# Influence of discretization granularity on learning classification models

Thi Hoang Anh Tran[1][0000−0001−7310−9371], Malina Lara Wiesner[1][0000−0003−2041−8407], and Dr.Ir. Maurice van Keulen[2][0000−0003−2436−1372]

[1] University of Twente, Drienerlolaan 5, 7522 NB Enschede, The Netherlands
{tranthihoanganh,m.l.wiesner}@student.utwente.nl
[2] University of Twente, Drienerlolaan 5, 7522 NB Enschede, The Netherlands
m.vankeulen@utwente.nl

**Abstract.** Discretization has been widely used in data pre-processing for machine learning algorithms to transform continuous attributes into categorical ones to improve model performance and explainability. Although it is well-known that discretization should maintain the distribution and patterns of the original continuous attribute, the actual influence of the resulting granularity of the various discretization methods on machine learning is largely an open question, both in terms of the number of categories as well as the determined ranges. We study this influence specifically for three classification algorithms through sensitivity analysis of parameters of five discretization methods (both supervised and unsupervised) on five datasets. Empirical results show that the influence of granularity in unsupervised discretization on the performance of classification models is more visible than that in supervised discretization. Specifically, there is a trade-off between classification/discretization bias and variance as the number of intervals increases. In addition, the negative correlation between interval inconsistency and classification accuracy is confirmed through experiments with unsupervised discretization. This research gives insight into how discretization methods and their parameters can be combined with machine learning models to improve model performance.

**Keywords:** explainable AI · machine learning · discretization · bin size · sensitivity · classification · granularity.

## 1 Introduction

Discretization has been widely used in data pre-processing for machine learning algorithms for improving model performance. Discretization algorithms have been well studied, but the granularity of the discretization method and its influence on machine learning models remains unanswered. This is the main motivation for us to conduct this research, to answer two questions:

- **RQ1: To what extent is the discretization output sensitive to the change in discretization parameters?**
- **RQ2: To what extent are machine learning models sensitive to the granularity of the discretizers used in pre-processing?**

Based on the literature review, we select representatives of discretization methods with flexibility in setting parameters. Then, we design the experiment with pre-defined discretization settings in combination with classification algorithms. We evaluate both intrinsic properties and model-related metrics.

Another contribution of this research is to provide a guideline on how to combine discretizers setting with classification models to achieve robustness in large datasets.

## 2 Related work

### 2.1 Nature of discretization

Data pre-processing is an important stage in machine learning (ML), which can influence the learning outcome significantly depending on the steps taken [6]. Discretization, as part of data pre-processing, transforms continuous attributes into discrete ones by assigning them to categorical intervals [6, 10, 14]. In other words, this technique transforms quantitative data into qualitative data [6]. *"Ideally, discretization should result in partitions that reflect the original distribution of the continuous attribute, maintain any patterns in the attribute without adding spurious ones, and are interpretable and meaningful to domain experts"* [18]. In addition, we should always thrive for minimal information loss when discretizing attributes, as the process naturally leads to the loss of some information [22]. Discretization is broadly associated with the two aspects of (1) selecting an appropriate number of intervals and (2) defining a suitable interval width. In ML, researchers can apply manual techniques or use heuristic measures that determine these aspects [6]. For a comprehensive overview of discretization methods, we refer to the work of Garcia et al. (2013) [10], Liu et al. (2002) [16] Gallego et al. (2016) [22] and Yang et al. (2010) [27]. The authors have introduced a taxonomy of 30 different discretization methods based on over 80 initial discretizers existing in related literature. According to this taxonomy, we can broadly distinguish between supervised and unsupervised learning depending on whether or not the discretizer considers class information [10].

### 2.2 Influence of discretization on classification models

Most of the previous studies support discretization as opposed to continuous attributes, underlining the positive effect of discretization in a variety of application contexts, especially in biomedical data analysis [6, 10, 14]. Some popular ML models such as decision trees perform better with discrete values [6, 22]. Recent research by Lin et al (2021) supports discretization as it adds context to the ML task at hand, supporting complexity reduction where controversial information from continuous training data may confuse the model [15]. However, one

**Table 1.** Previous empirical studies investigating the influence of discretization on classification performance

| Literature | # Discretizers | # Datasets | # ML models |
|---|---|---|---|
| Dimic (2008) [4] | Many (supervised and unsupervised) | One | One |
| Liu et al. (2002) [16], Baron et al. (2016) [1], Yang et al. (2009) [26] | Many (supervised and unsupervised) | Many | One |
| Dan et al. (1995) [24], Gupta et al. (2001) [11], Maslove et al. (2013) [18], Garcia et al. (2013) [10] | Many (supervised and unsupervised) | Many | Many |
| Bay et al. (2001) [2], Lavangnananda et al. (2017) [14], Gallego et al. (2016) [22] | Many (supervised) | Many | Many |

empirical study on six discretization methods concluded that using discretization might lead to "severe performance degradation" for supervised models [24].

The influence of discretization is normally linked to the performance of the ML models. We observe from previous empirical studies four types of experimental designs for evaluation purposes, as shown in Table 1. Despite the variation in experiment settings, a common approach to evaluating the effectiveness of discretizers is through the accuracy of ML models and computational time (without separating the time for discretization). Most of the algorithms used in these studies are non-ensemble classifiers such as Naïve Bayes, Decision Tree, KNN, SVM, etc.

Regarding the comparison between supervised and unsupervised discretization, most studies show that supervised discretizers help the ML models to yield higher accuracy than unsupervised discretizers [10, 16]. However, in the study by Yang et al. (2009), the authors introduced two modified versions of unsupervised discretization methods which outperform entropy minimal discretization, a commonly used supervised discretizer, in the context of Naïve Bayes classifier [25, 26]. Thus, it is worth investigating the performance of these discretizers in other classification algorithms to validate their effectiveness. Equally important, the simplicity of these unsupervised discretizers allows users to examine the influence of bin size in the splitting or merging process applied.

### 2.3 Evaluation of granularity parameters of discretization

Most reviewed studies conclude that the choice of discretization methods should depend on the nature of the data, problem context, and discretization task. However, the experiment design in most of these studies only considers the variation of discretization methods, without examining the flexibility in discretization parameters. One example is the number of intervals for unsupervised discretizers.

Specifically, an underlying assumption in these studies is that the number of intervals is optimal by the author's choice [10], or by common practices [26].

Three of the reviewed studies address the intrinsic properties of discretization, including Yang et al. (2009) [26], Garcia et al. (2013) [10], and Maslove et al. (2013) [18]. Specifically, these studies develop KPIs for discretizers such as (internal) consistency, discretization bias, and variance and perform statistical tests such as the Wilcoxon test, and two-sided unpaired t-test to compare discretizers of the same category. In this project, we will consider the intrinsic evaluation and the influence on model performance for each discretization method.

## 3 Experimental design

This section outlines the experimental setup in terms of datasets, discretization methods, and ML models used.

### 3.1 Datasets

Our selection criteria for datasets include:

- Cross-reference: Data reuse in previous research. Datasets with high cross-reference enable a comparison of the experiment outcomes.
- Balanced representation of 3 sizes-small, medium, and large datasets. We define the corresponding sizes as less than or equal to 5,000 instances, from 5,000 to less than 10,000 instances, and more than 10,000 instances.
- Diversity of application domain: As can be seen from Table 2, there are 10 domains, and the final five datasets cover different domains.
- Clean dataset: Since the focus of the experiment is discretization, we want to minimize other pre-processing data steps, such as data imputation for missing values.
- Reproducibility: All datasets are retrieved from public sources such as the UCI ML Repository and OpenML2. Thus, the experiment can be reproduced in future research.

From the literature review, 11 datasets were shorlisted from 130 datasets for the experiment. The summary of 11 datasets is listed in Table 2. We chose five datasets for the experiment, namely: Penbased, Satimage, Iris, Australian, and Pima.

### 3.2 Discretization methods and their parameters

We select discretization methods from the taxonomy by Garcia et al. [10]. With our main goal of investigating the influence of discretization granularity on ML performance, selected discretization methods should allow experimenting with parameters to produce coarse and granular bins. The following criteria were used in our selection of discretizers;

- Flexibility in parameter settings to allow for flexible bin sizing.

**Table 2.** Shortlisted datasets (final datasets are in bold)

| Dataset | Domain | Size | # Attributes | # Continuous attributes | # Classes | Cross-reference |
|---|---|---|---|---|---|---|
| **Iris** | Life | 150 | 4 | 4 | 3 | [10, 11, 14, 16, 26] |
| **Satimage** | Physical | 6,435 | 36 | 36 | 7 | [2, 10, 26] |
| **Penbased** | Computer | 10,992 | 16 | 16 | 10 | [10, 26] |
| **Australian** | Finance | 684 | 14 | 8 | 3 | [10, 11, 16, 26] |
| **Pima** | Health | 768 | 8 | 8 | 3 | [10, 11, 16, 26] |
| Blood | Business | 748 | 4 | 4 | | [11, 14] |
| Adult | Social | 48,827 | 14 | 6 | 2 | [2, 26] |
| Phoneme | Phonetic | 5,404 | 5 | 5 | 2 | [10] |
| Pageblocks | Document | 5,472 | 10 | 10 | 5 | [10] |
| Forest cover type | Life | 581,012 | 54 | 10 | 7 | [26] |
| Musk | Medical | 6,598 | 166 | 166 | 2 | [26] |

- Applicability to popular ML Algorithms to ensure applicability with common ML algorithms (i.e. Naive Bayes, Decision tree)
- Comparability to previous empirical research findings on discretization for comparison
- Availability of discretization method in public libraries

**Supervised discretization**

*Selection of discretizers*

Based on these criteria, we chose two supervised discretizers, namely ChiMerge (CM) [12] and MDLP, a discretizer based on the minimum description length principle [8]. Both methods are well-represented in previous studies, including different application domains. Empirical results from these studies show that both discretizers perform well with different classification algorithms including Naive Bayes and Decision Trees [1, 5, 10, 23].

ChiMerge is viewed as a particularly robust discretization approach as it is less sensitive to noise than other methods [24]. ChiMerge is a bottom-up, merging approach using Chi-square-based statistical evaluation to determine the discrete intervals over continuous data. As a parametric discretizer, ChiMerge requires users to set parameters such as the maximum number of intervals possible and confidence intervals as thresholds [10, 12]. The algorithm begins with sorting the numerical values and calculates the Chi-square value for every pair of adjacent intervals. Next, the discretizer determines when to merge two adjacent intervals looking for the lowest Chi-square value [12, 14]. ChiMerge employs the assigned significance level (and sets the maximum number of intervals) as a stopping

criterion during the discretization process [12, 16]. For a detailed explanation of the algorithm, we refer to the original article by Kerber [12].

MDLP discretization can result in an acceptable trade-off between the number of intervals and classification performance (accuracy) [10]. MDLP discretization is a top-down splitting approach that uses entropy as an evaluation measure. The algorithm is usually non-parametric, determining the number of intervals employing the algorithm itself [10]. MDLP starts with sorting the numeric attributes and uses the minimum description length principle as the criterion to determine the cut point between every pair of adjacent intervals [14]. For a complete explanation of MDLP, we refer to the original article by Fayyad et al. (1993) [8].

Despite the excellent performance, MDLP does not meet the first criteria of flexibility for parameters. Thus, we must find an alternative candidate for experimentation. We found the DecisionTreeDiscretiser by the repository feature-engine, which also implements a decision tree-based discretization.

*Parameter setting of discretizers*

**ChiMerge**

Most of the previous studies we have reviewed use the recommended parameter settings as suggested by Kerber (1992) [12], with significance levels of alpha between 0.01 and 0.1 and a maximum number of intervals between 10 and 15 [10, 11, 16]. Most studies reviewed do not further specify the parameters of ChiMerge, other than the confidence threshold used to produce intervals [10, 11, 16, 21, 24].

In addition, the public libraries for implementing ChiMerge discretization in both Python and R do not allow changing significance levels (alpha). We assume alpha as the underlying control variable in the implementation. Thus, the maximum interval is the final parameter for ChiMerge in our experiments.

**DecisionTreeDiscretiser**

For DecisionTreeDiscretizer[3], the maximum depth of the decision tree as parameter used allows for creating different bin sizes based on the principles of decision trees. Accordingly, the number of bins will be restricted to a maximum of $2^{max\_depth}$, which is suitable for experimenting on the granularity of discretizer [9, 20].

**Unsupervised discretization**

*Selection of discretizers*

Three unsupervised discretization methods are performed in the experiment: Equal Width Discretizer (EDW), Equal Frequency Discretizer (EFD), and Fixed Frequency Discretizer (FFD). EDW and EFD are the widely used unsupervised discretization methods for their simplicity and flexibility in parameter setting.

In EDW, the sorted continuous attribute is divided evenly into pre-defined k intervals whose widths are equal. EFD also uses a pre-defined k number of intervals to split the attribute such that each interval contains approximately

[3] `https://feature-engine.readthedocs.io/en/1.0.x/discretisation/DecisionTreeDiscretiser.html`

the same number of instances [16]. Specifically, if the sample size of the dataset is n, each interval then contains (n/k) training instances with adjacent (possibly identical) values [27].

FFD, developed by Yang et al. (2009) pre-defines a sufficient interval frequency (m). FFD divides the sorted values into intervals so that each interval has approximately the same number of training instances with adjacent (possibly identical) values. Thus, a dataset with a sample size n will have approximately (n/m) intervals after discretization [26].

Although both EFD and FFD form intervals with equal frequency, the underlying methodology of the two discretizers is different. According to Yang et al. (2009), "EFD fixes the interval number that is usually arbitrarily chosen. FFD fixes the interval frequency that is not arbitrary but to ensure each interval contains sufficient instances to supply information such as for estimating probability." [27].

*Parameter setting of discretizers*

**EWD and EFD**: Parameter: number of intervals, k.

In the study by Dougherty et al. (1995), the maximum number of intervals is determined by the formula: which k=max{1, 2×log(l)}, where l is the number of distinct observed values for the attribute that will be discretized. Applying this formula in the selected datasets results in two values for k (3 and 4). However, most of the studies we reviewed used a rule of thumb or common practice with k = 10. The study by Maslove et al. (2013) tests the sensitivity of several bin sizes for unsupervised discretizers. The author concludes that consistency improved very little beyond a bin size of k = 4, and that accuracy improved only sporadically and unpredictably beyond a bin size greater than k = 9 [18]. In our experiment, we will set 3 scenarios for k: k = 4, k = 7 and k = 10 to compare with existing studies [5].

**FFD**: Parameter: interval frequency, m.

In the study by Yang et al. (2009), the author evaluated Naive Bayes's classification bias and variance relative to m in the range from 10 to 100. Accordingly, when m increases, bias increases but variance decreases. m = 30 is used to evaluate FFD against other discretizers [26]. The value 30 is also used in 2 other papers, by Garcia et al. (2013) [10] and Fang et al. (2013) [7]. Thus, for our experiment, we chose 4 values for m: 10, 30, 60, and 100.

**Final discretization setting**

The final set of discretization methods is displayed in Table 3, including the chosen parameters for each one. For each discretization method, the number of models generated is equal to the number of parameter settings * number of datasets * number of algorithms. For example, for ChiMerge, there are *4 * 5 * 3 = 60* models. Thus, the total number of models generated for the whole experiment is 270 models.

**Table 3.** Final experimental set-up for discretization methods (supervised discretizers are in bold)

| Discretization method | Parameter | Final settings |
|---|---|---|
| **ChiMerge (CM)** | Maximum number of Intervals | k= 6, 8, 10, 15 |
| **Decision Tree (DT)** | Maximum depth of the tree max_depth | max_depth = 2, 3, 4, 5 |
| Equal Width (EWD) | Number of intervals | k = 4, 7, 10 |
| Equal Frequency (EFD) | Number of Intervals | k = 4, 7, 10 |
| Fixed Frequency (FFD) | Interval frequency | m = 10, 30, 60, 100 |

### 3.3 Classifiers

Like dataset selection, we prioritize algorithms that have been used in previous empirical studies about discretization to make a comparison for the experiment outcomes. In addition, as mentioned in the previous section (section 2, white box model), we simplify the algorithms by selecting popular classification algorithms: Decision Tree (ID3), KNN with customized distance metrics, and categorical Naïve Bayes (CNB). Since the datasets after the discretization process only have categorical data, we select these algorithms for three reasons:

- First, the algorithm must support the classification problem using only categorical features. Among some popular decision tree algorithms, only ID3 satisfies this criteria.[4]. ID3 has been used in the paper by Dan et. al. (1995) [24] which allows us to compare our experiment result.
- Second, the algorithm must support multiclass classification. We selected KNN, which has been used in two previous studies to compare our result [10, 14]. As our KNN model only uses categorical data, we cannot use default distance metrics. Instead, we use the Value Difference Metric (VDM), which is compatible only with brute force algorithm [5].
- Finally, Categorical Naïve Bayes (CNB) is selected for its popularity. It is one of the top 10 DM algorithms [3] and has been widely used in previous studies about discretization [4,10,11,14,18,22,25]. The Naive Bayes classifier computes the posterior probability of the classes, given the data, assuming independence between features for each class [11].

### 3.4 Evaluation

We perform two types of evaluations: (1) Sensitivity analysis within each discretization method and (2) Pair-wise comparison of discretization methods.
First, the purpose of sensitivity analysis is to explain how the intrinsic properties of the discretization method change corresponding to the change in parameters:

[4] `https://scikit-learn.org/stable/modules/tree.html#tree-algorithms-id3-c4-5-c5-0-and-cart`

[5] `https://scikit-learn.org/stable/modules/neighbors.html#nearest-neighbors-classification`

k - EWD/EFD, m - FFD, maximum interval - ChiMerge, maximum depth - Decision Tree. We use visual analytics for this purpose. The intrinsic properties include:

- Inconsistency rate: two instances are considered inconsistent if they match except for their class labels. The inconsistency rate is calculated by the sum of all the inconsistency counts divided by the total number of instances [16, 17]. We calculate this metric for each dataset after discretization [10].
- Discretization bias and discretization variance. In the study by Yang et al. (2009) the author proposes a theorem that (Naive Bayes) classification bias and variance imply discretization bias and variance. We also use this assumption for our experiments [26]. Specifically, classification error can be decomposed into three components: bias, variance and noise [13]. The noise does not depends on learning algorithms, as opposed to bias and variance. Bias measures the part of errors due to systematic change in the learning algorithm, and variance measures the part of errors due to changes in the training set. Thus, variance increases when the algorithm responds to sensitive changes in the training data.

In addition to intrinsic properties, we look at the influence of discretization parameters on the performance of ML models through classification accuracy. We also evaluate the efficiency of discretization methods by measuring computation time. Unlike most previous studies which only compare total execution time, we evaluate the execution time separately for discretization and time for training models.

Second, we perform statistical tests to compare the performance of discretization methods. The Wilcoxon signed-rank test [19] is chosen for two reasons:

- The experiment setting in this research is matched pairs studies. Specifically, observations are taken on the same subjects (5 datasets) under different conditions (discretizers and training algorithms).
- Wilcoxon signed-rank test[6] is a non-parametric method that does not require the normal distribution of the population.

The null hypothesis is that there is no difference in metric within pairs of models after discretization by method A and by method B. For more detail, please see section 4.

## 4 Experiment results

### 4.1 Sensitivity analysis

We divide our sensitivity analysis into three different aspects: bias-variance trade-off, inconsistencies after discretization vs. classification accuracy, and computation time for discretization vs. for training ML models.

[6] `https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.wilcoxon.html`

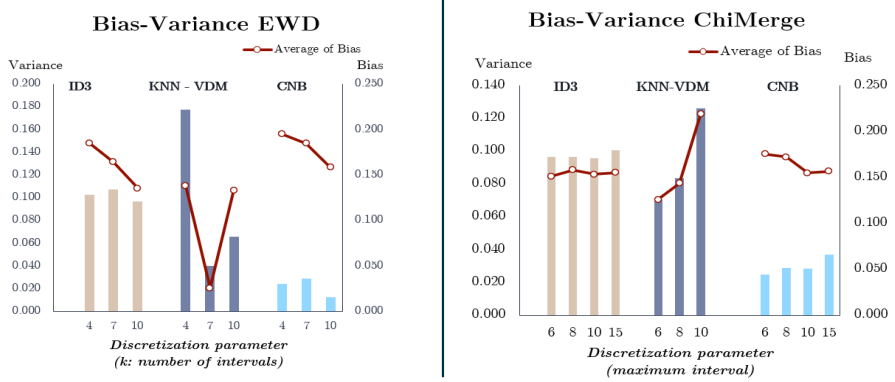


**Fig. 1.** Bias-variance trade - off EWD & ChiMerge discretization

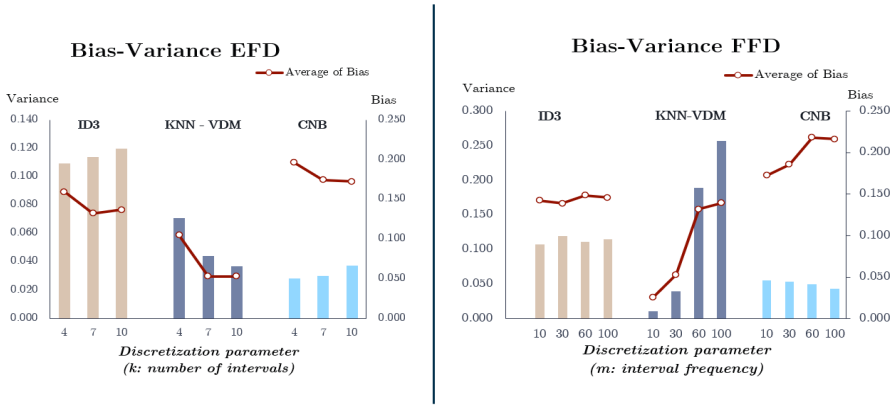


**Fig. 2.** Bias-variance trade - off EFD & FFD discretization

*Bias-variance trade-off*

The experiments conducted show how the discretization bias and variance behave when increasing the intervals (k), or equivalent to decreasing interval frequency (m) depending on the discretization method used. For three unsupervised discretizers (EWD, EFD, FFD) we observe that as number of interval increases, discretization bias decreases regardless classification models (See Figures 1 and 2).

Although discretization variance increases slightly in response to the increase in intervals, the trade-off in bias-variance is confirmed for ID3 and CNB models. Interestingly, in KNN models using data discretized by EFD and FFD, bias and variance change in the same patterns.

Regardless of supervised discretization methods (CM or DT), we observe little change in discretization bias when increasing the number of maximum intervals. The variance shows a similar trend for ID3 and CNB models. However, for KNN models, we can observe a sharp increase in variance when the maximum

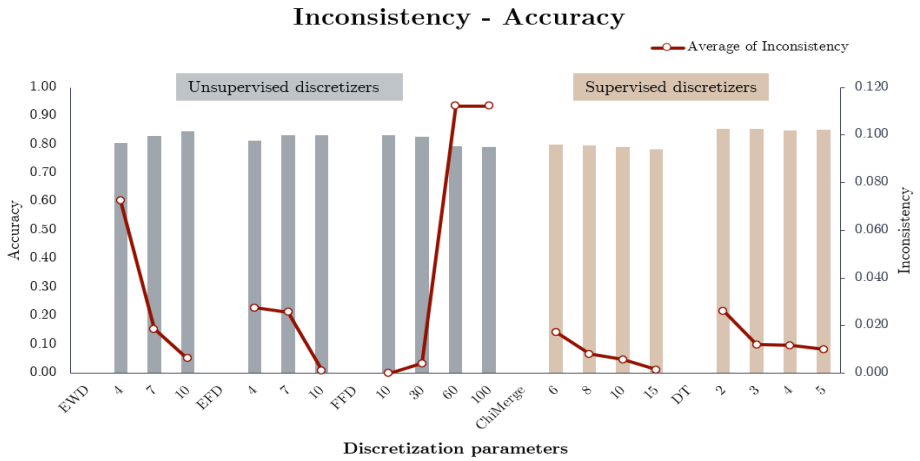


**Fig. 3.** Inconsistency rate - Classification accuracy (all algorithms)

interval increases using CM discretization (see Figure 1) or when max depth increases from 2 to 3 using DT. For KNN-CM combination, we cannot obtain this metric when the maximum interval is 15 due to the long computation time for the brute force algorithm.

*Correlation between internal inconsistency and classification accuracy*

In Figure 3 we plot the average classification accuracy and inconsistency rates of the datasets post discretization.

For three unsupervised discretizers, there is a negative correlation between ML model accuracy and inconsistency rate as the number of intervals increases. Indeed, when data is divided into more bins, classification models get higher accuracy and lower inconsistency rates.

Both supervised discretizers show stable inconsistency rates (close to 0) regardless the change in parameters. However, unlike unsupervised discretization, CM discretizer does not boost classification accuracy. In fact, we observe a steady decrease in accuracy when increasing the number of intervals for CM. This pattern is also observed for DT discretizer when maximum depth increases from 2 to 3 (which means more intervals) and then fades out.

*Computational time*

To evaluate the computational efficiency of discretization in training models, we separate times needed for discretization and for training.

In general, both supervised and unsupervised discretization processes take longer time as the number of intervals increases. In our experiment, this means decreasing m for FFD and increasing parameters for the remaining discretization methods. However, it does not necessarily take longer times to train models as the features fall into more intervals. For example, combining EWD with ID3

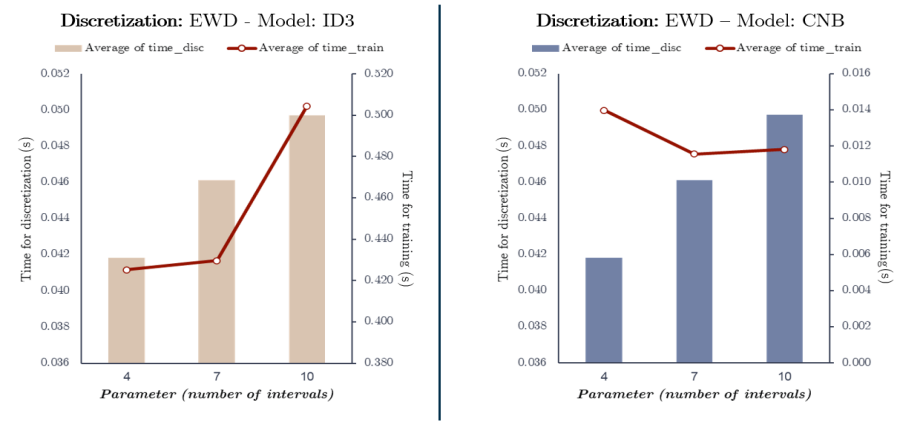


**Fig. 4.** Computation time for EWD: discretization vs. model training

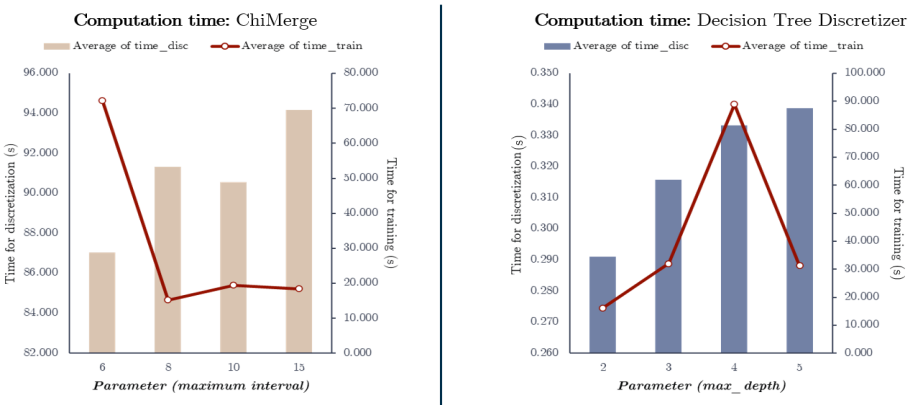


**Fig. 5.** Computation time for supervised discretization vs model training

results in an increase in training time, but this will decrease if we combine EWD with CNB algorithm (see Figure 4).

Using CM in pre-processing helps to reduce time significantly for training models when the maximum interval increases from 6 to 8. Training time slightly increases if the discretization parameter go beyond this level (see Figure 5). For DT, the training time starts falling as the maximum depth reaches 4 (which is the equivalent to maximum of 16 intervals). This pattern appears consistent with the other supervised counterpart (CM).

### 4.2 Pair-wise comparison of discretization methods

We perform two-sided Wilcoxon signed rank test at significance level 5% for the difference within pairs of models receiving different treatment, i.e. different

**Table 4.** Wilcoxon signed rank test results - Variance - Categorical Naive Bayes. ($\alpha = 5\%$)

| Comparison | Test statistic | P-value |
|---|---|---|
| ffd vs dt | 5.00 | 0.004883 |
| ffd vs cm | 10.00 | 0.020996 |
| ffd vs ewd | 12.00 | 0.03418 |
| cm vs efd | 13.50 | 0.04248 |

discretization methods (matched pairs design). The tests are performed at two levels: the general level and the algorithm level. At the general level, pairs of models are defined as models trained by three algorithms on the same datasets ($n_{max} = 60$). At the algorithm level, pairs of models refer to models trained by the same algorithm (CNB/ ID3/ KNN-VDM) on the same datasets ($n_{max} = 20$). Three metrics are tested, namely classification accuracy, classification bias, and classification variance.

*Wilcoxon signed rank test for classification accuracy*: At the general level, we find a significant difference in accuracy for models using FFD and those using DT prior to training process (p-value = 0.0453). At algorithm level, there are significant differences in the accuracy of ID3 model when training data is discretized by DT compared to unsupervised discretizers, including EWD and EFD. P-values of the two tests are 0.015 and 0.030 respectively.

*Wilcoxon signed rank test for classification bias*: Regarding classification bias, at general level, we also obtain similar significant difference between DT and FFD (p-value = 0.008847). In addition, the classification variance is significantly different when comparing unsupervised discretizers, EWD versus EFD. The differences are observed in Categorical Naive Bayes (p-value = 0.00048) and ID3 models (p-value = 0.03417).

*Wilcoxon signed rank test for classification variance*: At general, using FFD makes a significant difference in classification variance compared to those using CM. Adding a classification algorithm as a constraint gives more insight into this difference. Specifically, Categorical Naive Bayes models combining with FFD make a significant difference in classification variance compared to all other discretizers (see Table 4).

## 5 Discussion

Experiment results confirm some similar patterns in the sensitivity of discretization methods as mentioned in previous research. In general, supervised discretizers are less flexible to tune parameters as they look for the optimized discretization based on class information. By contrast, unsupervised discretizers enable flexibility to experiment with different interval numbers and frequencies and also require less computation time.

Similarly, CM implementation also confirms the observations from previous research showing a tendency to produce a larger number of bins if the stop-

ping criterion (maximum number of intervals) does not interfere with the the process [18, 21].

In terms of model performance, DT shows the highest accuracy in models than other discretizers, at all parameter settings. Unlike Yang et al. (2009), the statistical evidence does not confirm that FFD performs better than supervised discretizers (CM or DT) to improve model accuracy. However, our observation regarding the discretization bias-variance trade-off agrees with Yang's findings, indicating the trend of decreasing bias with an increasing number of intervals, independent of the classifier used. The discretization variance, however, behaves differently depending on the classifier, with little change for ID3 and CNB but increasing variance for KNN when the training datasets are discretized into more intervals.

Regarding the inconsistency of the dataset after discretization, we find CM not sensitive to a change in input parameters. The same conclusion applies to the accuracy of classification models using data discretized by CM. This could be explained by CM using class information to discretize. The unsupervised discretization methods show a trend of decreasing inconsistency with an increase in the number of intervals (EWD, EFD) or a decrease in interval frequency (FFD). These results are in line with empirical results from Garcia et al [10].

When using KNN classification, all models needed a significant amount of time for training, in particular for the medium to large datasets used. Another potential issue when using only categorical features for CNB model is the "index-out-of-bound error". This error occurs due to unseen categorical values in test sets and often be seen in models using DT or CM discretization with high parameter settings.

## 6 Conclusion and future research

This research has investigated the influence of supervised and unsupervised discretization on classification models. The experiments have been conducted with five discretization methods (with different parameter settings), on three different classifiers over five different UCI datasets.

Empirical results lead to a conclusion that classification models react more sensitive to unsupervised discretizers, showing higher fluctuations both in data inconsistency after discretization and model accuracy after classification (RQ2). Supervised discretization, in contrast, provides less flexibility in changing bin sizes and turned out less robust in terms of implementation (RQ1).

Another important implication is that there is no perfect combination of the discretization method and ML algorithm because it depends on the nature and size of the dataset. However, we can confirm the trade-off between discretization bias and variance and the correlation between data inconsistency and classification accuracy.

One limitation of this research is that we only used univariate discretization. It could be interesting to experiment in similar settings with multivariate

discretization methods. All discretization methods used are based on a "glue approach", meaning a discretization of the dataset as a whole before performing a train-test split. Future experiments could include a "test-on-learn" approach [1].

To enrich our findings, we propose to expand all dimensions of the current experimental setup, namely discretization methods, classifiers, and datasets. And to get a better understanding of the effect of discretization methods, further observations should be collected to measure information loss induced by discretization, which would enable a richer comparison between different discretizers next to the intrinsic properties.

We define our main contribution as the explanation of discretization efficiency and its influence on popular ML models, hoping to provide guidelines for choosing discretization methods and classifiers depending on the problem context.

## References


1. Baron, G., Harężlak, K.: On Approaches to Discretization of Datasets Used for Evaluation of Decision Systems. In: Czarnowski, I., Caballero, A.M., Howlett, R.J., Jain, L.C. (eds.) Intelligent Decision Technologies 2016. pp. 149–159. Springer International Publishing, Cham (2016)
2. Bay, S.D.: Multivariate discretization for set mining. Knowledge and Information Systems **3**(4), 491–512. https://doi.org/10.1007/PL00011680
3. Benenson, Z., Junger, M., Oliveira, D., Stringhini, G.: Cybersafety threats–from deception to aggression. In: Cybersafety Threats-from Deception to Aggression 2019 (2019)
4. Dimić, G., Rančić, D., Milentijević, I., Spalević, P.: Improvement of the accuracy of prediction using unsupervised discretization method: Educational data set case study. Tehnicki vjesnik - Technical Gazette **25**(2). https://doi.org/10.17559/tv-20170220135853
5. Dougherty, J., Kohavi, R., Sahami, M.: Supervised and unsupervised discretization of continuous features. In: Machine learning proceedings 1995, pp. 194–202. Elsevier (1995)
6. Elhilbawi, H., Eldawlatly, S., Mahdi, H.: The importance of discretization methods in machine learning applications: A case study of predicting icu mortality. In: International Conference on Advanced Machine Learning Technologies and Applications. pp. 214–224. Springer (2021)
7. Fang, J., Sui, L.N., Jian, H.Y.: Comparative analysis of continuous entropy estimation with different unsupervised discretization methods. In: Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering (ICCSEE 2013). pp. 367–370. Atlantis Press (2013/03). https://doi.org/10.2991/iccsee.2013.94
8. Fayyad, U., Irani, K.: Multi-interval discretization of continuous-valued attributes for classification learning (1993)
9. Feature engine developers: DecisionTreeDiscretiser — 1.3.0. `https://feature-engine.readthedocs.io/en/1.3.x/user_guide/discretisation/DecisionTreeDiscretiser.html#decisiontree-discretiser` (2022), [Online; accessed 20-June-2022]
10. García, S., Luengo, J., Sáez, J.A., López, V., Herrera, F.: A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning.

IEEE Transactions on Knowledge and Data Engineering **25**(4), 734–750 (2013). https://doi.org/10.1109/TKDE.2012.35
11. Gupta, A., Mehrotra, K.G., Mohan, C.: A clustering-based discretization for supervised learning. Statistics & Probability Letters **80**(9), 816–824 (2010). https://doi.org/10.1016/j.spl.2010.01.015, https://www.sciencedirect.com/science/article/pii/S0167715210000271
12. Kerber, R.: Chimerge: Discretization of numeric attributes. In: Proceedings of the tenth national conference on Artificial intelligence. pp. 123–128 (1992)
13. Kohavi, R., Wolpert, D.H.: Bias plus variance decomposition for zero-one loss functions. In: MACHINE LEARNING: PROCEEDINGS OF THE THIRTEENTH INTERNATIONAL. pp. 275–283. Morgan Kaufmann Publishers (1996)
14. Lavangnananda, K., Chattanachot, S.: Study of discretization methods in classification. In: 2017 9th International Conference on Knowledge and Smart Technology (KST). pp. 50–55 (2017). https://doi.org/10.1109/KST.2017.7886082
15. Lin, H.Y.: Feature clustering and feature discretization assisting gene selection for molecular classification using fuzzy c-means and expectation–maximization algorithm. The Journal of Supercomputing **77**(6), 5381–5397 (2021)
16. Liu, H., Hussain, F., Tan, C.L., Dash, M.: Discretization: An enabling technique. Data Mining and Knowledge Discovery **6**(4), 393–423. https://doi.org/10.1023/A:1016304305535
17. Liu, H., Setiono, R.: Feature selection via discretization. IEEE Transactions on Knowledge and Data Engineering **9**(4), 642–645 (1997). https://doi.org/10.1109/69.617056
18. Maslove, D.M., Podchiyska, T., Lowe, H.J.: Discretization of continuous features in clinical datasets. Journal of the American Medical Informatics Association **20**(3), 544–553 (10 2012). https://doi.org/10.1136/amiajnl-2012-000929
19. Moore, D., McCabe, G., Craig, B.: Introduction to the Practice of Statistics. Macmillan Learning (2017), https://books.google.nl/books?id=QgtBswEACAAJ
20. Niculescu-Mizil, A., Perlich, C., Swirszcz, G., Sindhwani, V., Liu, Y., Melville, P., Wang, D., Xiao, J., Hu, J., Singh, M., et al.: Winning the kdd cup orange challenge with ensemble selection. In: KDD-Cup 2009 Competition. pp. 23–34. PMLR (2009)
21. Peker, N., Kubat, C.: Application of chi-square discretization algorithms to ensemble classification methods. Expert Systems with Applications **185**, 115540 (2021)
22. Ramírez-Gallego, S., García, S., Mouriño-Talín, H., Martínez-Rego, D., Bolón-Canedo, V., Alonso-Betanzos, A., Benítez, J.M., Herrera, F.: Data discretization: taxonomy and big data challenge. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **6**(1), 5–21 (2016)
23. Ropero, R.F., Renooij, S., Van der Gaag, L.C.: Discretizing environmental data for learning bayesian-network classifiers. Ecological Modelling **368**, 391–403 (2018)
24. Ventura, D., Martinez, T.R.: An empirical comparison of discretization methods. In: In Proceedings of the Tenth International Symposium on Computer and Information Sciences. pp. 443–450 (1995)
25. Yang, Y.: Discretization for Naive-Bayes learning. Thesis (2017). https://doi.org/10.4225/03/59d42df10cb0a, https://bridges.monash.edu/articles/thesis/Discretization_for_Naive-Bayes_learning/5466919
26. Yang, Y., Webb, G.I.: Discretization for naive-bayes learning: managing discretization bias and variance. Machine Learning **74**(1), 39–74. https://doi.org/10.1007/s10994-008-5083-5
27. Yang, Y., Webb, G.I., Wu, X.: Discretization Methods, pp. 101–116. Springer US, Boston, MA. https://doi.org/10.1007/978-0-387-09823-4_6