PERGAMON

Expert Systems with Applications 22 (2002) 303–311

Expert Systems with Applications

www.elsevier.com/locate/eswa

Rule induction in data mining: effect of ordinal scales

Helen M. Moshkovich[a], Alexander I. Mechitov[a], David L. Olson[b,*]

[a]*Michael E. Stephens College of Business, University of Montevallo, Montevallo, AL 35115 USA*
[b]*Department of Management, University of Nebraska, Lincoln, NE 68588-0491, USA*

Abstract

Many classification tasks can be viewed as ordinal. Use of numeric information usually provides possibilities for more powerful analysis than ordinal data. On the other hand, ordinal data allows more powerful analysis when compared to nominal data. It is therefore important not to overlook knowledge about ordinal dependencies in data sets used in data mining. This paper investigates data mining support available from ordinal data. The effect of considering ordinal dependencies in the data set on the overall results of constructing decision trees and induction rules is illustrated. The degree of improved prediction of ordinal over nominal data is demonstrated. When data was very representative and consistent, use of ordinal information reduced the number of final rules with a lower error rate. Data treatment alternatives are presented to deal with data sets having greater imperfections. © 2002 Elsevier Science Ltd. All rights reserved.

Keywords: Data mining; Ordinal data; Classification

1. Introduction

Data mining may be viewed as the extraction of patterns and models from observed data (Berry & Linoff, 1997). The area of data mining is very broad as it incorporates techniques and approaches from different research disciplines. Data mining is usually used to answer two main types of application questions (Edelstein, 1997): (1) *generate predictions* on the basis of available data or (2) *describe behavior* captured in the data. Examples of the first type of tasks include banking (Kiesnoski, 1999), interested in the success of prospective loans; insurance (Goveia, 1999) interested in the probability of fraud, marketing (Peacock, 1998) interested in identifying the best prospects for direct-mail list campaigns. Examples of the second type include finding out which products are sold together, what infections are connected with surgery, in what time ranges which group of customers use a service. In this article we will concentrate on the first type of tasks.

To answer the first type of question, three main approaches are used (Edelstein, 1997): (1) classification, (2) regression, and (3) time-series. These models are differentiated on the basis of what we want to predict. If we want to forecast continuous values of the output attribute, regression analysis is mostly used (time-series if we are concerned with distinctive properties of time). If we have to predict a categorical value for a specific data item (categorical data fits into a small number of discrete categories such as 'good credit history' or 'bad credit history'), we have a classification task to solve. Examples of this type of tasks are medical or technical diagnostics, loans' evaluation, bankruptcy prediction, etc.

Classification is one of the most popular data mining task. There are many different methods, which may be used to predict the appropriate class for the objects (or situations). Among the most popular one are: logistic regression, discriminant analysis, decision trees, rule induction, case-base reasoning, neural networks, fuzzy sets, and rough sets (Kennedy, Lee, Van Roy, Reed, & Lippman, 1997). Other methods are used as well.

The majority of data mining techniques can deal with different data types. The traditional types of data mentioned in applications are continuous, discrete, and categorical. Among these three categories continuous scales are usually assumed to be numerical, while categorical and discrete data involve variety. Categorical information may be either ordinal (e.g. 'high', 'medium', 'low'), or nominal (e.g. 'blue', 'yellow', 'red'). Discrete data is an uncertain data type as different models can treat this type of data differently. The majority of data mining models (e.g. regression analysis, neural networks, etc.) will consider discrete data as numeric and apply models suitable for numeric data (Lippmann, 1987). In other cases, discrete data can be treated as categorical, viewing it as numeric codes for nominal data as done in decision trees or rough sets approaches (Quinlan, 1990; Slowinski, 1995). In the latter case, it can be ordinal as well, e.g. if we describe cars, we

* Corresponding author. Tel.: +1-402-472-4521; fax: +1-402-472-5855.
E-mail address: dolson3@unl.edu (David L. Olson).

PII: S0957-4174(02)00018-0

can have an attribute 'number of doors' with possible values of 2, 4, 5, 6. It can be considered numeric (with the more doors we have the better), or '4 doors' can be the most desirable characteristic, while others are less attractive.

Use of numeric information usually provides possibilities for more powerful analysis than ordinal data. On the other hand, ordinal data allows more powerful analysis when compared to nominal data. It is therefore important not to overlook the knowledge about ordinal dependencies in the data sets. Although not as popular in the area of data mining, the qualities of ordinal data were rather thoroughly examined in the area of decision analysis and expert systems (Ben-David, 1992; Larichev & Moshkovich, 1994, 1997; Mechitov, Moshkovich, Olson, & Killingsworth, 1995; Mechitov, Moshkovich, Bradley, & Schellenberger, 1996; Yager, 1981). Implementation of this knowledge may be useful in some classification problems.

The rest of the paper will investigate additional data mining support available from ordinal data. The effect of inclusion of the information on ordinal dependencies in the data set on the overall results of constructing decision trees and induction rules will be illustrated. Possibilities for using ordinal properties to evaluate quality of the training data set will be discussed.

2. Classification task and data mining methodology

In many cases, previous experience in business decisions is coded in the form of a classification. Classification predicts a categorical value for a specific data item and uses a small number of discrete categories (e.g. 'good credit history', 'bad credit history', etc.). Classification is one of the most popular knowledge discovery tasks applied to a variety of fields. Methods from different research areas are devoted to the analysis of such a problem.

Logistic regression is a traditional approach to a classification problem with numeric scales. It has the same assumptions as regression analysis, but allows use of discrete values for classes. Berry and Linoff (1997) comment that for data sets consisting entirely of continuous variables, regression is probably a very good method. Regression analysis identifies an additive function which links case values with the outcome class with the least error. Regression models can be designed to reflect non-linearities, such as the interactions across the variables describing cases or objects. Once the model is obtained it can be used to find corresponding class for new objects.

Artificial neural networks act much the same way, except that they try many different coefficient values in the additive functions to fit the training set of data until they obtain a fit as good as the modeler specifies. Artificial neural network models have the added benefit of considering variable interactions, giving it the ability to estimate training data contingent upon other independent variable values. (This could also be done with regression, but would require a tremendous amount of computational effort.) Artificial neural network models are based on automatic fitting of a model including non-linear combinations of variables. These models are self-adjusting, in that they train on a given set of input data. During the training stage, if the current model correctly predicts the next observation of data, it continues on to the next observation. However, if the current model is incorrect, the model adjusts to more accurately predict the current observation. This may lose the accuracy developed for past learning, so the system iterates through the data until an acceptable level of accuracy is obtained.

Neural networks have been applied to almost all types of data mining applications. They have the feature that the model remains hidden to the analyst, so that while the computer can quickly give its prediction for a variable, the analyst cannot take the model apart to understand why. Therefore, if explanation of results is very important, usually other methods, especially rule induction systems, are used instead of neural networks.

While logistic regression, discriminant analysis, and neural networks are best to use in cases when all attributes are presented with continuous scales, decision trees usually can deal with continuous and categorical data simultaneously (Quinlan, 1990). Decision trees are useful because they allow the decision process to be unveiled from the data (Linoff, 1998). Each branch in the tree represents a decision made on a particular attribute. The algorithm automatically determines which attributes are most important. The method has relative advantage over neural network and genetic algorithms in that a reusable set of rules are provided, thus explaining model conclusions (Michie, 1998). There are quite a number of systems for tree construction. Almost all data mining systems include tools for this type of data analysis. One of the most popular decision tree systems is C4.5 (Quinlan, 1993). It was used in a very large number of applications as well as the basis for comparison of new machine learning algorithms. The system also generates rules, which can be applied in production rule expert systems.

Fuzzy and rough sets approaches try to take into account the uncertainty of data. The assumption in traditional decision trees and rule induction systems is that the distinctions between classes and attribute levels are clear and crisp (e.g. in that somebody is either old, middle-aged, or young, with distinct and well-defined limits). However, someone aged 34 is not all that young, nor is someone aged 36 all that much older than someone 34. Fuzzy logic (Zadeh, 1965) considers degrees of membership in categories. Someone who is 36, for instance, might have a membership function of 0 for the category old, a 0.9 membership function for the category middle-aged, and a 0.3 membership function for the category young. A membership function provides a relative rating between 0 and 1 of membership in a set. As a result, fuzzy analysis provides for each instance a distribution of membership functions for all classes (or can choose a class with the highest membership as the outcome class).

This approach is considered to be good when we do not know the exact information about the data itself but have experts who are able to give stable numeric estimates for the membership functions.

Yager (1981) introduced a fuzzy set decision model in which the membership function is evaluated using an ordinal scale with seven gradations. Close to this approach is the theory of rough sets (Pawlak, 1991; Slowinski, 1995). In addition to precise rules (stating a single class for the data item) as done in rule-based systems, the rough sets approach introduces approximate rules (which can include several classes for the same data item). In this sense, rough sets can be viewed to some extent as fuzzy sets with a three-valued membership function (belongs to this class, does not belong to this class, may belong to this class). Hong, Wang, Wang, and Chien (2000) and Yahia, Mahmod, Sulaiman, and Ahmad (2000) present combinations of these approaches.

Rather a large number of classification tasks in business applications may be viewed as tasks with classes reflecting the levels of the same property. Evaluating creditworthiness of clients is rather often measured on an ordinal level as, e.g. 'excellent', 'good', 'acceptable', or 'poor' (Ben David, Sterling, & Pao, 1989). Articles submitted to the journals in the majority of cases are divided into four groups: 'accepted', 'accepted with minor revisions', 'may be accepted after revision and additional review', 'rejected' (Larichev & Moshkovich, 1997). Applicants for a job are divided into accepted and rejected, but sometimes there may also be a pool of applicants left for further analysis as they can be accepted in some circumstances (Ben-David, 1992; Slowinski, 1995). Bankruptcy prediction may be made with some numeric measure, or use of ordered symbolic values (Messier & Hansen, 1988). In car selection, cars may be divided into groups 'very good', 'good', 'acceptable', 'unacceptable' (Bohanec & Rajkovic, 1990). In all these examples, the peculiarity of the tasks is that data items with 'better' qualities (characteristics) logically are to be presented in better classes: the better the article in its characteristics the more close it is to the class accepted. Thus, if attributes characterizing data items are rank ordered from the highest to the lowest level (as the classes themselves), there is a dependency between the attributes' level the data item has and the class appropriate for it. This information (if taken into account) can be useful in data mining. In our opinion this information can: (1) lead to a smaller number of rules with the same accuracy; (2) enable the system to extend obtained rules to instances not presented in the training data set; (3) enlarge the supporting number of cases for the rule. In spite of this fact not many data mining techniques take into account this type of information. As we showed earlier, the majority of data mining approaches to classification work with numerical information. One of the few approaches that can deal with both numerical and categorical information is a decision tree and some induction algorithms (e.g. rough sets). But the majority of them do not differentiate categorical and ordinal information.

C4.5 is a popular software package for this type of task (Quinlan, 1993). It produces a decision tree and rules for almost any type of input (label, discrete, continuous) and has proven to be effective in many tasks. Still, it does not incorporate knowledge about ordinal dependencies in the data sets. The latest version of this software (C5 for Unix and See5 for Windows platform) includes some improvements into the system. One improvement is the possibility to mark some scales for attributes as 'ordered' (http://www.rulequest.com). This feature is mentioned in passing with no explanations of how it is used in the model, or any examples provided for its illustration. Therefore, we decided to analyze several tasks using this feature and to evaluate the result. We used several databases from the machine learning repository (http://www.ics.uci.edu/~mlearn/MLSummary.html) for this purpose.

3. Experimental results with categorical and discrete scales

We have selected several databases from the repository with the following characteristics: (1) a small number of ordered classes and (2) a majority of categorical attributes in the description of instances. The first database selected was car evaluation database (donated by Bohanec and Raikovic, 1990). This database represents a pure case of ordinal classification. It contains 1728 instances characterizing cars on the basis of six attributes. Four classes are used: unacceptable, acceptable, good, very good (it can be easily seen that those are the quality levels for the same overall acceptability of the car for a customer). Six attributes with categorical scales were presented as follows:

- buying price (vhigh, high, medium, low);
- price of maintenance (vhigh, high, medium, low);
- number of doors (2, 3, 4, 5 or more);
- capacity in terms of persons to carry (2, 4, more);
- trunk size (small, medium, big);
- safety (low, medium, high).

All these attributes have a monotone dependency with the corresponding class: very high price is worse than high with all other characteristics at the same level, low safety if worse than medium or high (especially with all other characteristics at the same level), and so on.

First, the traditional decision tree analysis was carried out for all 1728 cases. In this case all scales were marked as discrete (or categorical). The resulting tree contained 131 nodes and produces 3.7% errors. The corresponding set of induction rules consisted of 74 rules and produced 4.1% errors.

After that we marked all these attributes as ordered (using new feature of See5). The resulting tree produced 59 nodes and 0.6% errors. A corresponding set of induction rules contained 51 rules with .2% errors. Thus, even with very

Table 1
Data for car evaluation data set

Results for nominal scales: 1728 cases					Results for ordinal scales: 1728 cases				
Decision tree		Rules			Decision tree		Rules		
Size	Errors	No.	Errors		Size	Errors	No.	Errors	
131	64 (3.7%)	74	71 (4.1%)		59	10 (0.6%)	51	4 (0.2%)	
(a)	(b)	(c)	(d)	Class	(a)	(b)	(c)	(d)	Class
1186	21	3		(a): Unacc	1210				(a): Unacc
19	361	2	2	(b): Acc		382	2		(b): Acc
	11	55	3	(c): Good	1		68		(c): Good
	6	4	55	(d): Vgood		1		64	(d): Vgood

good stable data (representative data set with few inconsistencies) the introduction of the ordinal dependencies into the model has considerably improved the result (Table 1). A smaller number of induced rules with lower level of errors are usually a goal in decision tree analysis that is hard to achieve.

Analogous experiments were carried out with several other databases. The next database used was the database for teaching assistant evaluation (TAE) (Loh & Shih, 1997). The data consist of evaluations of teaching performance over three regular semesters and two summer semesters of 151 teaching assistant (TA) assignments. Three overall classes were used in the problem. Categories low, medium, and high evaluated overall quality of the teaching. Five attributes were used to describe instances: course instructor (categorical, 25), course (categorical, 26), class size (numerical, continuous), plus two attributes considered binary by the authors. The first one was 'English language being native to the TA' (1—English speaker, 2—non-English speaker). We considered that usually, a native English speaker has better opportunities for better teaching than the non-native one. The second binary attribute marked the course taught in a regular or summer semester (1—summer semester, 2—regular semester). We considered the regular semester to be better for better teaching than the summer semester. Thus, we marked the last two attributes as ordered. This database was rather poor for rule induction (too many instructors and courses with a small number of repeating values in 151 cases). The initial variant produced a decision tree with 9 nodes but 43.7% of errors in classification, or 8 induction rules with the same level of errors. The variant analysis with partially ordered information (two attributes) produced a decision tree with 34 nodes but only 11.9% of misclassifications and the corresponding rule base contained 25 rules with the same level of errors. Even in this case, the simple introduction of ordinal dependencies produced essential improvement to the classification.

The third database selected for this study represents data for differentiating DNA sequences (Noordewier, Towell, & Shavlik, 1991). The problem posed in this dataset is to recognize, given a sequence of DNA presented by 60 letters, the boundaries between exons (parts of the DNA sequence retained after splicing) and introns (parts of the DNA sequence that are spliced out). Three classes are presented in data: n (no boundary), ei (exons to introns boundary), and ie (introns to exons boundary). Each of 60 attributes is presented by one of the letters: A, T, G, C plus letters N, D, S, R to introduce some of the combinations of the previous letters (e.g. D is A or G or T). The overall database contains 3190 instances. The percentages of letters N, D, S, and R are very small (less than 0.002%).

In the initial data set all attributes are presented as categorical. The data set of 3190 cases produces very good results with the C4.5 algorithms. It is included with the See5 system under the name 'Genetics' as a sample case for pure categorical data. The resulting decision tree has 169 nodes with 3.7% of errors. The rule set has 84 rules with the same level of errors. The idea to apply ordered scales for this problem arose from the observation that the letters in the scales were not in an alphabetic order. That implied special meaning in their order for the researchers. Our trial showed that by marking all scales as ordinal it was possible to improve the results for this problem as well. The initial set of 3190 cases produced 111 nodes (instead of 169 with categorical scales) with a 1.8% error rate, while the rule set contained 79 rules and had 2.2% error rate.

Ordinal classification tasks are wide spread in business applications. In the majority of business applications the decision classes represent the levels of quality on the overall instance estimation. Loan applications' evaluation is one of these numerous examples and a very popular task in data mining. We used the 'German credit data set' from the machine learning repository. The data set is provided by Dr Hans Hofmann from the Insitut fur Statistik und Ekonimetric Universitut at Hamburg. It contains 1000 instances of loan applications with evaluation as 'good for loan' (class 1) and 'bad for loan' (class 2). The attractive feature of this data set is that all the 20 attributes used are described in the documentation (while many other data sets on loan applications are coded with no meaningful data of the attributes used).

The data set is presented by 20 attributes with 7 of them being numerical or continuous, and 13 marked as categorical. This data set seemed as a very interesting variant for

trying out ordered scales for some (or all) of categorical attributes. The attributes used were as follows:

1. Status of existing checking account (categorical): A11–A14.
2. Duration of the loan in months (numerical).
3. Credit history (categorical): A30–A34.
4. Purpose of the loan (categorical): A40–A410.
5. Credit amount (numerical).
6. Savings account/bonds (categorical): A61–A65.
7. Present employment (categorical): A71–A75.
8. Installment rate in percentage of disposable income (numerical).
9. Personal status and sex (categorical): A91–A95.
10. Others debtors/guarantors (categorical): A101–A103.
11. Present residence since (numerical).
12. Property (categorical): A121–A124.
13. Age in years (numerical).
14. Other installment plans (categorical): A141–A143.
15. Housing (categorical): A151–A153.
16. Number of existing credits at this bank (numerical).
17. Job (categorical): A171–A174.
18. Number of people being liable to provide maintenance for (numerical).
19. Telephone (categorical): A191–A192.
20. Foreign worker (categorical): A201–A202.

Categorical data are presented with description of each scale values, e.g. A11 is 'the existing checking account is less than 0 DM', A12 is 'the existing checking account is less than 200 DM', and so on. The presence of these definitions allowed us to analyze how the scale values are connected to the two resulting classes (good or bad for loan). Only two of the scales for categorical data were not clearly ordinal (Attribute 4: purpose of the loan and Attribute 9: personal status and sex). There is a possibility that the data were ordinal for the bank but we had no knowledge to state this. With other parameters it was much easier (e.g. better credit history is better for a good loan). Thus, all other attributes were marked as ordered for our analysis. The initial data (with categorical and numerical data only) provided a decision tree with 89 nodes and 14.3% error rate or 29 rules with 17.0% error rate. The same data for which ordered scales were marked produced a larger set of decision tree nodes and number of rules (109 and 35 correspondingly) but lowered the error rate to 10.3% for decision tree and 11.4% for rules.

Usually in loan application analyses, the errors in assigning 'good application' to an actual bad category is considered to be much more significant than the reverse (as it usually leads to direct losses while the reverse is more a lost opportunity with much less profit impact). To take this into account we carried out the same analysis with costs for the errors as was suggested in the initial data set: the error of assigning the class 'good application' to a bad application is five times more expensive than the reverse. Using costs in decision tree construction led to a 25.8% error rate at 0.26 cost while usage of rules lead to a 31.2% error rate at cost 0.56. Using ordinal scales the results were a 16% error rate for the decision tree with cost 0.17 and a 30% error rate for rules at cost of 0.34. Here the advantage of using ordinal scales is even more evident. We are sure that the result could be improved in the number of rules as well if we were able to have an expert consultation about the order for some of the longer scales used in the task.

To make sure that ordinal properties give advantage not only in deriving decision rules but also in implementing them to the testing data sets, all these data sets (except TAE due to its low quality) were used to construct decision rules on the basis of the 60% sample from the initial data set (the training data set). The other 40% of instances were used as a test data set to evaluate the quality of the rules. The summarized data on number of nodes, and error rates are presented in Table 2.

These results support the notion that the presence of ordinal dependencies in the data set should be accounted for as they may provide better results within the same task environment. At the same time, how ordinal information is accounted for in the traditional decision tree model is rather simple. This information is used while making splitting decisions. If scales are ordinal (e.g. high, medium, and low), we can look for a split not only among high, medium, and low, but also, high–medium and medium–low (and we do not need to analyze the combination high–low). This is a simple and useful technique that can provide effective results as we have shown earlier. But information about the ordinal nature of the problem can lead to a much more rich set of conclusions than this. In Section 4 we will look more closely at the possibilities of implementation of ordinal dependencies in ordinal classification tasks.

4. Ordinal information in a classification task

Tasks where data items are distributed among classes which reflect the levels of the same property are popular in the business world. In the car evaluation case, cars are divided into groups very good, good, acceptable, unacceptable. In all examples presented in Section 3, the peculiarity of the tasks is that data items with better characteristics are to be present in better classes. Thus, if attributes characterizing data items are rank ordered from the highest to the lowest level (as the classes themselves), there is a dependency between the attributes' level the data item has and a class it belongs to. The results of this quality to some extent were demonstrated in the previous examples.

Let us consider a less complicated case of loan evaluation based on three attributes:

1. age with possible values of young, middle, and old;
2. income with possible values of high, average, and low and

Table 2
Using ordinal scales instead of categorical in some data sets

	Databases					
Number of	Cars	TAE	Genetics	Loan	Loan with costs	
Classes	4	3	3	2	2	
Attributes	6	5	60	20	20	
All cases (categorical/ordinal)						
Cases	1728	151	3190	1000	1000	Cost
Nodes	131/59	9/34	169/111	89/109	88/130	0.26/0.17
Errors (%)	3.7/0.6	43.7/11.9	3.7/1.8	14.3/10.3	25.8/16.0	
Rules	74/51	8/25	84/79	29/35	30/63	0.56/0.34
Errors (%)	4.1/0.2	43.7/11.9	3.7/2.2	17.0/11.4	31.2/30.0	
60% of cases (categorical/ordinal)						
Cases	1037		1914	600	600	Cost
Nodes	78/41		134/59	49/64	82/61	0.21/0.20
Errors (%)	5.8/1.5		3.5/3.4	14.8/10.3	21.3/19.3	
Rules	54/32		61/40	22/27	43/36	0.42/0.34
Errors (%)	4.7/1.1		1.7/1.7	16.3/11.2	28.8/25.2	
Test cases (categorical/ordinal/cost)						
Tree-errors (%)	12.0/5.2		8.0/6.8	28.5/29.5	43.8/32.3	0.83/0.66
Rules-errors (%)	9.4/5.2		6.9/6.0	28.5/30.8	41.3/32.3	0.70/0.57

3. risk for the loan to be paid on-time with possible values of high, medium, and low.

The data comes from a hypothetical example presented in Olson and Courtney (1998).

650 instances of loan applications evaluated against these attributes were presented in the data set with 585 cases being 'loans paid on-time' and 65 cases 'loans paid late or default'. These data were supposed to be used to formulate several simple rules for loan officers in their everyday practice. Application of a standard version of the decision tree and rule induction algorithm produced one simple rule: consider all applications as good (yielding a 10% error rate as we have only 65 'bad credit' cases among 650).

Lets look at this problem as an ordinal classification task. Attribute scales were considered to be ordinal by loan officers with the following sequences:

1. for risk: low risk was considered to be better than medium (for the loan to be paid on-time), and medium better than high;
2. for income: high income was considered to be better for 'good loan' than average than low;
3. for age: old age was considered to be better for good loan than middle than young.

The ordinal classification task assumes that 'cases with better characteristics should be placed in a better class'. In this example it tells us that for example, a middle-aged applicant with low risk and high income has a better chance to pay the loan on-time than a young applicant with low risk and high income.

Formally the ordinal classification problem can be presented as follows. Let U (universe) present all possible objects in the task. $X \subseteq U$ is any subset of these objects (data items in a data set). Objects from X are distributed among k classes: $C_1, C_2, \ldots, C_k$, indicating the degree in which object satisfies the overall output quality (from the lowest to the highest). This means that if $x, y \in X$, and $C(x) > C(y)$, object x has higher overall output quality than object y. Each object is described by a set of attributes $A = \{A_1, A_2, \ldots, A_p\}$. Each attribute A_i has an ordinal scale S_i (possible values are rank ordered from the lowest to the highest in quality against this attribute). $A_i(x) \in S_i$, $A_i(y) \in S_i$, and $A_i(x) > A_i(y)$ means object x has higher quality on attribute A_i than object y.

This presentation of the ordinal classification task allows use of this knowledge to make some additional conclusions about the quality of the training set of objects in X. Ordinal classification allows to introduce the notion of the consistency of the training set as well as completeness of the training set. These two notions are described in Sections 4.1 and 4.2.

4.1. Consistency of the training set

Quinlan (1990) remarked that if attributes describing objects in a classification are adequate it is always possible to construct a tree ideally distributing objects among classes. Attributes are considered to be adequate if there are no two objects in the training set that have the same value for every attribute but belong to different classes. In the case of the ordinal classification task this quality of consistency in classification (the same quality objects have to belong to the same class) can be essentially extended. With categorical data, only objects with the same set of attribute values have to be in the same class (to have a consistent classification). In ordinal classification all objects

with higher quality among attributes have to belong to a class at least as good as objects with lower quality. This condition can be easily expressed as follows: if $A_i(x) \geq A_i(y)$ for each $i = 1, 2, \ldots, p$, then $C(x) \geq C(y)$.

Let us return to our loan example. Let us assume that the data showed a middle-aged applicant with average income and medium risk usually pays a loan on-time. This means that all applicants who are old or middle-aged with high or average income and low or medium risk usually pay their loan on-time. Thus one special case allows us to formulate a rule that includes seven additional combinations of possible values against these three attributes:

(old, high income, low risk);
(old, high income, medium risk);
(old, average income, low risk);
(old, average income, medium risk);
(middle age, high income, low risk);
(middle age, high income, medium risk);
(middle age, average income, low risk).

All these cases have at least one attribute value that is of better quality for paying the loan on-time than the initial case (middle age, medium income, medium risk). Using this quality we can evaluate the consistency of the training set as for each case we can calculate the number of cases contradicting it, using not only cases with the same set of attribute values (which may be rather few) but also all dominating (or dominated) ones. In our loan example we will have to take into account all cases representing the above seven combinations (plus the initial case) and mark number of times they were assigned to different classes.

If the data set is highly inconsistent it is not reasonable to expect any logical result from any type of the mining tool. If inconsistency is slight, almost any mining tool will be able to deal with it reasonably.

Let us assume that there is a small number of objects in the data set whose classification contradicts a large number of other objects' classes. If we are interested in a consistent set of logical rules, there can be several venues to follow. These objects may be eliminated from the data set (like outliers in a statistical analysis). They can be reclassified (put into a different class more in accordance with the ordinal dependences of the task). There are approaches which helps in finding the minimal number of such changes that results in a consistent data set (Larichev & Moshkovich, 1994). Sometimes the situation can be discussed with an expert in the field (concerning contradictory objects and/or ordinal statements).

The analysis of 650 cases for loan applications in our example (585 of whom paid on-time) showed lot of cases contradicting each other directly (the same instances with different final classes) and indirectly (through ordinal dependencies). For example, there were 192 instances with characteristics (young, average income, high risk) belonging to class 'loan paid on-time'. They contradicted 10 cases being in the class 'loan not paid on-time' and some other cases with better characteristics in the class 'loan not paid on-time'. This is rather a usual problem with loan evaluation data as the vast majority of loans would usually be paid back (making the first class dominate the data). At the same time, loan officers are interested in a rather conservative set of rules as cost of an error of putting a 'bad loan' into a better class is much more than the reverse. Usually this problem is dealt with by introducing costs for mistakes. In this case the costs were proposed as 1000 and 100 for more and less important error correspondingly. Using See5 with the costs produced a rule: every loan is to be considered 'not on-time' with 585 errors of classification (90% error at cost 90) which was also not satisfying as the rule to approve all loans with a 10% error rate for the task without costs.

Analysis of the ordinal structure of the cases showed that case with characteristics (young, income low, risk high) was presented in the class 'loan paid on-time'. This case was repeated 50 times in the data set with this class and five times in the second ('loan not paid on-time'). This is the worst possible combination of attribute values. If it constitutes a 'good loan' all loans are good and the result of data mining assigning all cases to this class is not surprising. This data set cannot produce any other reasonable result on this basis.

In some circumstances a highly inconsistent data set may indicate that the data are not discriminating between classes in this presentation. It may be necessary to recode data or use other attributes as well. In the case of ordinal dependencies between attribute scales and classes, we can rather easily obtain a very substantial amount of information about the consistency of the training set even if we do not have the objects with the same attribute values in it. As this requirement is based on the characteristics of the task logical analysis, the cause of the inconsistencies can be carried out and informed decisions may be made. If the position of the investigator is that we must not change anything in the initial data, a possible outcome is to allow more than one class for some of the objects, like a higher border for the class in rough sets (Slowinski, 1995).

4.2. Completeness of the initial data set

In general, if the size of the task is not very large we can evaluate the representativeness of the training set (discriminating power of the knowledge in it). We can form all possible objects in U (we can do that as we have a finite number of attributes with a small finite number of values in their scales) and check how many of them are presented in the training data set. It is evident that the smaller this proportion the less discriminating power we will have for the new cases.

Let us return to our example of 650 cases of loans with three characteristics: age, income, and risk. There are only 27 possible combinations of attribute values (three attributes with three possible values each). Although we had 650

cases, only 16 out these 27 combinations, were presented among them. This shows that 11 combinations may be very rare in real life cases, but it also gives us the idea that the predictive power of the induced rules will be lower due to these missing combinations.

Taking into account ordinal dependences between attributes and classes may allow us to make some valid conclusions even about some of the 'missing' attribute values combinations. For example, in our loan evaluation example, the combination (old, high income, medium risk) is not present in the data set, but as it was stated in Section 4.1, if the combination (middle age, average income, medium risk) is attributed to the class 'loan paid on-time', then the combination (old, high income, medium risk) can also be assigned to this class as it has two attribute values (age and income) more preferable than the initial one.

Ordinal classification allows a different view of the construction of decision rules from the training data set.

Let $X_j \subseteq X$ be a subset of objects from X belonging to class C_j. Two subsets can be differentiated among these objects. One subset will be called the lower border (LB_j) of the class and will consist of objects with the lowest possible attribute values in this class. The second subset will be called the upper border (UB_j) of the class and will consist of objects with the highest possible attribute values in this class.

These two borders accurately represent the jth class (as it was presented in the training set). We can classify any other object from X_j as belonging to this class just because its attribute values are between the values of objects from LB_j and UB_j. Let us look at object $x \in X_j$ which is not in the upper or lower border of the class. As it is not in UB_j it means that there is an object $y \in \mathrm{UB}_j$ for which $A_i(y) \geq A_i(x), i = 1, 2, \ldots, p$. This leads to $C(y) \geq C(x)$. Analogously there is object $z \in \mathrm{LB}_j$ for which $A_i(x) \geq A_i(z)$, $i = 1, 2, \ldots, p$. Thus $C(x) \geq C(z)$. But $C(y) = C(z) = C_j$. Thus, $C(x) = C_j$.

Let us say that in our example of loan evaluation the elements of the upper and lower borders of class 1 ('loan paid on-time') and class 0 ('loan not paid on-time') are as follows:

UB_1: (old, high income, low risk). This represents the best combination of attribute values in this task.
LB_1: (middle age, average income, medium risk), (young, average income, low risk).
UB_0: (old, high income, high risk), (young, high income, medium risk).
LB_0: (young, low income, high risk). This represents the least attractive combination of attribute values.

Constructed borders allow us to classify some objects with attribute values not presented in the training data set. Any object from U whose attribute values are between those presented in upper and lower border of the class may be classified as belonging to this class. For example, let us say we have a new loan application with a combination of characteristics as follows: (young, low income, medium risk). This combination was not presented in the training set. We can easily see that this loan application has less attractive attribute value in income than the second element on UB_0. Thus, it belongs to the class 'loan not paid on-time'.

In cases with more than two classes even if information is not enough to find the class, borders can still reduce the number of possible classes for the object, e.g. if the object has less attractive attributes than even one of the elements of the lower border of the class, this object must not belong to this class.

Borders summarize the knowledge presented in the training data set. The same object can be presented in upper as well as in the lower border of the class if its attribute values are incomparable with any of the other objects in the class (it has some values higher and some lower when compared to any other object in the class). If there are inconsistencies in the training data set (see Section 4.1), the same object may be within borders of more than one class (inconsistencies are not resolved by using borders).

The notion of borders is useful. On one hand, in some cases the subset of objects from each class can be analyzed for rules (instead of the whole data set) if the model *takes into account ordinal scales*. On the other hand, these borders present rules for classification by themselves, as any object (not from the training data set) whose attribute values are between lower and upper borders of the class will be classified as belonging to this class.

Using our ordinal model for the car evaluation example we found the following. First, the data set was representative enough to classify all possible combinations of attributes, but five cases (assigned to two possible classes out of four). We were able to form borders for each class and resolve inconsistencies by reclassifying 18 cases.

5. Discussion and conclusion

Knowledge discovery is a complicated process of extracting useful information from data. It includes many steps such as data warehousing; target data selection; data cleaning, preprocessing, and transformation; model development and choosing suitable data mining algorithms; evaluation and interpretation of results; using and maintaining the discovered knowledge.

In a majority of the real cases, the knowledge discovery process is iterative and interactive in nature. Results obtained at any step of the process may stimulate changes at earlier steps. At the core of the knowledge discovery process are the data mining methods for extracting patterns from data. These methods can have different goals and may be applied successively to achieve the desired result in the knowledge discovery process (any method that can help in obtaining more information from data is useful). Although the ultimate goal of knowledge discovery is an automated

knowledge discovery, the majority of available tools today are designed for expert analysts. They work with initial data providing the right data for the right analysis. They analyze interim results at each step of the process and re-adjust data, models, and techniques as necessary.

The understanding that some attributes possess ordinal qualities regarding categorical classes may significantly improve the results of the analysis in classification tasks. The data presented in Section 4 shows that just stating these ordinal dependencies in the traditionally used analytical tools may lead to a better predictive model. It usually leads to a reduced number of rules and nodes with simultaneous reduction in error rate.

The car evaluation data set proved to be highly representative and consistent. The primary gain of using ordinal information in this case was in reducing the number of final rules with a lower error rate. In other tasks with less adequate information ordinal analysis may show the chances for the stable and reliable outcome as was illustrated in the loan application example. This data represented an example of a very inconsistent and incomplete set of data even though the dimensionality of the problem was relatively small. It is important to evaluate this factor before applying a data mining technique as it may require some action, as long as the representativeness of the data is not distorted. Inconsistent data sets can be improved by eliminating instances involving too many contradictions, or by reclassifying some of them, or by reevaluating the description of cases (criteria and scales used in the task). In any case, the information on the quality of the data set makes it possible to understand why this data set is not good enough for the task being considered. The substantial additional information that can be obtained due to ordinal dependences between attribute scales and decision classes is valuable.

Although algorithms for ordinal classification are labor intensive (not meant for very large data sets), they can be used at some stages of data analysis; for example, after reducing the number of attributes with other methods (Saarenvirta, 1999), or after feature construction (Major, 1998). Incorporation of ordinal classification into the data mining classification technique can improve overall results for specific tasks.

References

Berry, M. J. A., & Linoff, G. (1997). *Data mining techniques*, New York: Wiley.

Ben-David, A. (1992). Automated generation of symbolic multiattribute ordinal knowledge-based DSSs: Methodology and applications. *Decision Sciences*, *23* (6), 157–1372.

Ben David, A., Sterling, L. A., & Pao, Y. H. (1989). Learning and classification of monotonic ordinal concepts. *Computational Intelligence*, *5* (1), 45–49.

Bohanec, M., & Raikovic, V. (1990). Expert system for decision making. *Sistemica*, *1* (1), 145–157.

Edelstein, H. (1997). Mining for gold. *Information Week*, 4/21/1997.

Goveia, T. (1999). Short circuiting crime. *Canadian Insurance*, *104* (5), 16–17 http://www.ics.uci.edu/~mlearn/MLSummary.html, http://www.rulequest.com/.

Hong, T. P., Wang, T. T., Wang, S. L., & Chien, B. C. (2000). Learning a coverage set of maximally general fuzzy rules by rough sets. *Expert Systems with Applications*, *19* (2), 97–103.

Kennedy, R. L., Lee, Y., Van Roy, B., Reed, C. D., & Lippman, R. P. (1997). *Solving data mining problems through pattern recognition*, New York: Wiley.

Kiesnoski, K. (1999). Customer relationship management. *Bank Systems and Technology*, *36* (2), 30–34.

Larichev, O. I., & Moshkovich, H. M. (1994). An approach to ordinal classification problems. *International Transactions on Operations Research*, *82*, 503–521.

Larichev, O. I., & Moshkovich, H. M. (1997). *Verbal decision analysis for unstructured problems*, The Netherlands: Kluwer Academic Publishers.

Linoff, G. (1998). Which way to the mine? *As/400 Systems Management*, *26* (1), 42–44.

Lippmann, R. P. (1987). An introduction to computing with neural nets. *IEEE ASSP Magazine*, 4–22.

Loh, W. -Y., & Shih, Y. -S. (1997). Split selection methods for classification trees. *Statistica Sinica*, *7*, 815–840.

Major, R. L. (1998). Using decision trees and feature construction to describe consumer household spending habits. *Proceedings of the 29th Annual Meeting of the DSI (Las Vegas)*, Vol. 2, pp. 627–629.

Mechitov, A. I., Moshkovich, H. M., Olson, D. L., & Killingsworth, B. (1995). Knowledge acquisition tool for case-based reasoning systems. *Expert Systems with Applications*, *9* (2), 201–212.

Mechitov, A. I., Moshkovich, H. M., Bradley, J. H., & Schellenberger, R. K. (1996). An ordinal model for case-based reasoning in a classification task. *International Journal of Expert Systems*, *9* (2), 225–242.

Messier, W. F., & Hansen, J. (1988). Inducing rules for expert systems development: An example using default and bankruptcy data. *Management Science*, *34* (12), 1403–1415.

Michie, D. (1998). Learning concepts from data. *Expert Systems with Applications*, *15* (34), 193–204.

Noordewier, M. O., Towell, G. G., & Shavlik, J. W. (1991). Training knowledge-based neural networks to recognize DNA sequences. *Advances in neural information processing systems*, Vol. 3. San Mateo, CA: Morgan Kaufmann.

Olson, D. L., & Courtney Jr., J. P. (1998). *Decision support models and expert systems*, Houston: Dame Publications.

Pawlak, Z. (1991). *Rough sets: Theoretical aspects of reasoning about data*, Dorchert, The Netherlands: Kluwer Academic Publishers.

Peacock, P. R. (1998). Data mining in marketing: Part I. *Marketing Management*, *6* (4), 8–18.

Quinlan, J. R. (1990). Decision trees and decision making. *IEEE Transactions on SMC*, *20* (2), 339–346.

Quinlan, J. R. (1993). *C4.5: Programs for machine learning*, San Mateo, CA: Morgan Kaufmann.

Saarenvirta, G. (1999). Data mining for direct-mail: A lesson in predictive modeling. *DB2 Magazine*, *4* (1), 40–49.

Slowinski, R. (1995). Rough set approach to decision analysis. *AI Expert*, 19–25.

Yager, R. (1981). A new methodology for ordinal multiobjective decisions based on fuzzy sets. *Decision Sciences*, *12* (4), 589–600.

Yahia, M. E., Mahmod, R., Sulaiman, N., & Ahmad, F. (2000). Rough neural expert systems. *Expert Systems with Applications*, *18* (2), 87–99.

Zadeh, L. A. (1965). Fuzzy sets. *Information and Control*, *8*, 338–353.