CViz: An Interactive Visualization System for Rule Induction

Jianchao Han, Aijun An, Nick Cercone

Department of Computer Science, University of Waterloo
Waterloo, Ontario, N2L 3G1, Canada
Email: {j2han, aan, ncercone}@math.uwaterloo.ca

Abstract. We introduce an interactive visualization system, CViz, for rule induction. The process of learning classification rules is visualized, which consists of five components: preparing and visualizing the original data, cleaning the original data, discretizing numerical attributes, learning classification rules, and visualizing the discovered rules. The CViz system is presented and each component is discussed. Three approaches for discretizing numerical attributes, including *equal-length, equal-depth*, and *entropy-based approaches*, are provided. The algorithm ELEM2 for learning classification rules is introduced, and the approaches to visualizing discretized data and classification rules are proposed. CViz could be easily adapted to visualize the rule induction process of other rule-based learning systems. Our experimental results on the IRIS data, Monks data, and artificial data show that the CViz system is useful and helpful for visualizing and understanding the learning process of classification rules.

Keywords: Interactive visualization, knowledge discovery, machine learning, rule induction.

1 Introduction

Interactive visualization techniques allow to visualize the results of presentations on the fly in different perspectives, and thus help users understand the discovered knowledge better and more easily. This makes the knowledge discovery process straightforward and accessible.

Many techniques and systems for data visualization have been developed and implemented [4, 7, 8, 11]. One common feature of these business systems is their dependence on computer graphics and scientific visualization. Most existing visualization systems lack the abilities to visualize the entire process of knowledge discovery. The complex data is carefully arranged and displayed in a specific visual form, and the knowledge behind the original data is left to the users who must observe and determine the meaning of the pictures. This usually requires a wealth of background knowledge. Silicon Graphics developed a series of visualizers like Map Visualizer, Tree Visualizer, etc. [7] to visualize the knowledge discovered according to different techniques such as decision trees, neural networks, etc. but only the results are displayed. Interactive visual knowledge discovery should provide a user with not only the discovery results but also the entire process in a visual form so that the user can participate in the discovery process and understand the knowledge discovered.

The CViz system stresses the visualization of the classification rule discovery. It consists of five components, including original data visualization on the *parallel coordinates system* [10, 11], interactive data reduction (horizontally and vertically) by removing some attributes and/or attribute values, numerical attribute discretization and visualization with three approaches, rule induction using learning algorithm ELEM2 [2], and rule visualization.

The framework of the CViz system and its components are presented in Section 2. Three approaches to discretizing numerical attributes are discussed in Section 3. In Section 4, the learning algorithm ELEM2 used in the CViz system is introduced. The rules discovered by ELEM2 can be displayed on screen in visual forms, which is the topic of Section 5. In Section 6, the experiment results with artificial data and UCI data are showed. Related work is outlined in Section 7, and Section 8 is the concluding remark.

2 The CViz System

CViz is an interactive knowledge discovery visualization system, which uses interactive visualization techniques to visualize the original data, help the user clean and preprocess the data, and interpret the rules discovered. CViz also conducts numerical attribute discretization and rule induction to learn classification rules based on the training data set. CViz consists of five components, shown in Fig. 1.

In CViz, the original data is visualized based on *parallel coordinates* technique [11]. Suppose the training data are represented as n-ary tuples. n equidistant axes are created to be parallel to one of the screen axes, say Y-axis, and correspond to the attributes. The axes are scaled to the range of the corresponding attributes and normalized, if necessary. Every tuple corresponds to a polyline which intersects each of the axes at the point that corresponds to the value for the attribute [10]. Fig. 5 shows the effects of using parallel coordinates technique to visualize multidimensional data sets. Each coordinate is interpreted further with a list of values for a categorical attribute, while a numeric attribute can be attached with the minimum and maximum, even mean, variance, etc. As the algorithm proceeds, numeric intervals can be added, including the start point and the end point for each interval and the number of total intervals once the numeric attribute is discretized.

The second component is to reduce the original data, if necessary. Once the original data is visualized on the parallel coordinates, the user can get a rough idea about the data distribution so that data cleaning might be done. The data reduction involves two aspects: *horizontal* and *vertical* reduction. An attribute that the user thinks irrelevant to the knowledge discovery can be interactively removed by *deleting-click*, thus the data is horizontally reduced. While an attribute value can be deleted if the distribution of the data tuples passing the

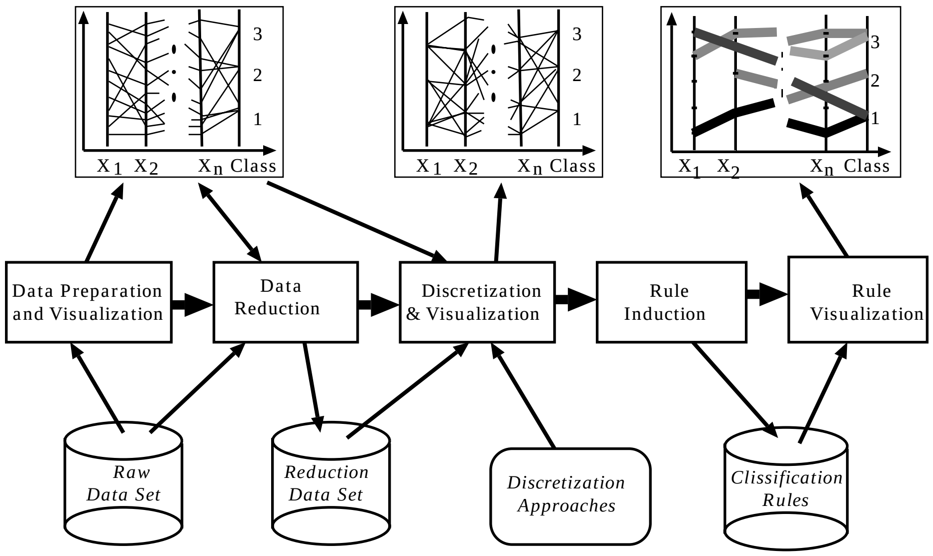

Fig. 1. The CViz System

value is too sparse, thus the data tuples that have the value could be removed so that the data is vertically reduced. The data reduction is highly dependent on the user [8].

The third component is to discretize numerical (continuous) attributes. The CViz system provides three approaches: *equi-length*, *equi-depth*, and *entropy-based* methods [3]. The user can select the method that he/she likes, or select all of them for different learning runs for comparison of the learning results. The discretized attributes are visualized again where an attribute interval corresponds to a point on the attribute coordinate. Fig. 6 shows the discretized parallel coordinates.

Rule induction and visualization are the last two components. The ELEM2 learning algorithm is used to learn classification rules [2], while the final rules are visualized as colored strips (polygons). Each rule is illustrated by a subset of coordinates with corresponding values and/or intervals. The coordinates in the subset are connected together through the values or intervals. The rule accuracy and quality values are used to render the rule strips. Fig. 7 through Fig. 12 illustrate a part of the classification rules obtained in our experiments.

3 Discretizing Numerical Attributes

The CViz system provides three methods to discretize numerical attributes, equi-length, equi-depth, and entropy-based approaches.

The *equi-length approach* partitions the continuous domain into intervals with equal length. For example, if the domain of attribute *age* is $[0, 100]$, then it can be divided into small intervals of length 10, thus we have *age* intervals $[0,10],(10,20],(20,30],\ldots,(90,100]$. This approach is can be easily implemented. Its main drawback is that many useful rules may be missed since the distribution of the data values is not considered.

The second discretization approach used in CViz is called *bin-packing based equi-depth approach*, which is different from existing equi-depth approaches. The domain of the numerical attributes may contain an infinite number of points. To deal with this problem, KID3 employs an *adjustable buckets method* [13], while the approach proposed in [14] is based on the concept of a partial completeness measure. The drawback of these approaches is in time-consuming computation and/or large storage requirements. CViz exploits a simple and direct method, which is described in Fig. 2.

Assume the window size used to visualize the data set is M (width or height) in pixels, and each pixel corresponds to a *bin.* Thus we have M bins, denoted $B[i], i=0,\ldots,M-1$. We map the raw data tuples to the bins in terms of the mapping function. Suppose $B[i]$ contains $T[i]$ tuples, and further, the attribute is to be discretized into N buckets. According to the equi-depth approach, each bucket will contain $d=\sum_{i=0}^{M-1}T[i]/N$ tuples. We first assign $B[0], B[1], \ldots$, to the first bucket until it contains at least d tuples, and then assign the following bins to the second bucket. We can repeat this process until all buckets contain a roughly equal number of tuples.

```
Procedure for bin-packing discretization
    j=0;
    for $(i=0; i<N; i++)$
        $Bucket[i]=0;$
        for $(k=j, K<M, k++)$
            $Bucket[i]+=T[j++];$
            if $Bucket[i]\geq d$
                break;
```

Fig. 2. Bin-packing based Equi-depth Discretization

The storage requirement in this approach is $O(M+N)$, depending on the number of buckets and the size of the visualization window, regardless of the domain of the attributes and the size of the data set. This approach does not need to sort the data and the execution time is linear in the size of the data set. This method, however, may not produce enough buckets, because each bin must be assigned to only one bucket, and cannot be broken up. For instance, if the data concentrates in several bins, then the buckets that contain these bins will contain many more tuples than others. This case could happen especially when the visualization window has a small size.

The third approach uses EDA-DB (Entropy-based Discretization According to Distribution of Boundary points) method [3]. Unlike solely entropy-based discretization methods (such as the method based on the Minimum Description Length Principle [5]), EDA-DB first divides the value range of the attribute into several *big* intervals and then selects in each interval a number of cut-points based on the entropy calculated over the current entire data set. The number of cut-points selected for each interval is determined by estimating the probability distribution of the boundary points over the data set. The maximum number of selected cut-points is determined by the number of class labels and the number of distinct observed values for the continuous attribute.

Let l be the number of distinct observed values for a continuous attribute A, b be the total number of boundary points for A, and k be the number of classes in the data set. Then the discretization of A is described in the procedure illustrated in Fig. 3,

```
Procedure for EDA-DB discretization
1. Calculate: $m = max\{2, k * log_2(l)\}$.
2. Estimate the probability distribution of boundary points:
   (a) Divide the value range of $A$ into $d$ intervals, where $d = max\{1, log_e(l)\}$.
   (b) Calculate the number $b_i$ of boundary points in each interval $iv_i$, where
       $i = 1, 2, \cdots, d$ and $\sum_{i=1}^{d} b_i = b$.
   (c) Estimate the probability of boundary points in each interval $iv_i$ ($i =
       1, 2, \cdots, d$) as $p_i = b_i/b$.
3. Calculate the quota $q_i$ of cut-points for each interval $iv_i$ ($i = 1, 2, \cdots, d$) ac-
   cording to $m$ and the distribution of boundary points: $q_i = p_i * m$.
4. Rank the boundary points in each interval $iv_i$ ($i = 1, 2, \cdots, d$) by increasing
   order of the class information entropy of the partition induced by the boundary
   point. The entropy for each point is calculated globally over the entire data
   set.
5. For each interval $iv_i$ ($i = 1, 2, \cdots, d$), select the first $q_i$ points in the above
   ordered sequence. A total of $m$ cut-points are selected.
```

Fig. 3. EDA-DB discretization of continuous attribute

4 Learning Classification Rules

The CViz system exploits ELEM2 as the approach to discovering knowledge. ELEM2 is a rule induction system that learns classification rules from a set of data [2]. Given a set of training data, ELEM2 sequentially learns a set of rules for each of classes in the data set. To induce rules for a class C, ELEM2 conducts general-to-specific heuristic search over a hypothesis space to generate a disjunctive set of propositional rules. ELEM2 uses a *sequential covering* learning strategy; it reduces the problem of learning a disjunctive set of rules to a sequence of simpler problems, each requiring that a single conjunctive rule be learned that covers a subset of positive examples. The learning of a single conjunctive rule begins by considering the most general rule precondition, (e.g., the empty test that matches every training example,) then greedily searching for an attribute-value pair that are most relevant to the class label C according to the following attribute-value pair evaluation function:

$$SIG_C(av) = P(av)(P(C|av) - P(C))$$

where av is an attribute-value pair and P denotes probability [2]. The selected attribute-value pair is then added to the rule precondition as a conjunct. The process is repeated by greedily adding a second attribute-value pair, and so on, until the hypothesis reaches an acceptable level of performance. In ELEM2, the acceptable level is based on the consistency of the rule: it forms a rule that is as consistent with the training data as possible. Since this *consistent* rule may be a small disjunct that overfits the training data, ELEM2 may *post-prune* the rule after the initial search for this rule is complete.

To post-prune a rule, ELEM2 first computes a rule quality value according to a formula that measures the extent to which a rule R can discriminate between the positive and negative examples of class label C:

$$log \frac{P(R|C)(1 - P(R|\bar{C}))}{P(R|\bar{C})(1 - P(R|C))}.$$

ELEM2 then checks each attribute-value pair in the rule in the reverse order in which they were selected to see if removal of the attribute-value pair will decrease the rule quality value. If not, the attribute-value pair is removed and the procedure checks all the other pairs in the same order again using the new rule quality value resulting from the removal of that attribute-value pair to see whether another attribute-value pair can be removed. This procedure continues until no pair can be removed.

After rules are induced for all the classes, the rules can be used to classify new examples. The classification procedure in ELEM2 considers three possible cases when a new example matches a set of rules.

- *Single match.* The new example satisfies one or more rules of the same class. In this case, the example is classified to the class indicated by the rule(s).
- *Multiple match.* The new example satisfies more than one rule that indicates different classes. In this case, ELEM2 activates a conflict resolution scheme for the best decision.
- *No match.* The new example is not covered by any rule. In this case, the decision score is calculated for each class and the new example is classified into the class with the highest decision score.

5 Rule Visualization

The basic idea of visualizing classification rules in the CViz system is to represent a rule as a strip, called *rule polygon*, which covers the area that connetcs the corresponding attribute values. A classification rule induced by the ELEM2 algorithm is a logical statement that consists of two parts: *condition part* and *decision part*. The *decision* part is simply the class label. The *condition* part is composed of a set of simple relations, each of which corresponds to a specific attribute.

(1) *For categorical attributes :*
$A = a_1, a_2, \ldots,$ *or* a_n, meaning that attribute A has any values of $a_1, a_2, \ldots$, *or* a_n, where $n >= 1$;
$A! = a_1, a_2, \ldots,$ *or* a_n, meaning that attribute A has any values except $a_1, a_2, \ldots$, *and* a_n, where $n >= 1$.
(2) *For continuous attributes :*
$A <= a$, meaning that attribute A has any values between the minimum and a;
$a < A$, meaning that attribute A has any values between a and the maximum;
$a_1 < A <= a_2$, meaning that attribute A has any values between a_1 and a_2.

For example, the following is a classification rule obtained in our experiment with an artificial data set, and is visualized in Fig. 8:

$$(30 < Age <= 60)(1.0 < Score <= 3.5)(Color = red\ or\ yellow) \Rightarrow Class = bad.$$

It means that if numerical attributes *Age* and *Score* have values between 30 and 60 and between 1.0 and 3.5, respectively, and categorical attribute *Color* has value *red* or *yellow*, then the instance will be in class *bad*. The corresponding *rule polygon* consists of two polygons which are enclosed by the points that correspond to value *red* and *yellow* on coordinate *Color*, respectively, the points correspond to 30 and 60 on coordinate *Age*, the points that correspond to 1.5 and 3.5 on coordinate *Score*, and the point corresponding to class *bad* on the decision coordinate.

The *rule polygon* is constructed by a *rule polygon generating* procedure, which is described in Fig. 4.

To distinguish between *positive* conditions and *negative* conditions represented with unequal symbol (!), the positive condition is drawn with backward hatch, while the negative condition is drawn with forward hatch. In addition, the rule accuracy and the rule normalized quality are used to calculate the color of the rule polygon. The more accurate the rule, the redder the polygon, while the lower the rule quality, the more bright the polygon,

6 CViz Implementation and Experiment

The CViz system has been implemented in Visual C++ 6.0. The data preparation is accomplished by choosing a data file and an attributes file which describes the attributes, including attribute name, type, length, position in the tuple, domain, etc. This is implemented in dialog windows (under the file menu and

```
Procedure for drawing rule polygons
  $preCond$ = first condition;
  repeat
    $postCond$ = next condition;
    for each $or$ component of $preCond$ do
      if $preCond$ corresponds to a categorical attribute
      then compute the points $p_0, p_1$ according to the specific categorical value;
      else compute the points $p_0, p_1$ according to a numerical interval;
      for each $or$ component of $postCond$ do
        if $postCond$ corresponds to a categorical attribute
        then compute the points $p_2, p_3$ according to the specific categorical value;
        else compute the points $p_2, p_3$ according to a numerical interval;
        draw a small polygon with four points $p_0, p_1, p_2$, and $p_3$;
    $preCond$ = $postCond$;
  until all conditions are processed;
  draw the small polygon from last condition to the decision attribute.
```

Fig. 4. Procedure for drawing rule polygons

setting menu). The steps of visualizing and discovering knowledge are controlled by the *control menu*, which consists of the five steps.

CViz has experimented with several data sets from UCI Repository of Machine Learning Databases [12], including IRIS, Monk's, etc. and also with artificial data sets. An artificial data set is designed to involve two numerical attributes *Age, Score*, one condition categorical attribute *Color*, and one decision categorical attribute *Class*. *Age* is of integer type and has range $[0, 90]$. *Score* is of real-valued type and has range $[0.0, 5.0]$. *Color* has four discrete values: *red, blue, yellow*, and *green*. The decision attribute has two class labels: *good* and *bad*. The relationship between the class labels and the condition attributes in the data set is designed as follows:

```
IF   (30<Age<=60) and (1.5<Score<=3.5) and (Color=red or yellow)
THEN Class=bad
ELSE Class=good
```

Fig. 5 through 9 illustrate the original data visualization, discretized data visualization, all final rules visualization, rules for class *bad*, and rules for class *good*, respectively. If many rules are obtained and visualized on the same display, it is hard to distinguish them. CViz allows user to specify a class label to view all rules that have this class label as the decision value. Fig. 8 and Fig. 9 are decomposed from Fig. 7.

From Fig. 5 through 9, we can see that one rule is obtained for class *bad*, which is the same as the above one; and three rules are for class *good*, which correspond to three conditions:

```
Age<=30 or Age>60,
```

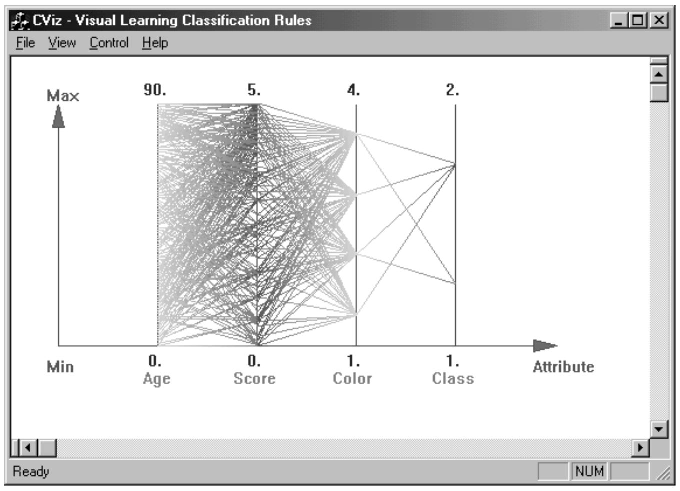

Fig. 5. The original data visualized on the parallel coordinates

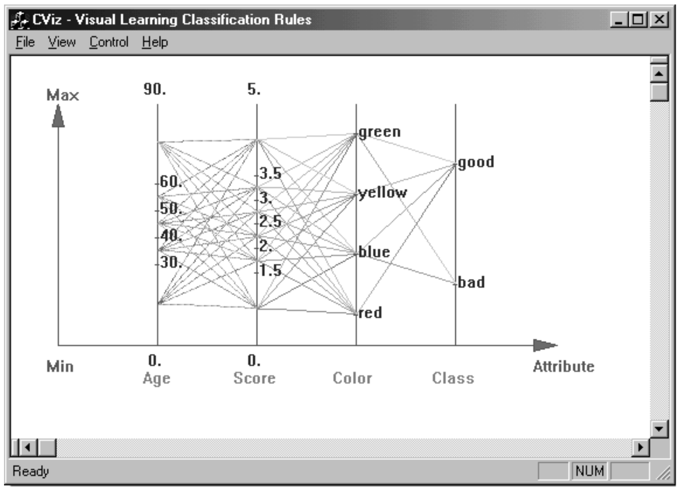

Fig. 6. Discretization of continuous attributes with EDA-DB method

```
Score<=1.5 or Score>3.5,
Color!=(red or yellow),
```

respectively.

Fig. 10 and Fig. 11 show the final results that ELEM2 learns from the Monks-1 and IRIS data sets [12], which involve 9 and 8 classification rules, respectively. In Fig. 10, five rules are obtained for class *0*, while there are four rules for class *1*. In Fig. 11, we have four rules for class *3*, three rules for class *2*, and one rule for class *1*, respectively. Fig. 12 illustrates all rules for class *2* discovered from the IRIS data set, and the rules for class *1* and class *3* discovered from the IRIS

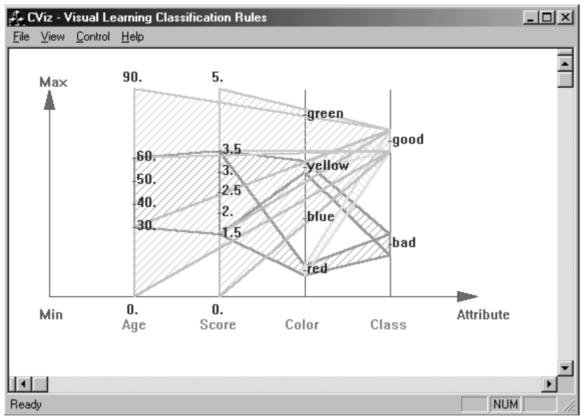

Fig. 7. All rules discovered from the artificial data set

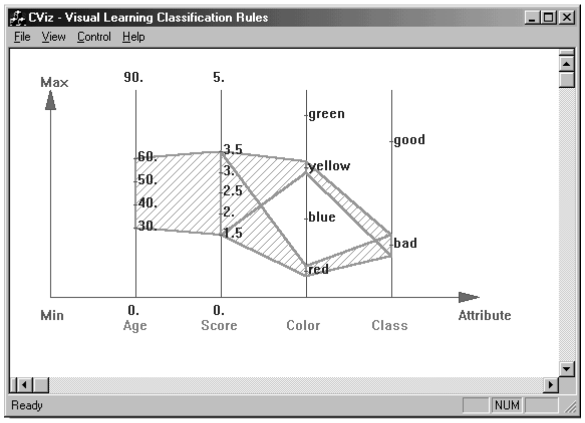

Fig. 8. The rules for class *bad*

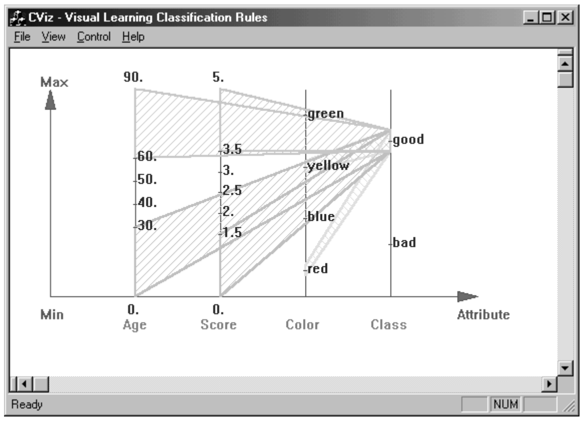

Fig. 9. The rules for class *good*

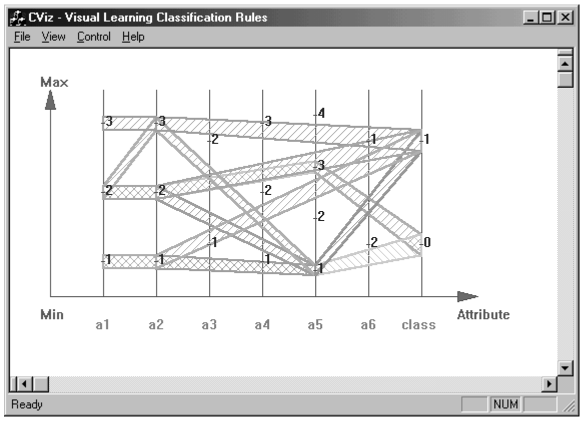

Fig. 10. The rules discovered from the *Monks-1* data set

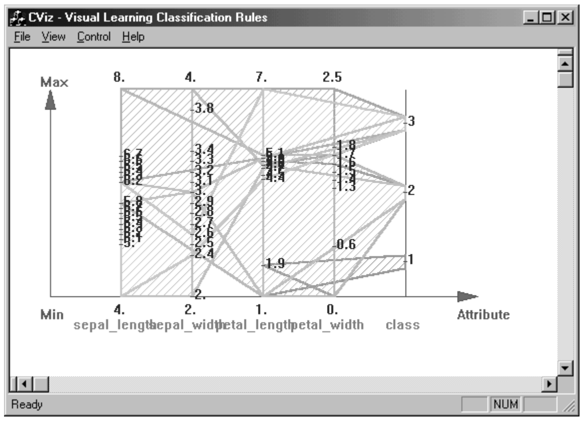

Fig. 11. The rules discovered from the *IRIS* data

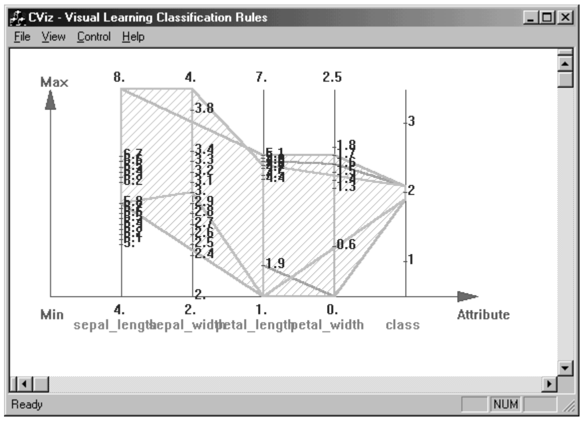

Fig. 12. The rules for class *2* discovered from the *IRIS* data

data set have the similar display form.

7 Related Work

Keim and Kriegel compare the different techniques for visualizing information [11] and divide these techniques into eight categories: geometric techniques, icon-based techniques, pixel-oriented techniques, hierarchical techniques, graph-based techniques, 3D techniques, dynamic techniques, and hybrid techniques. Fukuda *et al.* proposed the SONAR system which discovers association rules from two dimensional data by visualizing the original data and finding an optimized rectangular or admissible region [6]. Han and Cercone implemented the DViz system for visualizing various kinds of knowledge [8] and the AViz system for visualizing three dimensional numerical association rules [9]. Silicon Graphics developed a series of visualizers [7] to visualize different knowledge, like decirion trees, neural nets, etc. There have been many well-known visualization system like VisDB [11], Spotfire [1], Visage [4], KnowledgeSeeker, DataMind [7], etc. but none of them implements visualization of the entire process of knowledge discovery.

8 Concluding Remark

CViz is an interactive system for visualizing and learning classification rules. It can be used to visualize the original data on a parallel coordinates system. The user can interactively reduce the data set horizontally and vertically by removing irrelevant attributes and/or attribute values. The user can also interactively select his/her favorite approaches to discretizing numerical attributes. The discretized attributes are treated as categorical ones and each interval corresponds to a discrete value. The ELEM2 induction algorithm is used to learn classification rules which are displayed in visual forms. The user can interactively choose a class to view the corresponding rules. Classification rules may have complex logical form. In our implementation, we emphasize the human-machine interaction, since we believe that interactive visualization plays an important role in the process of discovering knowledge. Our experiment results have also demonstrated that it is useful for users to understand the relationships among data and to concentrate on the meaningful data to discover knowledge. The capability of CViz will be expanded so that the user can interactively specify the rule accuracy threshold and/or the rule quality threshold, which might be used to limit the search space for final rules.

It could be easy to adapt CViz to other rule-based learning systems because the CViz system just encompasses the learning algorithm as a learning step of the visualization process and the difference between rule-based learning systems is each of them uses a different learning algorithm. It remains a research topic, however, to adapt CViz to learning systems other than rule-based ones because CViz is based on the parallel coordinate technique which may not be plausible for representing decision trees and neural networks.

Acknowledgments: The authors are members of the Institute for Robotics and Intelligent Systems (IRIS) and wish to acknowledge the support of the Networks of Centers of Excellence Program of the Government of Canada, the Natural Sciences and Engineering Research Council, and the participation of PRECARN Associates Inc.

References

1. Ahlberg, C. 1996. Spotfire: An Information Exploration Environment, *ACM SIGMOD Record*, 25(4).
2. An, A. and Cercone, N. 1998. ELEM2: A Learning System for More Accurate Classifications. *Proc. of the 12th Biennial Conference of the Canadian Society for Computational Studies of Intelligence*, Vancouver, Canada.
3. An, A. and Cercone, N. 1999. Discretization of Continuous Attributes for Learning Classification Rules, *Proc. of the 3rd Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp.509-514.
4. Derthick, M., Kolojejchick, J. and Roth, S. F. 1997. An Interactive Visualization Environment for Data Exploration, *KDD-97*, pp.2-9.
5. Fayyad, U.M. and Irani, K.B. 1993. Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. *IJCAI-93*. pp. 1022-1027.
6. Fukuda, T., Morimoto, Y., Morishita, S. and Tokuyama, T. 1996. Data Mining Using Two-Dimensional Optimized Association Rules: Scheme, Algorithms, and Visualization, *Proc. of the ACM SIGMOD International Conference on Management of Data*, pp.13-24.
7. Groth, R. 1998. Data Mining: A Hands-on Approach for Business Professionals, *Prentice Hall PTR.*
8. Han, J. and Cercone, N. 1999. DVIZ: A System for Visualizing Data Mining, *Proc. of the 3rd Pacific-Asia Conference on Knowledge Discovery in Databases*, pp.390-399.
9. Han, J., Cercone, N. and Hu, Xiaohua 1999. An Interactive Visualization System for Mining Numerical Association Rules, *Granular Computing and Data Mining*, ed. by T. Y. Lin, Y. Y. Yao, and L. A. Zadeh, to appear.
10. Inselberg, A. and Dimsdale, B. 1990. Parallel Coordinates: A Tool for Visualizing Multi-Dimensional Geometry, *Proc of IEEE Visualization Conference*, pp.361-370.
11. Keim, D. A. and Kriegel, H. P. 1996. Visualization Techniques for Mining Large Databases: A Comparison, *Transactions on Knowledge and Data Engineering, 8(6)*, pp.923-938.
12. Murphy, P. M. and Aho, D. W. 1994. UCI Repository of Machine Learning Databases, URL: *http://www.ics.uci.edu/m̃learn/MLRepository.html.*
13. Piatetsky-Shapiro, G. 1991. Discovery, Analysis, and Presentation of Strong Rules, *Knowledge Discovery in Databases* ed. by Gregory Piatetsky-Shapiro and William J. Frawley, pp.229-260.
14. Srikant, R. and Agrawal, R. 1996. Mining Quantitative Association Rules in Large Relational Tables, *Proc. of the ACM SIGMOD International Conference on Management of Data*, pp.1-12.