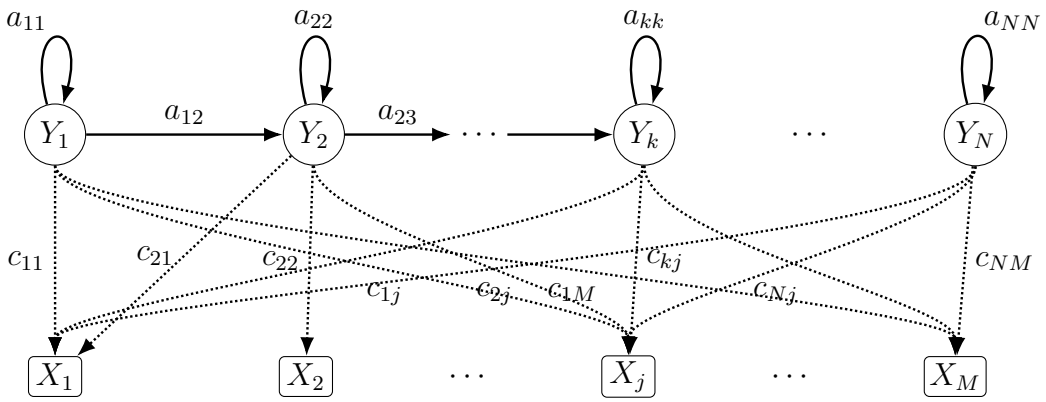

Figure 1

probability vector π was assumed to be uniform, reflecting the absence of any prior bias toward a particular class at the start of training. The transition matrix A was fixed to the identity matrix, as temporal dependencies were not exploited in this classification framework, and the emission matrix C was estimated from conditional frequencies $P(Y_k = f_{jk} \mid X = e_i)$ is obtained from the training data and normalized to satisfy probability constraints. This configuration provides a clear and reproducible definition of the HMM parameters and extends our previous work on the multiclass HMM classifier with left inverse algorithm [33], where the left-inverse estimation approach was first introduced and theoretically validated for a probabilistic classification.

2.2. Hidden Markov Models for Supervised Classification

In the context of supervised classification problems, we define a scenario where each instance is characterized by a vector of attribute values and assigned to one of a predefined set of classes. The objective of classification is to construct a classifier based on training data with known classes, which can then be used to predict the class of new instances [11].

Benyacoub et al. [35] propose novel model of supervised classification leveraging the capabilities of HMMs as probabilistic classifiers. The observed process Y is defined at discrete time instances, and the relationship between Y and X is expressed by a matrix C, representing the HMM parameters estimated from the data. This setup facilitates the computation of conditional probabilities of each class given an observation, thus enabling the classification of new instances [11].

The observation conditional probability can be decomposed into deterministic and stochastic components in the dynamics equations:

$$Y_t = CX_t + W_t$$

The coefficients of matrix C represent the parameters of the HMM and are estimated from the data.

To determine a set of linear functions corresponding to each group and to identify the expression for the estimated probability, the above equation has been used to develop an estimation algorithm for posterior probabilities $P(X = e_i \mid \mathbf{Y})$ for $i = 1, 2, \ldots, N$. The expression for the probability that X will take on the i-th possible value is presented as: $P(X = e_i \mid \mathbf{Y} = \mathbf{y}) = \sum_{k=1}^{l} \sum_{j=1}^{m_k} u_{ij}^k Y_k, \quad i = 1, \ldots, N$, where u_{ij}^k represents the coefficients of the classification model.
The obtained linear discriminant functions are then used to classify new observations.

Given a probabilistic output, we can compute our "best guess" for the true label using the formula:

$$\hat{X} = \hat{f}(\mathbf{Y}) = \arg \max_{e_1,\ldots,e_N} p(X = e_i \mid \mathbf{Y})$$

This corresponds to the most probable class label, known as the mode of the distribution $p(X \mid \mathbf{Y})$. This estimate is also referred to as the maximum a posteriori (MAP) estimate [36].

2.3. Discretization for HMM Classifiers

The objective of classification is to develop a classifier that acts as a decision rule to categorize new data into predefined groups. The learning process involves several key steps, starting with data collection and analysis. Typically, the raw datasets collected are not immediately suitable for model training, necessitating significant pre-processing efforts.

A crucial part of this pre-processing is the discretization of continuous attributes. Discretization transforms continuous data into categorical data, which is essential for methods such as Hidden Markov Models (HMMs) that require discrete input. Several algorithms have been proposed to address this task, including the Boolean reasoning algorithm, the entropy-based algorithm, and the naive algorithm.

The discretization process involves dividing the range of each continuous attribute into discrete intervals, effectively converting continuous features into categorical ones. This step is critical to ensure that the input data is in a suitable format for training the HMM.

To efficiently derive linear discriminant functions for classification, the input data is structured as a matrix, where each column represents an observed characteristic. This organization facilitates the implementation of our algorithm, ensuring that the model can be trained effectively. Thus, the entire process of constructing the HMM model can be outlined as follows: [35].

In our framework, each discretization interval defines a categorical symbol of the observation variable, Y. This design choice directly determines the structure of the emission probability matrix C, where each entry is given by $c_{ji} = P(Y^k = f_{jk} \mid X = e_i)$. A coarse discretization merges a wide range of values into a single category, which reduces the discriminative power of the corresponding columns in C. However, an excessively fine discretization produces a large and sparse matrix, which may lead to numerical instability when estimating posterior probabilities. The observation model $Y = CX + W$, where $U = (C^T C)^{-1} C^T$ provides an estimator of the class vector X as $\hat{X} = UY$. Consequently, the quality of the posterior probabilities $P(X \mid Y)$ is directly tied to the conditioning of C, which is a function of the selected discretization scheme. This theoretical clarification explains how discretization bounds influence HMM parameters and why selecting appropriate intervals can substantially improve the model performance.

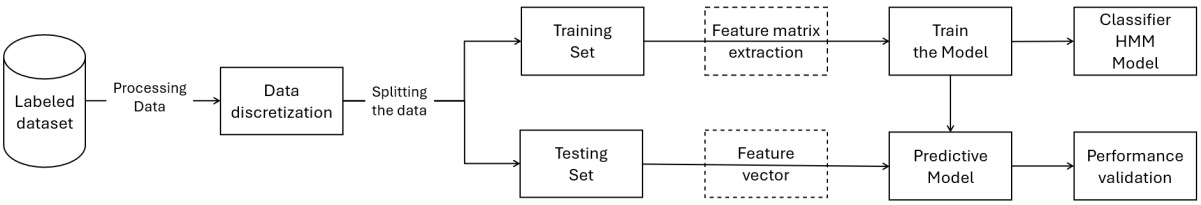

Figure 2. Supervised learning for HMM classifier workflow

2.4. Model Training and Testing Phases

The training phase involves learning the coefficients associated with a set of probabilities $P(Y^k = f_{ik})$. Through this process, we derive a linear combination of probabilities $P(Y^k = f_{ik})$, which can then be used to calculate

the probability of belonging to each class. The performance of these derived equations is subsequently evaluated using a testing set [35].

We now present the testing process using the developed model. At this stage, the model's equations define a relationship between the l-dimensional explanatory variables $\{Y^k; k = 1, 2, \ldots, l\}$ and the posterior probabilities $P(X = e_i \mid Y)$. Each explanatory variable Y^k is represented in the model by a set of categorical symbols $\{f_{1k}, f_{2k}, \ldots, f_{mkk}\}$. Notably [35]:

$$P(Y^k = f_{ik}) = \begin{cases} 1 & \text{if } Y^k = f_{ik} \\ 0 & \text{otherwise} \end{cases}$$

Each observation is represented by a vector of categorical symbols. For each equation in the model, there is a corresponding vector of coefficients. The classification procedure involves assigning each element from the test data to the most probable group based on these probabilities.

3. Discritization methods

In our study, we consider 10 differents methods of discretization in depth that will be applied within our classification framework: [18]

3.1. Equal Width Interval

This technique involves first determining the range of observed values for a given continuous attribute, delimited by its minimum (ymin) and maximum (ymax) values. The entire range is then divided into k equally sized intervals $I_i (i = 1, \ldots, k)$, where k is a parameter defined by the user. Each interval represents a bin into which data points are categorized based on their values. This method is a global, static and operates independently of any labels, making it an unsupervised method . Then this method computes the bin width [18, 37].

$$Interval = \frac{y_{\max} - y_{\min}}{k}$$

. The boundaries between bins are determined using the specified formula, where i ranges from 1 to $k - 1$ [38, 39].

$$Boundaries = y_{\min} + (i * interval)$$

By dividing data into these uniform bins, equal-width discretization facilitates the analysis and interpretation of continuous variables in a variety of analytical contexts.

3.2. Equal Frequency

Equal frequency discretization is a methodological approach for dividing numerical attributes into k intervals, where each interval, I_i for $i = 1, \ldots, k$, includes approximately $\frac{n}{k}$ contiguous observations from a sorted list of n observed values. This technique provides approximately the same distribution of data across all intervals, making it ideal for analyzing datasets with varied distributions. It is a global, unsupervised and static method [39, 11].

3.3. k-means clustering

Cluster analysis, also known as clustering, is a static method used to discretize a numeric attribute, Y, by dividing the values of Y into k clusters $C_i \subset Y$ and $C_i \cap C_j = \varnothing$ for $(1 \leq i, j \leq k)$, each cluster is represented by the centroid of the data points in the cluster [41, 40].

Partitioning datasets into clusters entails determining the minimum squared error between individual data points and the mean of each cluster. Subsequently, each data point is assigned to the nearest cluster center based on this

criterion [42].

$$J(C) = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|^2$$

where, μ_i the mean of each cluster. The K-means algorithm aims to minimize the sum of the square error for each k cluster [42].
This static method offers advantages over the Equal Width Discretization (EWD) and Equal Frequency Discretization (EFD) methods as it aggregates data according to inherent characteristics. However, this approach also presents certain limitations. Firstly, it is a parametric method, necessitating that users specify the number of intervals. Secondly, the discretization intervals are contingent upon the specified k value and the seed value of the cluster [43].

3.4. Minimum Description Length Principle (MDLP)

This methodology, as introduced by Fayyad et al. [52], employs a division-based approach to compute entropy a quantitative measure of the information content intrinsic to the class labels [38, 41]. Entropy, H , for a dataset is defined as:

$$H = -\sum_{i=1}^{n} p(x_i) \log p(x_i)$$

where $p(x_i)$ represents the probability of occurrence of class label x_i.

The discretization technique assesses potential cut points, defined as the midpoints between each successive pair of sorted attribute values. The selection of the optimal cut point is guided by the criterion of minimal entropy, thereby ensuring the most informative bipartition of data. Following this, the attribute value range is divided into two intervals, which are further partitioned recursively. The discretization process terminates when the Minimum Description Length (MDL) criterion, which balances the complexity and fit of the model to the data, is satisfied [11]. This approach allows for an efficient handling of continuous attributes, facilitating improved performance in subsequent analytical tasks.

3.5. Class Attribute Interdependency Maximization (CAIM)

Kurgan et al. introduced the Class-Attribute Interdependence Maximization (CAIM) algorithm, a supervised discretization method characterized by its global, static, and top-down approach. This algorithm is designed to optimize the interdependence between class labels X and and a discretization variable D applied to an attribute Y, enhancing the efficacy of data classification in various analytical contexts [19, 11].
Fo given quanta matrix [19], the CAIM criterion is mathematically expressed as:

$$\text{CAIM}(X, D(Y)) = \frac{\sum_{r=1}^{n} \left(\max_i M_{ir}^2\right)}{n} \tag{1}$$

where n represents the number of intervals; r iterates across these intervals, from 1 to n; $\max_i M_{ir}$ denotes the maximum value found in the r-th column of the quanta matrix across all i indices, where i ranges from 1 to S, and M_{ir} quantifies the total count of continuous attribute values of Y situated within the interval $[d_{r-1}, d_r]$ [45] .

3.6. Optimal binning

Optimal binning is a technique developed for the multi-interval discretization of continuous-value variables within the context of classification learning. This method transforms continuous features into discretized or nominal variables to achieve optimal data fitting. The approach was pioneered by Usama Fayyad in the year 1993 [46].
In a specific implementation, the optimal binning process comprises two principal stages. Initially, a pre-binning phase produces a granular initial discretization of each variable. This is followed by a refinement or optimization stage, where the discretization is adjusted to meet predefined constraints. The adjustment and transformation of features are complemented by the aggregation of bin ranges for each variable, thereby optimizing the model's efficiency and precision.[47].

3.7. Autonomous Discretization Algorithm (Ameva)

The Autonomous Discretization Algorithm (Ameva) employs a top-down approach based on the chi-square (chi^2) statistic. A distinctive characteristic of Ameva is its minimal number of intervals. Being a top-down method, Ameva starts the discretization process with a single interval, which is successively subdivided into multiple intervals according to the Ameva criterion. This process is mathematically represented as follows [49, 50]:

$$\text{Ameva}_k = \frac{X^2(k)}{k(l-1)} \tag{2}$$

Where, k denotes the interval currently selected for subdivision, and l represents the total number of classes. This iterative subdivision continues until the criteria outlined by the Ameva algorithm are satisfied, facilitating an efficient discretization of the data set.

3.8. Entropy minimization discretization (EMD)

In the empirical mode decomposition (EMD) process, candidate cut points are identified as the midpoints between each consecutive pair of sorted values. For the assessment of each candidate cut point, the dataset is segmented into two intervals, following which the associated class information entropy is computed. The optimal cut point is then determined by identifying the minimum entropy among all candidates. This binary discretization strategy is recursively applied, consistently opting for the optimal cut point. Furthermore, a minimum description length (MDL) criterion is utilized to ascertain the termination point for the discretization process. This methodological approach ensures a systematic and efficient partitioning of data based on entropy minimization, facilitating enhanced data analysis in various scientific applications [51, 52] .

3.9. ChiMerge

The ChiMerge [48] algorithm comprises two primary phases: initialization and a bottom-up merging process. Initially, training examples are sorted by their attribute values for discretization. Each example is allocated to a distinct interval, demarcated by boundaries placed before and after each example. The merging process involves a cyclical repetition of two main steps. In the first step, the χ^2 (Chi-squared) statistic is computed for each pair of adjacent intervals using the formula [1]:

$$\chi^2 = \sum_{i=1}^{m}\sum_{j=1}^{k}\frac{(A_{ij}-E_{ij})^2}{E_{ij}}$$

where:

- $m = 2$ (the two intervals under comparison),
- k denotes the number of classes,
- A_{ij} is the count of examples in the i-th interval and j-th class,
- $R_i = \sum_{j=1}^{k} A_{ij}$ represents the total examples in the i-th interval,
- $C_j = \sum_{i=1}^{m} A_{ij}$ is the count of examples in the j-th class,
- $N = \sum_{i=1}^{k} C_j$ is the overall number of examples,
- E_{ij}, the expected frequency of A_{ij}, is calculated as $\frac{R_i \times C_j}{N}$.

The second step entails merging the adjacent interval pair with the lowest χ^2 value. This process is continued until the χ^2 values for all interval pairs exceed a predetermined threshold, χ^2-threshold. This threshold is established based on a desired significance level and derived from either a statistical table or a mathematical formula.

4. Experimental results

4.1. Datasets

In the present study, we have curated a collection of 10 datasets from the UCI Machine Learning Repository, each representing a distinct domain problem.Table 1 presents some basic information of the 9 datasets, each of which has different characteristics [1].
These datasets comprise features ranging from 54 to 5000. The number of categorical features varies between 0 and 243, and the number of numerical features spans from 10 for the 'Covtype' dataset to 5000 for the 'Gisette' dataset. Moreover, these datasets consist of 2, 7, 10, or 26 target labels. Additionally, the collection includes datasets of varying sizes, ranging from a small dataset with 2,000 records in the 'mfeat' dataset to a relatively larger dataset with 581,012 records in the 'Covtype' dataset [21].
The **Covtype** dataset originates from the Department of Forest Sciences at the College of Natural Resources, Colorado State University. It aims to predict forest cover types using cartographic variables from four wilderness areas within the Roosevelt National Forest in northern Colorado. Another dataset, **MNIST**, comprises images of handwritten digits, while **Isolet** includes images of handwritten English letters. The **ds1.10** dataset, a collection from the life sciences domain, encapsulates a series of experiments in chemistry and biology, with each row representing an experiment and the outputs indicating the reactivity of observed compounds.
Additionally, the **MFEAT** dataset contains images of handwritten digits, and the **CS578** dataset addresses a real-world problem in ornithology. The **GISETTE** dataset focuses on the recognition of handwritten digits, specifically designed to differentiate between the easily confusable digits '4' and '9'. This dataset is part of the NIPS 2003 feature selection challenge. The **optdigits** dataset consists of normalized bitmaps of handwritten digits derived from preprinted forms, contributed by a cohort of 43 individuals. Finally, the **MADELON** dataset represents an artificial construct set on the vertices of a five-dimensional hypercube, involving data points grouped into 32 clusters and arbitrarily labeled +1 or -1, posing a two-class classification challenge with continuous input variables [11].

Table 1. Basic Information for the 9 Datasets.

ID	DATASET	#Variables	#Classes	N_{num}	N_{cat}	#Samples
1	Covtype	54	7	10	44	581012
2	optdigits	64	10	56	8	5620
3	ds100	100	2	100	0	26733
4	CS578	176	2	143	33	75000
5	Madelon	500	2	500	0	2600
6	ISOLET	617	26	617	0	7797
7	mfeat	649	10	406	243	2000
8	mnist	784	10	784	0	70000
9	Gisette	5000	2	5000	0	7000

4.2. Experimental setup

In this section, we evaluate the performance of our methods using 10-5 fold cross-validation and report the mean classification accuracy of the differents datasets [21] . Each one underwent a separate discretization process during the training phase, employing various methods outlined in the subsequent section. The resulting intervals were then utilized to facilitate the training phase. Parameters of the model were learned from the training data across all nine datasets. Subsequently, the refined model was employed to classify new instances. Performance metrics such as accuracy (ACC), area under the ROC curve (AUC), average rank, and standardized ranking ratio (SRR) were utilized to assess and compare the efficacy of each discretization method in enhancing the predictive performance of the models.

4.3. Results Analysis

Table 2. Comparative Performance Metrics for Different Discretization Methods

Data ID	Equal Width			Equal Frequency			K-means			Mdlp		
	ACC	STD	AUC	ACC	STD	AUC	ACC	STD	AUC	ACC	STD	AUC
1	52.76	19.03	54.38	33.89	9.78	58.26	46.01	2.86	59.55	45.30	9.10	58.53
2	88.06	1.20	98.21	87.79	1.90	98.21	87.89	1.92	98.21	87.95	1.38	98.22
3	93.94	3.24	77.81	96.30	2.25	80.18	95.70	2.66	79.57	94.65	2.98	93.95
4	60.85	18.10	69.89	60.72	18.39	69.79	60.79	18.22	69.95	60.52	17.86	69.91
5	61.11	1.54	64.85	60.96	2.12	64.93	60.53	2.16	65.40	62.03	2.75	68.81
6	82.85	3.17	97.75	83.21	3.17	97.75	82.98	3.02	97.84	81.57	2.59	97.61
7	95.87	1.65	99.36	94.37	1.49	99.34	95.62	0.47	99.36	92.62	0.853	99.24
8	76.89	1.50	81.33	75.90	1.68	80.23	77.89	1.30	81.93	76.68	1.67	81.88
9	87.28	1.51	90.43	80.31	1.78	82.73	87.30	1.31	92.43	86.79	1.45	94.87

Table 3. Comparative Performance Metrics for Various Discretization Methods

Data ID	CAIM			Decision Tree			Optimal Binning			AMEVA		
	ACC	STD	AUC	ACC	STD	AUC	ACC	STD	AUC	ACC	STD	AUC
1	42.50	13.61	50.55	34.47	14.51	55.30	50.50	13.50	49.50	48.26	14.05	47.01
2	87.18	1.50	96.91	87.47	1.45	98.21	89.00	1.40	92.00	87.81	1.69	90.87
3	91.88	3.76	97.21	93.44	3.20	88.25	92.00	4.00	97.50	91.13	4.64	97.02
4	56.69	6.98	59.50	60.60	18.26	69.79	59.00	6.50	62.00	57.06	6.10	60.54
5	63.61	5.26	69.50	76.50	3.20	84.93	68.00	5.50	72.00	66.07	5.80	70.32
6	80.87	2.38	97.24	84.19	4.23	98.59	84.00	2.50	82.00	82.55	2.77	80.34
7	93.00	1.95	99.33	93.34	0.88	99.35	94.50	1.22	99.35	94.90	1.52	98.72
8	72.29	3.51	80.56	89.27	2.05	93.78	78.00	2.00	81.00	76.32	2.15	79.00
9	85.80	1.96	89.76	92.15	1.89	97.35	88.00	1.70	88.50	86.77	1.76	86.00

Table 4. Comparative Performance Metrics for EMD and ChiMerge Methods

Data ID	EMD			ChiMerge		
	ACC	STD	AUC	ACC	STD	AUC
1	45.25	13.85	59.63	44.50	13.70	50.32
2	85.10	2.75	89.25	84.75	0.83	89.31
3	95.00	0.70	99.31	95.56	2.49	99.29
4	65.25	4.65	79.44	57.03	5.88	59.34
5	61.50	4.90	69.56	70.00	12.88	79.41
6	85.75	9.05	87.69	81.23	2.79	89.46
7	93.50	1.29	99.32	94.65	0.94	99.04
8	74.50	6.65	89.56	75.66	2.40	78.41
9	95.00	0.70	99.52	87.07	1.87	91.39

4.4. Results and Discussion

Tables II through V report the ten–fold cross–validated performance (mean $\pm$ standard deviation) of the HMM classifier trained with ten discretization strategies across nine benchmark datasets. Two complementary measures are used: (i) Accuracy (ACC), representing the proportion of correctly classified instances; and (ii) the Area Under the ROC Curve (AUC), quantifying the model's ability to discriminate between classes. Statistical validation of the differences among methods was conducted using the Friedman and Nemenyi tests (Table 5).

Equal-Width vs. Equal-Frequency. Equal-Width (EWD) and Equal-Frequency (EFD) discretization exhibit dataset-dependent behavior. EFD notably improves performance on *DS100* (ACC = 96.30%, AUC = 80.18%) compared to EWD (93.94%, 77.81%), indicating that frequency-balanced intervals are better suited to skewed data distributions. Conversely, *Covtype* shows a slight decrease in ACC but a minor AUC gain under EFD, suggesting a trade-off between precision and separability. These findings confirm that uniform-width partitioning preserves numeric continuity, whereas frequency-based partitioning adapts more effectively to heterogeneous densities.

K-means vs. MDLP. K-means and MDLP yield similar AUC values but differ in accuracy. K-means achieves higher ACC on low-dimensional datasets (e.g., Dataset 3: 95.70%), while MDLP provides superior AUC on complex data such as *Gisette*. Entropy-based MDLP minimizes conditional uncertainty between attributes and classes, generating cut points that align well with the emission probabilities $C_{ij} = P(Y_t = f_j | X_t = e_i)$ of the HMM. In contrast, K-means optimizes intra-cluster compactness, favoring variance homogeneity. Hence, K-means enhances structural coherence, whereas MDLP strengthens class-based discrimination.

CAIM vs. Decision Tree. CAIM achieves higher mean ACC and lower variability across datasets 1, 3, 5, and 9, confirming its robustness under balanced class priors. The Decision-Tree (DT) discretization method, however, performs better on high-dimensional and heterogeneous datasets (*Covtype*, *Gisette*), due to its recursive impurity-reduction mechanism. CAIM's global optimization of class–attribute interdependence enhances emission homogeneity, but its performance degrades in imbalanced settings where the quanta matrix becomes unevenly distributed. Overall, CAIM favors consistency, while DT offers flexibility at the cost of higher variance.

Optimal Binning vs. AMEVA. Table V shows that Optimal Binning (OBin) outperforms AMEVA in datasets 2, 4, 7, and 9. OBin's mathematical programming formulation optimizes cut points by minimizing within-bin variance under monotonic constraints, resulting in stable emission matrices C. AMEVA performs competitively on low-variance datasets (e.g., 1, 3, 5) where its χ^2-based analytical rule provides balanced partitions. AUC results are mixed: AMEVA yields stronger separability for datasets 1 and 7, while OBin dominates on 4 and 9. Both methods exhibit low standard deviations ($< 6\%$), confirming robustness. In practice, OBin delivers the best balance between predictive accuracy, stability, and interpretability in HMM preprocessing.

EMD vs. ChiMerge. Entropy-Minimization Discretization (EMD) consistently outperforms ChiMerge in accuracy across nearly all datasets except *Gisette*. EMD achieves outstanding performance on datasets 3, 7, and 9 (ACC $\approx$ 95%, AUC $\approx$ 99%), highlighting its ability to capture fine-grained distributional shifts crucial for posterior-probability estimation. ChiMerge, based on the χ^2 merging criterion, remains competitive in AUC for datasets 3 and 7, demonstrating resilience to noise and moderate imbalance. EMD also exhibits lower standard deviations, confirming stronger reproducibility across repetitions.

Statistical Validation. A Friedman test revealed statistically significant differences among discretization methods ($Q = 20.297$, Iman–Davenport $F = 2.675$, $p = 0.0097$). The Nemenyi post-hoc test ($CD = 4.732$, $\alpha = 0.05$) identified CAIM as significantly inferior to Optimal Binning (OBin) and Equal-Width (EWD), while no other pair exceeded the critical difference. Accordingly, OBin and EWD form the leading group, confirming the advantage of optimization-driven and uniform discretization strategies for HMM-based classification.

Table 5. Statistical validation of discretization methods using Friedman and Nemenyi tests across nine datasets.

Method	**Average Rank**	**Rank (Best→Worst)**
Optimal Binning	3.78	1
Equal Width	3.89	2
K-means	4.11	3
Decision Tree	4.67	4
Entropy Minimization	5.00	5
AMEVA	6.00	6
MDLP	6.11	7
Equal Frequency	6.33	8
ChiMerge	6.44	9
CAIM	8.67	10
Friedman / Iman–Davenport: $Q = 20.297$, $F = 2.675$, $p = 0.0097$		
Nemenyi post-hoc: $CD = 4.732$ at $\alpha = 0.05$		
Significant pairs: CAIM vs. OBin, CAIM vs. EWD		

Interpretation and Implications. Discretization directly affects the emission matrix C and, consequently, the posterior probability $P(X|Y)$. Coarse partitions reduce discriminative power, while excessively fine ones produce ill-conditioned matrices. Empirically, a moderate number of bins ($k = 10$) provided the most stable results. Entropy-based methods (MDLP, AMEVA) improve separability but may suffer from sparsity; optimization-based methods (OBin, EMD) offer both stability and accuracy. The experimental evidence underscores that discretization should be chosen according to dataset properties—dimensionality, skewness, and imbalance—to ensure reliable HMM classification performance.

5. Conclusion

Our investigation focused on evaluating and comparing the efficacy of various discretization methods applied to the HMM classifier. Through systematic analysis, as detailed in the results from Tables, it was evident that certain discretization techniques significantly enhance the performance of our classification model.

REFERENCES

1. C.-F. Tsai, and Y.-C. Chen, *The optimal combination of feature selection and data discretization: An empirical study*, Information Sciences, vol. 505, pp. 282–293, 2019.
2. H. Liu, et al., *Discretization: An enabling technique*, Data Mining and Knowledge Discovery, vol. 6, pp. 393–423, 2002.
3. M. Aly, *Survey on multiclass classification methods*, Neural Networks, vol. 19, no. 1-9, pp. 2, 2005.
4. P. Blunsom, *Hidden markov models*, Lecture Notes, August 15.18-19, pp. 48, 2004.
5. R. S. Mamon and R. J. Elliott, eds., *Hidden Markov models in finance*, New York: Springer, vol. 4, 2007.
6. Y. Zeng and S. Wu, eds., *State-space models: Applications in economics and finance*, Springer Science & Business Media, vol. 1, 2013.
7. H.-K. Lee and J.-H. Kim, *An HMM-based threshold model approach for gesture recognition*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 21, no. 10, pp. 961–973, 1999.
8. M. Inoue and N. Ueda, *Exploitation of unlabeled sequences in hidden Markov models*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, no. 12, pp. 1570–1581, 2003.
9. L. R. Bahl, F. Jelinek, and R. L. Mercer, *A maximum likelihood approach to continuous speech recognition*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 2, pp. 179–190, 1983.
10. A. Krogh, et al., *Hidden Markov models in computational biology: Applications to protein modeling*, Journal of Molecular Biology, vol. 235, no. 5, pp. 1501–1531, 1994.

11. B. Benyacoub, et al., *A comparative study of discretization method for HMM classifiers*, 2018 4th International Conference on Optimization and Applications (ICOA), IEEE, 2018.
12. S. Wang, et al., *A semi-supervised adaptive discriminative discretization (SADD)*, Expert Systems with Applications, Elsevier, 2023.
13. Y. Liu, et al., *Supervised discretization of continuous-valued attributes for classification*, Expert Systems with Applications, Elsevier, 2023.
14. M.-W. Huang, et al., *Combining data discretization and missing value imputation for incomplete medical datasets*, PLOS ONE, 2023.
15. R. Ranjan, et al., *Interaction effect between data discretization and data resampling for class-imbalanced medical datasets*, Concurrent Engineering, SAGE Publications, 2024.
16. C.-J. Tsai, C.-I. Lee, and W.-P. Yang, *A discretization algorithm based on class-attribute contingency coefficient*, Information Sciences, vol. 178, no. 3, pp. 714–731, 2008.
17. S. Ramírez-Gallego, et al., *Multivariate discretization based on evolutionary cut points selection for classification*, IEEE Transactions on Cybernetics, vol. 46, no. 3, pp. 595–608, 2015.
18. J. Dougherty, R. Kohavi, and M. Sahami, *Supervised and unsupervised discretization of continuous features*, Machine Learning Proceedings 1995, Morgan Kaufmann, pp. 194–202, 1995.
19. L. A. Kurgan and K. J. Cios, *CAIM discretization algorithm*, IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 2, pp. 145–153, 2004.
20. J. Dougherty, R. Kohavi, and M. Sahami, *Supervised and unsupervised discretization of continuous features*, Machine Learning Proceedings 1995, Morgan Kaufmann, pp. 194–202, 1995.
21. E. Toulabinejad, M. Mirsafaei, and A. Basiri, *Supervised discretization of continuous-valued attributes for classification using RACER algorithm*, Expert Systems with Applications, vol. 244, 2024.
22. M. R. Chmielewski and J. W. Grzymala-Busse, *Global discretization of continuous attributes as preprocessing for machine learning*, International Journal of Approximate Reasoning, vol. 15, no. 4, pp. 319–331, 1996.
23. S. Wang, et al., *A max-relevance-min-divergence criterion for data discretization with applications on Naive Bayes*, Pattern Recognition, vol. 149, 2024.
24. S. Garcia, et al., *A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning*, IEEE Transactions on Knowledge and Data Engineering, vol. 25, no. 4, pp. 734–750, 2012.
25. N. Peker and C. Kubat, *Application of Chi-square discretization algorithms to ensemble classification methods*, Expert Systems with Applications, vol. 185, 2021.
26. P. Blunsom, *Hidden markov models*, Lecture Notes, August 15.18-19, pp. 48, 2004.
27. S. Z. KHANGHAH and K. MAGHOOLI, *EMOTION RECOGNITION FROM HEART RATE VARIABILITY WITH A HYBRID SYSTEM COMBINED HIDDEN MARKOV MODEL AND POINCARE PLOT*, Applied Computer Science, vol. 20, no. 1, pp. 106–121, 2024.
28. K. Safi, et al., *Hidden Markov Model for Parkinson's Disease Patients Using Balance Control Data*, Bioengineering, vol. 11, no. 1, pp. 88, 2024.
29. A. Ravari, S. F. Ghoreishi, and M. Imani, *Optimal Inference of Hidden Markov Models Through Expert-Acquired Data*, IEEE Transactions on Artificial Intelligence, 2024.
30. A. Martino, G. Guatteri, and A. M. Paganoni, *Multivariate hidden Markov models for disease progression*, Statistical Analysis and Data Mining: The ASA Data Science Journal, vol. 13, no. 5, pp. 499–507, 2020.
31. D. Zamalieva, A. Yilmaz, and T. Aldemir, *Online scenario labeling using a hidden Markov model for assessment of nuclear plant state*, Reliability Engineering & System Safety, vol. 110, pp. 1–13, 2013.
32. I. Saadi, et al., *Hidden Markov Model-based population synthesis*, Transportation Research Part B: Methodological, vol. 90, pp. 1–21, 2016.
33. B. Ouriarhli, B. Benyacoub and H. Benazza, *A Multiclass Classifier Based on Hidden Markov Model with Left Inverse Algorithm*, *International Journal of Data Science and Analytics*, Springer Nature Switzerland AG, DOI: 10.1007/s41060-025-00809-9, 2025.
34. B. Esmael, et al., *Improving time series classification using Hidden Markov Models*, 2012 12th International Conference on Hybrid Intelligent Systems (HIS), IEEE, 2012.
35. B. Benyacoub, et al., *Classification with hidden markov model*, Applied Mathematical Sciences, vol. 8, no. 50, pp. 2483–2496, 2014.
36. K. P. Murphy, *Machine learning: a probabilistic perspective*, MIT press, 2012.
37. K. Das and O. P. Vyas, *A suitability study of discretization methods for associative classifiers*, International Journal of Computer Applications, vol. 5, no. 10, 2010.
38. M. Hacıbeyoğlu and M. H. Ibrahım, *Comparison of the effect of unsupervised and supervised discretization methods on classification process*, International Journal of Intelligent Systems and Applications in Engineering, vol. 4, Special Issue-1, pp. 105–108, 2016.
39. S. J. Jonnalagadda, *Global Entropy Based Greedy Algorithm for discretization*, The University of Texas Rio Grande Valley, 2016.
40. T. H. Dwiputranto, N. A. Setiawan, and T. B. Adji, *Rough-set-theory-based classification with optimized k-means discretization*, Technologies, vol. 10, no. 2, pp. 51, 2022.
41. S. Garcia, J. Luengo, and F. Herrera, *Data preprocessing in data mining*, Springer International Publishing, vol. 72, Cham, Switzerland, 2015.
42. A. M. Ikotun, et al., *K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data*, Information Sciences, vol. 622, pp. 178–210, 2023.
43. P. Chaudhari, et al., *Discretization of temporal data: a survey*, arXiv preprint arXiv:1402.4283, 2014.
44. U. M. Fayyad and K. B. Irani, *Multi-interval discretization of continuous-valued attributes for classification learning*, Ijcai, vol. 93, no. 2, 1993.
45. S. V. Vora and R. G. Mehta, *MCAIM: Modified CAIM discretization algorithm for classification*, International Journal of Applied Information Systems, vol. 3, no. 5, pp. 42–50, 2012.
46. T. J. Cleophas, et al., *Optimal Binning*, Machine Learning in Medicine: Part Three, pp. 37–46, 2013.

47. G. Navas-Palencia, *Optimal binning: mathematical programming formulation*, arXiv preprint arXiv:2001.08025, 2020.
48. R. Kerber, *Chimerge: Discretization of numeric attributes*, Proceedings of the tenth national conference on Artificial intelligence, 1992.
49. L. Gonzalez-Abril, et al., *Ameva: An autonomous discretization algorithm*, Expert Systems with Applications, vol. 36, no. 3, pp. 5327–5332, 2009.
50. K. Lavangnananda and S. Chattanachot, *Study of discretization methods in classification*, 2017 9th International Conference on Knowledge and Smart Technology (KST), IEEE, 2017.
51. Y. Yang and G. I. Webb, *A comparative study of discretization methods for naive-bayes classifiers*, Proceedings of PKAW, vol. 2002, 2002.
52. U. M. Fayyad and K. B. Irani, *Multi-interval discretization of continuous-valued attributes for classification learning*, Ijcai, vol. 93, no. 2, 1993.
53. B.-B. Jia, J.-Y. Liu, and M.-L. Zhang, "Pairwise statistical comparisons of multiple algorithms," *Frontiers of Computer Science*, vol. 19, no. 12, 2025. DOI: 10.1007/s11704-025-41325-0.