A Comparison of Binary Discretization and ChiMerge

TingWei Zhang, JiaHui Ni, Chuan Liu
{ftw4zhang, jhni, chuan.liu}@uwaterloo.ca
University of Waterloo
Waterloo, ON, Canada

Abstract

Many machine learning algorithms are limited, requiring training data in categorical values. It raises the problem of converting continuously valued data to categorical values. This introduces the process of discretization. This paper presents empirical comparison between two multi-level discretization methods. One is Binary Discretization Method, and the other approach is ChiMerge Discretization Method. The project introduces two real-world problems which are solved by ID3 algorithm. The discretization methods are used as preprocessing step to convert training data to categorical values, which is fed into further machine learning algorithms. Binary Discretization achieves lower testing accuracy compared to ChiMerge when the data is small. While after performing pruning on the decision trees, both algorithms largely improves its prediction accuracy.

Introduction

When building decision tree, most classification algorithms require the training data to be categorical values. However, for most problems in real life, data provided is quantitative. Even for algorithms that can directly deal with quantitative data, learning is often less efficient and less effective. This introduces the preprocessing step of discretization. Discretization is the process of converting continuously valued data into categorical values by assigning ranges of real values to one category. It can turn numeric attributes into discrete values. The training data is segmented based on certain heuristic, and derives multiple intervals. The method to address discretization process is an important problem.

In this project, we present and compare two multi-level discretization methods. One is Binary Discretization Method, and the other approach is ChiMerge Discretization Method.

For ID3 algorithm introduced in class, the method of discretization is to use Binary Discretization Method. The ranges are set by values between two instances, with different classes in the sequence of instances sorted by a quantitative attribute. It can generally be used in any algorithm for classification, that handles continuously valued attributes by quantizing their ranges into two intervals. Pruning is added at each iteration to remove nodes that are not necessary for the model. The model experiments with different depths, and the optimal tree is selected with the highest testing accuracy. The other method of discretization is to use ChiMerge algorithm. The algorithm uses the χ^2 (Chi square) statistic to determine if the relative class frequencies of adjacent intervals are distinctly different, or if they are similar enough to just merge them into a single interval. It consists of an initialization step and a bottom-up merging process, where intervals are continuously merged until a termination condition is met, which is determined by a significance level(set manually). Post pruning is performed after ChiMerge method. The approach seeks the optimal tree that minimizes the expected error rate on the testing data.

The two datasets used in the project for comparing two algorithms are both from UCI. The first data set is the Iris data set, which consists of 3 classes of 50 instances each, and each instance has 4 attributes. The second data set is the Occupancy Detection data set. The dataset consists of 20560 instances. Each instance has 6 attributes, and the response is a binary data representing whether the room is occupied. The performance of each algorithm is evaluated based on several factors, including its accuracy on testing data, its runtime and space complexity and the compactness of decision tree created.

The project addresses the issue of converting quantitative data to qualitative data. It compares two common techniques for discretization process, and evaluates their respective pros and cons. It offers following contributions:

- After feeding data sets into the discretization methods, the results produce a concise summarization for numeric attributes in the training data.
- Proper discretization of numeric values from training data is an essential step. It produces proper data which could be fed into machine learning algorithms and solve further problems.
- As the classification algorithms would differ according to the number of intervals created, discretization quality depends on the classification algorithm chosen. While it is not possible to have an optimal discretization with which to compare results, some notion of quality would provide engineers with assistance for evaluation of algorithms.

Related Work

In machine learning, decision tree learning is a very common method for solving decision analysis problems. The method was first formally introduced by Quinlan in 1986 in his paper *Induction of Decision Trees*. It constructs a tree structure from the observed data and makes predictions based on the constructed tree model. Sometimes the decision tree construction algorithm requires the train data to be discrete rather than continuous values, this leads to the invention of a class of data pre-processing algorithms called discretization algorithms. These discretization algorithms convert continuous data values to discrete values, based on different criteria.

The simplest and most intuitive data discretization algorithms are *equal-width-intervals* and *equal-frequency-intervals* algorithms. The *equal-width-intervals* algorithm is to divide the sorted data rows between the minimum and maximum values into N intervals of equal size (N being a user-supplied parameter). If A and B are the low and high values, respectively, then the intervals will have width $W = (B - A)/N$ and the interval boundaries will be at $A + W, A + 2W, \ldots, A + (N - 1)W$. And the *equal-frequency-intervals* algorithm is to evenly divide the train data into N intervals, if $N = 10$ then the data will be split into 10 intervals of equal size. Since these types of discretization algorithms do not make use of class membership information during the discretization process, they are categorized as unsupervised discretization algorithms.

In 1992, the *ChiMerge* data discretization algorithm was first introduced by Kerber in his paper *ChiMerge: Discretization of Numeric Attributes*. In contrast to unsupervised discretization algorithms, Kerber's ChiMerge discretization algorithm uses the Chi-square statistic to determine if the relative class frequencies of adjacent intervals are distinctly different, and thus is considered to be a supervised discretization algorithm. This formed the foundation for all Chi-square based data discretization algorithms. In 1995, Liu and Setiono proposed the Chi2 data discretization algorithm in their paper *Chi2: Feature Selection and Discretization of Numeric Attributes*. And several other Chi-square based data discretization algorithms were published after that.

On the other hand, under the same category, there are entropy based supervised discretization algorithms. An example of entropy-based algorithm is the *D-2* algorithm explored by Catlett in 1991. The algorithm uses several conditions as criteria for stopping the recursive formation of partitions for each attribute: a minimum number of samples in one partition, a maximum number of partitions, and a minimum information gain. Fayyad and Irani (1993) used a recursive entropy minimization heuristic for discretization and coupled this with a Minimum Description Length criterion to control the number of intervals produced over the continuous space. In 1995, Pfahringer introduced a value binding method in his paper *Compression-Based Discretization of Continuous Attributes*. The method uses entropy to select a large number of candidate split-points and employs a best-first search with a Minimum Description Length heuristic to determine a good discretization.

When working with decision trees, the tree can be trained extremely well (100% accuracy) against the train data set, as it can be constructed in a way that it perfectly fits the train data set, this leads to the problem of overfitting. To make the decision tree model more generalized and perform equally well on unseen data, two categories of tree pruning techniques are generally used: pre-pruning and post-pruning techniques. Pre-pruning methods come into effectiveness before the tree is fully constructed, it usually takes an user-supplied parameter as its termination condition, and once the tree reaches the condition it eliminates the construction on that sub-branch. A widely used termination condition is the max-depth condition. In terms of post-pruning methods, the most simple but effective method is the *error complexity tree pruning* algorithm introduced by Cestnik and Bratko in 1991. During the process of pruning, the algorithm testifies the importance of the nodes and assign a cost value to every node. At each iteration it cuts off the node that has less importance and continues performing the cutting operation until the validation accuracy starts going down.

In addition to the discretization and generalization algorithms, the model validation technique used during the training process also has a significant influence on predicting the performance of the trained model. Cross validation, one of the most popular and simple model validation techniques, was invented by Mosteller and Tukey in 1968 in his paper *Data analysis, including statistics*. It takes use of the limited data in a smart way, estimate the performance (usually accuracy) of a model on unseen data. It takes a parameter K (user supplied) and evenly distributes the data into K even sets, then uses these equal-size sets to simulate train and future data. In this way, researches can use the train data sets to train the model, and validate its expected future performance by testing it on the unseen data sets.

Scholars and researchers have been comparing these two categories of supervised discretization algorithms for years. In 1995, Dougherty, Kohavi, and Sahami published a comparison paper titled *Supervised and Unsupervised Discretization of Continuous Features*. The paper provides a fundemental and detailed comparison on all types of data discretization algorithms. And in 2006, Kotsiantis and Kanellopoulos formed a survey which has more focus on comparing Chi-square based and entropy based discretization algorithms. In 2011, Dash, Rajib, and Dash also formed a paper which compares supervised and unsupervised discretization techniques. More recently in 2016, Grzymala-Busse and Mroczek proposed a detailed comparison on four entropy based discretization algorithms in their paper *A Comparison of Four Approaches to Discretization Based on Entropy*.

Data Set

In this project, two different data sets are used to evaluate the performance of two discretization methods.

The first dataset is the Iris data set, consisting of 3 classes of 50 instances each, where each class refers to a type of iris plant, including Iris Setosa, Iris Versicolour and Iris Virginica. Each instance has 4 attributes, namely the sepal length, the sepal width, the petal length and the

petal width, representing a single iris plant. The data set can be downloaded at `https://archive.ics.uci.edu/ml/datasets/iris`.

The second data set is the Occupancy Detection data set. The data set consists of 20560 instances. Each instance has 6 attributes: temperature, relative humidity the light, concentration of CO2, and humidity ratio, representing state of a room. The response column is a binary data representing whether the room is occupied. The data set can be downloaded at `https://archive.ics.uci.edu/ml/datasets/Occupancy+Detection+`.

Both data sets have all their attributes continuously typed. The Iris data set is a small, balanced and multi-class data set while the Occupancy data set is a large, unbalanced and binary set. Hence, the performance is tested on two very different data sets. If the results are different, then we can draw a conclusion that one algorithm is better in one case, and the other is better on another case.

In this project, the ID3 algorithm is used to construct the decision tree. Since the attributes in both data sets are continuous, we need to discretize the data first by using either Binary Discretization or the ChiMerge algorithm. We then compare the performance of ID3 + Binary Discretization vs ID3 + ChiMerge.

Evaluation

ID3

The ID3 (Iterative Dichotomiser 3) algorithm is a decision tree learning algorithm based on information entropy. It is proposed by Quinlan in 1986 and its predecessor is the CLS algorithm. The idea is as follow:

Let S be a collection of data samples. Assume S has k distinct classes $c_1, c_2, ..., c_k$. Let A_i be a categorical attribute of S, taking the values $a_{i1}, a_{i2}, ..., a_{in}$.

The information entropy of S is defined as:

$$Ent(S) = -\sum_{i=1}^{k} p(c_i, S) \log p(c_i, S)$$

where $p(c_j, S) = \dfrac{\text{number of instances in S having class } c_j}{\text{number of instances in S}}$

Define $S_{a_{ij}} = \{x \in S, A_i(x) = a_{ij}\}$, a subset of S by selecting only instances having their attribute A_i equals to a_{ij}.

When constructing a decision tree, branching S according to attribute A_i is equivalent as partitioning S into n mutually exclusive subsets $S_{a_{i1}}, S_{a_{i2}}, ..., S_{a_{in}}$. The expected entropy after branching is:

$$Ent_{after}(S, A_i) = \sum_{j=1}^{n} \frac{|S_{a_{ij}}|}{|S|} Ent(S_{a_{ij}})$$

The expected information gain by branching S according to A_i is:

$$IG(S, A_i) = Ent(S) - Ent_{after}(S, A_i)$$

The ID3 algorithm is as follow:

Algorithm 1 ID3 Algorithm(S)

1: if all instances have the same class c_i, return a leaf node with decision c_i
2: if S has no attributes, return a leaf node with the majority class in S
3: if $|S| = 0$, return a leaf node with the majority class in the parent data
4: *else*:
5: choose attribute A_i that has the highest information gain
6: **for** a_{ij} in A_i **do**
7: find $S_{a_{ij}}$
8: $S_{a_{ij}}*$:= remove attribute A_i from $S_{a_{ij}}$
9: add subtree $ID3(S_{a_{ij}}*)$ to children

However, some attributes may not be categorical when constructing the decision tree. To deal with continuous attributes, it's required to apply a discretization method first to convert continuous attributes into categories, and then apply ID3 algorithm.

Binary Discretization

A simple approach to deal with continuous data is to use the Binary Discretization algorithm.

Initially, select a continuously valued attribute A_i, and sort the instances according to feature A_i. Then, find all the *candidate cut points* associated with the attribute A_i, where a *candidate cut point* is defined as the midway between two different sorted values of attribute A_i. For example, suppose $a_{i1}, a_{i2}, ..., a_{in}$ are all the possibles values that exist in the data set S, and suppose $a_{i1} < a_{i2} < ... < a_{in}$, then the *candidate cut point* t_i is equal to $\frac{a_{i2}-a_{i1}}{2}$. Hence there are $n-1$ *candidate cut points* for the attribute A_i.

Then, evaluate each *candidate cut point*. Let $S_{A_i:[t_{j-1},t_j]}$ be the set of instances that have their attribute A_i in the range of $[t_{j-1}, t_j]$, a *candidate cut point* t_i is considered valid if $\exists x \in S_{A_i[t_{j-1},t_j]}, y \in S_{A_i[t_j,t_{j+1}]}$ such that the class of x differs from y. The valid *candidate cut point* is called a *cut point*.

To apply the ID3 algorithm with Binary Discretization, set each tuple of (*cut point*, feature) as a separate feature. Hence, a (original) feature can be used multiple times with different *cut points*.

ChiMerge

An alternative method to discretize the continuous attributes is by using a supervised, bottom-up approach called the ChiMerge algorithm.

The ChiMerge algorithm is based on the idea of performing Pearson's Chi Square Test on adjacent intervals of instances to determine whether they are distinctly different. The test statistics can be computed using the following formula:

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

where:

$m = 2$ (number of intervals being compared)

$k =$ number of classes in S

$O_{ij} =$ number of instances from the i-th interval having class c_j

$E_{ij} = \frac{(\sum_{r=1}^{m} O_{rj} - \sum_{s=1}^{k} O_{is})^2}{\sum_{r=1}^{m} \sum_{s=1}^{k} O_{rs}}$ (the expected frequency)

Note that χ^2 follows the Chi-square distribution with degree of freedom $k - 1$.

The ChiMerge algorithm consists of two parts: an interval initialization step and an interval merging step.

Interval initialization step:

1. use the same approach as in Binary Discretization to split the instances into several sorted intervals.
2. define a χ^2-$threshold$

Interval merging step:

1. compute the χ^2 for every two adjacent intervals.
2. choose the interval pair having the lowest χ^2, if it is less than χ^2-$threshold$, then we merge the two intervals.
3. repeat step (1) until all χ^2's are greater or equal to χ^2-$threshold$

After applying ChiMerge algorithm, each attribute will be divided into several intervals where each interval is treated as a different category. Hence the ID3 algorithm can be applied as if it is dealing with categorical data.

Pruning

Different pruning methods are used for the two types of discretization method.

For the Binary Discretization, a pre-pruning method is used by limiting the depths of decision tree during the generation process. For example, if the maximum depth is set to α, then all non-leaf nodes that is at level α will be converted into a leaf node by taking the majority of classes as decision. The parameter α can be optimized using cross validation on the dataset.

For ChiMerge algorithm, a post-pruning method called Error-Complexity Pruning is used. It is an iterative algorithm that keeps removing the "unimportant" nodes until the accuracy can not be improved further. The reason this algorithm is chosen instead of the maximum depth pre-pruning is because the decision tree generated from ID3 + ChiMerge has a smaller depth (depth equal to number of features). Hence, removing one level is equivalent to removing a feature completely from the data set, which is not a reasonable approach. To run the Error-Complexity Pruning algorithm (explained below), split the data into 3 sets: the training set, the validation set and the test set. The training set is used for generating the decision tree. The validation set is used for pruning the decision to avoid overfitting. The test set is used to measure the final accuracy of the decision tree classifier. 10:3:2 is recommended ratio for the 3 sets.

Error Complexity Pruning

Let $R(t)$ be a function to calculate the error cost of a node t:

$$R(t) = r(t) \times p(t)$$

where:

$r(t) = \frac{\textit{number of misclassified instance in node t}}{\textit{total number of instance in node t}}$ is the miss classification rate

$p(t) = \frac{\textit{number of instance in node t}}{\textit{total number of example in the tree}}$ is the probability a node t being touched

Let $R(T)$ be the error cost of a tree $R(T)$. Then,

$$R(T) = \sum_{i \in f(T)} R(i)$$

where $f(T)$ is the set of leaves of tree T

Denote T_t the subtree in T rooted at node t, and $T - T_t$ the subtree by removing T_t from T.

Then the function we want to optimize

$$R_\alpha(T) = R(T) + \alpha|f(T)|$$

Then the change in error by removing subtree T_t from T is:

$$R_\alpha(T - T_t) - R_\alpha(T) = R(t) - R(T) + \alpha(1 - |f(T)|)$$

Define $\alpha'(t) = \frac{R(t) - R(T_t)}{|f(t)| - 1}$ as the error complexity, then:

- $\alpha' > \alpha$ if the change in error is positive
- $\alpha' = \alpha$ if the change in error is null
- $\alpha' < \alpha$ if the change in error is negative

The Error-Complexity Pruning algorithm consists of following step:

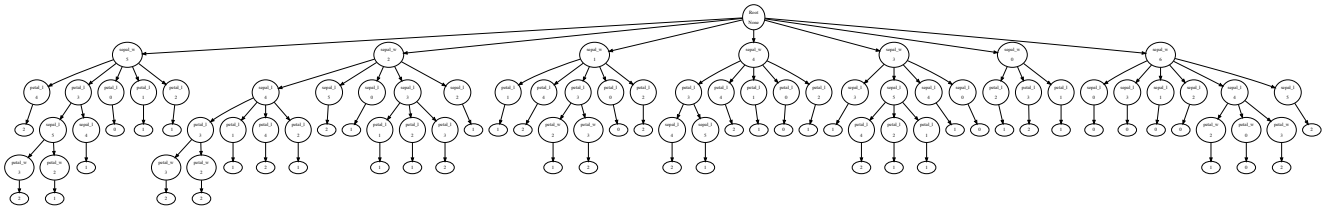
1. compute α' for each node
2. try to prune node with minimum α'
3. if the pruned tree has better accuracy, repeat the step above

Comparison

To compare the algorithms, first compute and compare the accuracies of resulting decision trees generated by ID3 + Binary Discretization and ID3 + ChiMerge at full depth on both datasets. For the iris data set, a $\chi^2_{\textit{threashold}}$ of 0.75 since it is a large data set, then merge as many of the nodes as possible. For the occupancy data set, only 150 instances are observed, hence a larger $\chi^2_{\textit{threashold}}$ of 0.9 is used. The results from running the algorithm at full depth are denoted as "Binary Full" and "ChiMerge Full" in the Accuracy Table of the Results section.

Then, pruning is added to both two algorithms. For ID3 + Binary Descretization, use the maximum depth pre-pruning method with a 5-fold cross validation to find the best maximum depth α. The result is denoted as "Binary Pruned" in Accuracy Table of the Results section. For ID3 + ChiMerge,

Figure 1: Iris Chimerge Tree



the data is separated into 3 sets instead of 2, as explained in the Pruning subsection, and the test set is used to measure the accuracy. The result is denoted as "ChiMerge Pruned" in Accuracy Table of the Results section.

In addition, the shape of the generated decision tree is also compared, by measuring the depth and number of nodes as criterion.

Finally, the relative time efficiency is measured unofficially by running the manually all the algorithms in a same machine.

Results

Accuracy

Table 1: Test Accuracies

Accuracy	Occupancy	Iris
Binary Full	0.992120623	0.77
Binary Pruned	0.992169261	0.84
Chimerge Full	0.991976659	0.9
Chimerge Pruned	0.993920233	0.966666667

When training small data set, pruned tree largely helps increase the testing accuracy (see Table 1). Because pruning prevents overfitting on the training data, which ultimately improves the performance of prediction. Chimerge method has much higher testing accuracy than that of Binary Discretization method when the data set is small. However, when the size of training data gets large, there's no obvious advantage between two methods, and their prediction accuracies are relatively similar.

Shape

Table 2: Depth

Depth	Occupancy	Iris
Binary Full	20	10
Binary Pruned	11	4
Chimerge Full	5	4
Chimerge Pruned	5	4

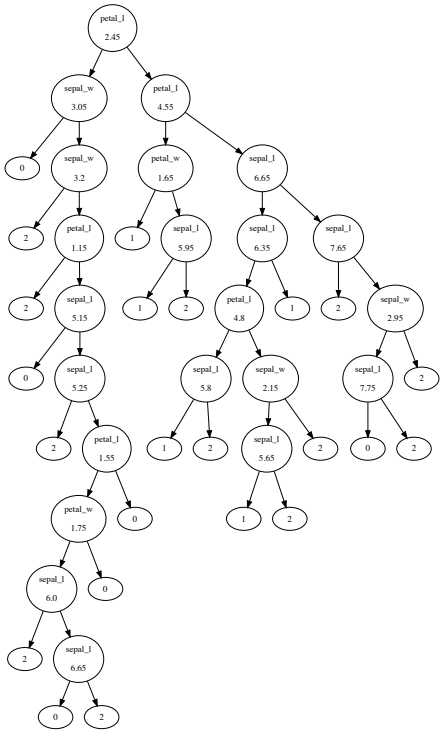
Table 3: Number of Nodes

Number of Nodes	Occupancy	Iris
Binary Full	341	45
Binary Pruned	201	13
Chimerge Full	8552	117
Chimerge Pruned	8418	111

Comparing depth of trees generated by Binary Discretization and Chimerge methods, the tree built by Binary Discretization method is smaller with larger height (see Table 2 and Table 3). Meanwhile, Chimerge method generates a much bigger tree with constant height, that equals to number of features in the data.

The comparison is conducted on two algorithms, targeting on the total running time, including time to preprocess training data, build tree and compute accuracy. If the data set is small, the difference between two algorithms is insignificant. However, when the training data is large, Chimerge performs much slower than Binary Discretization method.

Figure 2: Iris Binary Tree



Conclusion

In summary, when the training data is small, Chimerge with pruning is preferred. However, Binary Discretization with pruning method is preferred if the training data is large, for the purpose of time efficiency.

References

[Catlett] Catlett, J. 1991. On changing continuous attributes into ordered discrete attributes. In Kodratoff, Y., ed., *Machine Learning — EWSL-91*, 164–178. Berlin, Heidelberg: Springer Berlin Heidelberg.

[Cestnik and Bratko] Cestnik, B., and Bratko, I. 1991. On estimating probabilities in tree pruning. In *Proceedings of the European Working Session on Machine Learning*, EWSL '91, 138–150. London, UK, UK: Springer-Verlag.

[Dash, Rajib, and Dash] Dash, R.; Rajib, P.; and Dash, R. 2011. Comparative analysis of supervised and unsupervised discretization techniques. *Int. J. Adv. Sci. Technol.* 2.

[Dougherty, Kohavi, and Sahami] Dougherty, J.; Kohavi, R.; and Sahami, M. 1995. Supervised and unsupervised discretization of continuous features. In *ICML*.

[Fayyad and Irani] Fayyad, U. M., and Irani, K. B. 1993. Multi-interval discretization of continuous-valued attributes for classification learning. In *IJCAI*.

[Grzymala-Busse and Mroczek] Grzymala-Busse, J., and Mroczek, T. 2016. A comparison of four approaches to discretization based on entropy. *Entropy* 18:69.

[Kerber] Kerber, R. 1992. Chimerge: Discretization of numeric attributes. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, AAAI'92, 123–128. AAAI Press.

[Kotsiantis and Kanellopoulos] Kotsiantis, S. B., and Kanellopoulos, D. 2006. Discretization techniques: A recent survey.

[Liu and Setiono] Liu, H., and Setiono, R. 1995. Chi2: Feature selection and discretization of numeric attributes. In *Proceedings of the Seventh International Conference on Tools with Artificial Intelligence*, TAI '95, 88–. Washington, DC, USA: IEEE Computer Society.

[Mosteller and Tukey] Mosteller, F., and Tukey, J. W. 1968. Data analysis, including statistics. In Lindzey, G., and Aronson, E., eds., *Handbook of Social Psychology, Vol. 2*. Addison-Wesley.

[Pfahringer] Pfahringer, B. 1995. Compression-based discretization of continuous attributes. In *Proceedings of the Twelfth International Conference on International Conference on Machine Learning*, ICML'95, 456–463. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

[Quinlan] Quinlan, J. R. 1986. Induction of decision trees. *Machine Learning* 1(1):81–106.