Feature Selection via Discretization

Huan Liu and Rudy Setiono

***Abstract*— Discretization can turn numeric attributes into discrete ones. Feature selection can eliminate some irrelevant and/or redundant attributes. Chi2 is a simple and general algorithm that uses the χ^2 statistic to discretize numeric attributes repeatedly until some inconsistencies are found in the data. It achieves feature selection via discretization. It can handle *mixed* attributes, work with *multi-class* data, and remove *irrelevant* and *redundant* attributes.**

***Keywords*— discretization, feature selection, pattern classification**

I. Introduction

Feature selection can eliminate some irrelevant and/or redundant attributes. By using relevant features, classification algorithms can in general improve their predictive accuracy, shorten the learning period, and form simpler concepts. There are abundant feature selection algorithms. Some use methods like principle component to compose a smaller number of new features [11,12]; some select a subset of the original attributes [1,5]. This paper considers the latter since it not only has the above virtues, but also serves as an indicator on what kind of data (along those selected features) should be collected. In the latter category of feature selection, the algorithms can be further divided in terms of data types. The two basic types of data are nominal (e.g., attribute *color* may have values of red, green, yellow) and ordinal (e.g., attribute *winning position* can have values of 1, 2, and 3, or attribute *salary* can have 22345.00, 46543.89, etc. as its values). Many feature selection algorithms [1,3,5,10] are shown to work effectively on discrete data or even more strictly, on binary data (and/or binary class value). In order to deal with numeric attributes, a common practice for those algorithms is to discretize the data before conducting feature selection. This paper provides a way to select features directly from numeric attributes while discretizing them. Numeric data are very common in real world problems. However, many classification algorithms require that the training data contain only discrete attributes, and some would work better on discretized or binarized data [2,4]. If those numeric data can be *automatically* transformed into discrete ones, these classification algorithms would be readily at our disposal. Chi2 is our effort towards this goal: discretize the numeric attributes as well as select features among them.

The problem to attack is: given data sets with numeric attributes (some of which are irrelevant and/or redundant and the range of each numeric attribute could be very wide), find an algorithm that can automatically discretize the numeric attributes as well as remove irrelevant/redundant ones.

This work is closely related to Kerber's ChiMerge [4]which discretizes numeric attributes based on the χ^2 statistic. ChiMerge consists of an initialization step and a bottom-up merging process, where intervals are continuously merged until a termination condition, which is determined by a significance level α (set manually), is met. It is an improvement from the most obvious simple methods such as *equal-width-intervals*, which divides the number line between the minimum and maximum values into N intervals of equal size; or *equal-frequency-intervals*, in which the interval boundaries are chosen so that each interval contains approximately the same number of training examples. Instead of defining a width or frequency threshold (which is not easy until scrutinizing each attribute and knowing what it is), ChiMerge requires α to be specified (ideally one α for each attribute). Nevertheless, too big or too small an α will over- or under-discretize an attribute. An extreme example of under-discretization is the continuous attribute itself. Over-discretization will introduce many inconsistencies[1] nonexistent before, thus change the characteristics of the data. In short, it is not easy to find a proper α for ChiMerge. It is thereby ideal to let the data determine what value α should take. This leads to our Chi2 algorithm. Naturally, if we let the discretization continue as long as no more inconsistencies generated than in the original data, each attribute is discretized to the maximum and some attributes may be discretized into one interval. Hence, these attributes can be removed without affecting the discriminating power of the original data.

In the following, we describe the Chi2 algorithm, the experiments, and its various aspects in turn.

II. Chi2 Algorithm

The Chi2 algorithm (summarized below) applies the χ^2 statistic which conducts a significance test on the relationship between the values of an attribute and the categories. It consists of two phases. In the first phase, it begins with a large significance level (α), e.g., 0.5, for all numeric attributes to be discretized. Each attribute is sorted according to its values. Then, for each attribute, the following is performed: 1. calculate the χ^2 value as in equation (1)

Department of Information Systems and Computer Science, National University of Singapore, Kent Ridge, Singapore 119260, {liuh,rudys}@iscs.nus.sg

[1]By inconsistency we mean that two patterns match, but belong to different categories.

for every pair of adjacent intervals (at the beginning, the number of intervals equals the number of distinct values of an attribute); 2. merge the pair of adjacent intervals with the lowest χ^2 value being the critical value. Merging continues until all pairs of intervals have χ^2 values exceeding the parameter determined by α (if initially it is 0.5, its corresponding χ^2 value is 0.455 if the degree of freedom is 1, more below). The above process is repeated with a decreased α until the discretized data's inconsistency rate exceeds δ. Phase 1 is, as a matter of fact, a generalized version of ChiMerge of Kerber [4]. Instead of specifying a χ^2 threshold, Phase 1 of Chi2 wraps up ChiMerge with a loop that automatically increments the χ^2 threshold (or equivalently decreases α). A consistency checking is also introduced as a stopping criterion to make sure that the discretized data set accurately represents the original one. With these two new features, Chi2 automatically determines a proper χ^2 threshold that keeps the fidelity of the original data.

Phase 2 is a finer process of Phase 1. Starting with $\alpha 0$ determined in Phase 1, each attribute i is associated with a sigLvl[i], and takes turns for merging. Consistency checking is conducted after each attribute's merging. If the inconsistency rate is not exceeded, sigLvl[i] is decreased for attribute i's next round of merging; otherwise attribute i will not be involved in further merging. This process is continued until no attribute's values can be merged. The round-robin discretization achieves two objectives: (1) removal of irrelevant/redundant attributes; and (2) better coordination among the discretized attributes.

Chi2 Algorithm:

```
Phase 1:  (att - attribute)
  set α = .5;
  do while (InConCheck(data) < δ) {
     for each numeric att {
       Sort(att, data); /* sort data on att */
       chi-sq-init(att, data); /* refresh data */
         do {
           chi-sq-calculation(att,data)
        } while (Merge(data))      }
     α0 = α;
     α = decreSigLevel(α);  }
Phase 2:
  set all sigLvl[i] = α0 for att i;
  do until no-att-can-be-merged {
    for each mergeable att i {
      Sort(att, data); /* sort data on att */
      chi-sq-init(att, data); /* refresh data */
      do {
        chi-sq-calculation(att,data)
     } while (Merge(data))
      if (InConCheck(data) < δ)
        sigLvl[i] = decreSigLevel(sigLvl[i]);
      else  att i is not mergeable;  } }
```

Function InConCheck() returns an inconsistency rate found in the discretized data. Function Merge() returns true or false depending on whether the concerned attribute is merged or not. Function decreSigLevel() decreases the significance level by one level according to the implemented χ^2 table. Function chi-sq-init() prepares for the χ^2 computation. The formula for computing the χ^2 value is:

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

where:

k = number of classes,

A_{ij} = number of patterns in the ith interval, jth class,

R_i = number of patterns in the ith interval = $\sum_{j=1}^{k} A_{ij}$,

C_j = number of patterns in the jth class = $\sum_{i=1}^{2} A_{ij}$,

N = total number of patterns = $\sum_{i=1}^{2} R_i$,

E_{ij} = expected frequency of $A_{ij} = R_i * C_j / N$. If either R_i or C_j is 0, E_{ij} is set to 0.1. The degree of freedom of the χ^2 statistic is one less the number of classes.

The inconsistency rate of a dataset is calculated as follows: (1) two instances are considered inconsistent if they match except for their class labels; (2) for all the matching instances (without considering their class labels), the inconsistency count is the number of the instances minus the largest number of instances of class labels: for example, there are n matching instances, among them, c_1 instances belong to label$_1$, c_2 to label$_2$, and c_3 to label$_3$ where $c_1 + c_2 + c_3 = n$. If c_3 is the largest among the three, the inconsistency count is $(n - c_3)$; (3) the inconsistency rate is the sum of all the inconsistency counts divided by the total number of instances.

The purpose of the two-phase implementation of Chi2 is twofold: (1) a direct comparison with ChiMerge. Since in a sense, Phase 1 of Chi2 is an automated version of ChiMerge; and (2) consideration of computational efficiency (to be discussed in Section IV). At the end of Phase 2, if an attribute is merged to only one value, it simply means that this attribute is not needed in representing the original data set. As a result, when discretization ends, feature selection is accomplished.

III. Experiments

Two sets of experiments are conducted. In the first set of experiments, real-world data is used and the evaluation is done indirectly, i.e. through/against a classifier. We want to establish that (1) Chi2 helps improve predictive accuracy; and (2) Chi2 properly and effectively discretizes data as well as eliminates some irrelevant/redundant attributes (this explains why the predictive accuracy of a classifier is improved.) C4.5 [8] is used for these purposes. The reasons for our choice are (1) C4.5 works well for many problems and is well known, thus requiring no further description; and (2) C4.5 selects relevant features by itself in tree branching so it can be used as a benchmark, as in [1,5,9], to verify the effects of Chi2. In the second set of experiments, we directly examine Chi2's ability of discretizing and feature selection by introducing synthetic data sets and adding noisy attributes to one real-world data set. Through experiments on these controlled data sets, we can better understand how effective Chi2 is.

A. Real-world data

Three data sets used in experiments are Iris, Wisconsin Breast Cancer and Heart Disease[2]. They have different types of attributes. The Iris data are of continuous attributes, the breast cancer data of ordinal discrete ones, and the heart disease data of mixed attributes (numeric and discrete). The three data sets are described below.

Iris data contains 50 patterns each of the classes *Iris setosa, Iris versicolor*, and *Iris virginica*. Each pattern is described using four numeric attributes: *sepal-length, sepal-width, petal-length,* and *petal-width*. The originally odd numbered data are selected for training (75 patterns), the rest for testing (75 patterns).

Breast cancer data contains 699 samples of breast fine-needle aspirates collected at the University of Wisconsin Hospital. There are 9 discrete attributes valued on a scale from 1 to 10. The class value is either benign or malignant. The data set is split randomly into two sets, 350 patterns for training and 349 for testing.

Heart disease data contains medical cases of heart diseases. It contains numerically-valued features; there are 8 nominally-valued and 5 numerically-valued attributes. Two class values are: healthy and diseased heart. Removing patterns with missing attribute values, we use 299 patterns, one third of which are randomly chosen for testing, and the rest are for training.

B. Controlled data

Three extra data sets are designed to test if various noisy attributes can be removed. The first two are synthetic, the third one is the Iris data added with noisy attributes.

The synthetic data, S1, consists of 600 items and is described by four attributes among which only one attribute determines each item's class label. The values, v_1 of attribute A_1 are generated from a uniform distribution between the lower bound (L = 0) and the upper bound (U = 75), each item's class label is determined as follows: $v_1 < 25 \rightarrow$ class 1, $25 \leq v_1 < 50 \rightarrow$ class 2, $50 \leq v_1 < 75 \rightarrow$ class 3. Then we add *irrelevant* attributes[3] A_2, A_3, and A_4. The values of A_2 are generated from a normal distribution with mean $\mu = U/2$ (i.e. 37.5) and standard deviation $\sigma = \mu/3$. The values of A_3 are generated from two normal distributions with $\mu_1 = U/3$ (i.e. 25), $\mu_2 = 2 * U/3$ (i.e. 50) and $\sigma_1 = \mu_1/3, \sigma_2 = \mu_2/3$ respectively, 300 values from each distribution. The values of A_4 are generated from a uniform distribution between L and U.

The synthetic data set, S2, contains both *irrelevant* and *redundant* attributes, and is made up of 600 items. Attributes A_1 and A_2 are similarly constructed as A_1 and A_2 in S1 with A_2 being irrelevant. A value of attribute A_3 is obtained by multiplying the corresponding value of A_1 by a constant factor, 3. Thus either A_1 or A_3 alone can determine an item's class label, i.e. one of them is redundant.

The third data set, S3, is a modified version of Iris data. Four noisy attributes A_5, A_6, A_7 and A_8 are added to the Iris training data corresponding to the four original attributes. The values of each noisy attribute are determined by a normal distribution with $\mu = ave$ and $\sigma = (max - min)/6$, where ave, max and min are the average, maximum and minimum values of the original attribute. Now there are total eight attributes, still 75 items.

C. Empirical results on real-world data

First we show that after discretization, the number of attributes decreases for the three data sets (in Table 1). For the Iris data, the number of attributes is reduced from 4 to 2 (petal length and petal width), each has three values. For the breast cancer data, 3 attributes are removed from the original 9 attributes. The remaining 6 attributes have 2, 3, 3, 4, 2, and 2 discrete values respectively. For the heart disease data, the discrete attributes are left out in discretization and feature selection although they are used for consistency checking. Among the 5 continuous attributes (1, 4, 5, 8 and 10), only 2 attributes (5 and 8) remain as suggested by Chi2, having 7 and 3 discrete values respectively. For the breast cancer and heart disease data, δ is set as 0, for the iris data, δ is 5%.

Table 1: Change in Number of Attributes

	Iris	Heart	Breast
Before	4	13	9
After	2	10	6

Second, we run C4.5 on both the original data sets and the discretized ones. C4.5 is run using its default setting. Chi2 discretizes the training data and generates a mapping table, based on which the testing data are discretized.

Shown in Tables 2 and 3 are predictive accuracies and tree sizes of C4.5 for the three data sets. Predictive accuracy improves and tree size drops (by half) for the breast cancer and heart disease data. As for the Iris data, accuracy and tree size remain the same by using two attributes only (with 4 values each). In a way, it shows that C4.5 works pretty well without Chi2 for this small data set.

Table 2: Change in Predictive Accuracy

	Iris	Heart	Breast
Before	94.7%	72.7%	92.6%
After	94.7%	78.8%	94.6%

Table 3: Change in Size of a Decision Tree

	Iris	Heart	Breast
Before	5	43	21
After	5	22	11

D. Empirical results on Controlled data

The purpose of experimenting on the controlled data is to verify how effective Chi2 is in removing noisy attributes through discretization. Therefore, it is only necessary to

[2] They are all obtained from the University of California-Irvine, machine learning repository via anonymous ftp to *ics.uci.edu*.

[3] By which we mean that these attributes are not related to class values.

see if Chi2 can (1) discretize the relevant attribute(s) properly, (2) remove the irrelevant attributes, and (3) remove both irrelevant and redundant attributes.

For the synthetic data S1, Chi2 merged A_1 into three discrete values (1, 2 and 3) corresponding to three classes (1,2, and 3); merged the other three attributes A_2, A_3, and A_4 into one value at the end of Phase 1. That is, only A_1 stays, and the noisy attributes are removed.

For the synthetic data S2, Phase 1 of Chi2 merged A_1 and A_3 into three discrete values (1, 2 and 3), A_2 (irrelevant attribute) into one value. It is Phase 2 of Chi2 that merged A_3 (redundant one) into one value. At the end, both irrelevant and redundant attributes were removed.

For the modified Iris data S3, Phase 1 of Chi2 merged A_1, A_2, A_3 and A_4 into 3, 2, 3 and 3 discrete values, and discretized attributes A_5, A_6, A_7 and A_8 into one value. Recall that the last four attributes are added irrelevant attributes. In Phase 2, attributes A_1 and A_2 were further merged into one value only. Attributes A_3 and A_4 remained with 3 discrete values respectively identical to those found in the experiment with the original data.

This set of controlled experiments has shown that Chi2 effectively discretizes numeric attributes and removes irrelevant and redundant attributes. Redundant attributes are removed in Phase 2.

IV. Discussions and Conclusion

Since each *Merge* in the Chi2 algorithm only reduces the number of intervals by one, in the worst case (there are n different values and all of them can be merged into one value), the innermost loop requires $n-1$ times of calling the χ^2 function (n is the number of patterns in the training data[4]), but each *Sort* needs $O(n \log n)$. So for m attributes, the reimplemented ChiMerge requires $O(mn \log n)$. Consider the worst case, checking data *Consistent* or not (refer to the Chi2 algorithm) takes $O(mn)$. The outermost loop is determined by the number of incremental steps, K, of the χ^2 value. Hence, the computational complexity of Phase 1 is $O(Km(n + n \log n))$, i.e. $O(Kmn \log n)$. Similar complexity can be obtained for Phase 2. The complexity result gives a guideline on how long it would take to run Chi2 for a given data set.

The two-phase implementation is due to the concern of efficiency. Phase 1 is mainly designed to improve efficiency, specially when m is large. Due to the consistency checking which takes $O(n)$, the saving can be as much as $(m-1)n$ for each outermost loop by implementing the two phases.

Chi2 can only be used to discretize data and select features for supervised learning tasks since class information is vital in the χ^2 statistic. Also, Chi2 works on ordinal attributes only. If there are mixed (nominal and ordinal) attributes, Chi2 can be specified to operate only on the ordinal attributes for discretization and feature selection. Chi2 is only attempting to discover first-order (single attribute - class) correlations, thus might not perform correctly when there is a second-order correlation without a corresponding first-order correlation. Some feature weighting methods as in [5,6] can be helpful when higher order correlation in the data has to be considered.

Another issue is how to determine an initial α. Too large an α will make Chi2 run longer. However, the final α values for numeric attributes will remain the same for different initial α values if α is not set too small (0.05 for instance) at the beginning. In addition to α, the only threshold required is the tolerable rate of inconsistency, δ. Its default value is 0 assuming that the data set is consistent, and can be reset to any value between 0 and 1. A reasonable approximation is the rate of inconsistency found in the training data, which is not difficult to compute.

Chi2 is a simple and general algorithm that can automatically select a proper critical value for the χ^2 test, determine the intervals of a numeric attribute, as well as select features by removing irrelevant and redundant attributes according to the characteristics of the data. By using the inconsistency criterion, it guarantees that the fidelity of the training data can remain after Chi2 is applied. The empirical results on both the real-world data and controlled data have shown that Chi2 is a useful and reliable tool for discretization and feature selection of numeric attributes.

[4] n is also the upper limit of the number of values an attribute can take in the sample data.

References

[1] H. Almuallim and T.G. Dietterich. Learning boolean concepts in the presence of many irrelevant features. *Artificial Intelligence*, 69(1-2):279–305, November 1994.

[2] J. Catlett. On changing continuous attributes into ordered discrete attributes. In *European Working Session on Learning*, 1991.

[3] U.M. Fayyad and K.B. Irani. The attribute selection problem in decision tree generation. In *AAAI-92, Proceedings Ninth National Conference on Artificial Intelligence*, pages 104–110. AAAI Press/The MIT Press, 1992.

[4] R. Kerber. ChiMerge: Discretization of numeric attributes. In *AAAI-92, Proceedings Ninth National Conference on Artificial Intelligence*, pages 123–128. AAAI Press/The MIT Press, 1992.

[5] K. Kira and L.A. Rendell. The feature selection problem: Traditional methods and a new algorithm. In *AAAI-92, Proceedings Ninth National Conference on Artificial Intelligence*, pages 129–134. AAAI Press/The MIT Press, 1992.

[6] H. Liu and W.X. Wen. Concept learning through feature selection. In *Proceedings of First Australian and New Zealand Conference on Intelligent Information Systems*, 1993.

[7] J. Murdoch and J.A. Barnes. Statistical tables for science, engineering, management and business studies. THE MACMILLAN PRESS LTD, 1986.

[8] J.R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[9] H. Ragavan and L. Rendell. Lookahead feature construction for learning hard concepts. In *Machine Learning: Proceedings of the Seventh International Conference*, pages 252–259. Morgan Kaufmann Pub. San Mateo, California, 1993.

[10] H. Liu and R. Setiono. A probabilistic approach to feature selection - A filter solution. In *Proceedings of the 13th International Conference on Machine Learning*, pages 319–327, 1996.

[11] I. Sethi, and G. Savarajudu. Hierarchical classifier design using mutual information. In *IEEE Trans. PAMI*, Vol 4, pages 441–445, July 1982.

[12] N. Wyse, R. Dubes, and A.K. Jain. A critical evaluation of intrinsic dimensionality algorithms. In E.S. Gelsema and Kanal L.N., editors, *Pattern Recognition in Practice*, pages 415–425. Morgan Kaufmann Publishers, Inc., 1980.