Generating the Classification Rule Using Decision Tree Algorithm

Theint Theint Aye, Khin Sandar
University of Computer Studies, Mawlamyine
kayhpaanthu@gmail.com

Abstract

Data mining has been used very frequently to extract hidden information from large databases. The classification rule generation process is based on the decision tree as a classification method where the generated rules are studied. This technique is forms of data analysis that can be used extract models to describe important data class. The main purpose of the classification system is to induce rules and accuracy that describe decision tree. In this paper, there are two mains phases. In the training phase, attributes are analyzed by C4.5 algorithm. By using the training data, this system will construct the model or classifier to generate the form of classification rules. In the testing phase, compare the input test data and the classification rules to obtain the result. This system can apply to implement the classification system for IT Technicians job roles.

1. Introduction

Classification is one of the most frequent decision making tasks performed by human. Decision trees are commonly used in classification systems because they are easy to interpret, accurate, and fast. The branching decision at each node is determined by a certain splitting criterion that generates a minimal tree, meaning that the tree has a minimum number of branches.

Decision-tree learning is one of the most successful learning algorithms, due to its various attractive features: simplicity, comprehensibility, no parameters, and being able to handle mixed-type data. In decision-tree learning, a decision tree is induced from a set of labeled training instances represented by a tuple of attribute values and a class label. Because of the vast search space, decision-tree learning is typically a greedy, top-down and recursive process starting with the entire training data and an empty tree. An attribute that best partitions the training data is chosen as the splitting attribute for the root, and the training data are then partitioned into disjoint subsets satisfying the values of the splitting attribute. For each subset, the algorithm proceeds recursively until all instances in a subset belong to the same class.

Decision tree induction is one of the most common techniques to solve the classification problem [6, 7]. It consists of nodes, branches, leaf nodes, and a root. To classify an instance, one starts at the root and finds the branch corresponding to the value of that attribute observed in the instance. This process is repeated at the sub tree rooted at that branch until a leaf node is reached. The resulting classification is the class label on the leaf. The main objective of a decision tree construction algorithm is to create a tree such that the classification accuracy of the tree. Decision tree induction algorithms operate in two phases, the *Construction phase* and *Pruning phase.* The *construction phase* of decision tree usually results in a complex tree that often overfits the data.

For the aim of classification the objects are mostly described by the values or certain attributes like education, punished, total services etc. In the following, the value of an attribute will also be called a feature. Classification is one of the most frequent decision making tasks performed by human. Decision trees are commonly used in classification systems because they are easy to interpret, accurate, and fast. The branching decision at each node is determined by a certain splitting criterion that generates a minimal tree, meaning that the tree has a minimum number of branches. Decision tree learning is a method for approximating value target functions, in which the learned function is represented by a decision tree [11].

The rest of the paper is organized as follows. Section 2 summarizes the related work. In section 3 we describe necessary background theory. Section 4 presents the classification system design. Finally, we conclude the paper in Section 5.

2. Related work

The research by [10] has used Tough Set theory as a classification approach to analyze student data where the Tosetta toolkit was used to evaluate the

student data to describe different dependencies between the attributes and the student status. The data set used in their experiments is the student data of Suranaree University of technology (SUT) during the academic year 2001-2002.

The research by [8] describes the results of analyzing data from a large collection of the so called concurrent version system (CVS) created by many students working on a small set of identical projects (course assignments) in the second year undergraduate computer science course.

Numerous techniques have been developed to speed up decision-tree learning [6], such as designing a fast tree-growing algorithm, parallelization, and data partitioning. The model of Delavari et al. in [2] is a motivation toward enhancing the proposed analysis model presented in [3] and that is used as a roadmap for the application of data mining in higher educational system.

In this paper, we generate the classification rules using decision algorithm for IT Technicians job roles.

3. Background

In this section, we describe necessary background theory to implement this classification system.

3.1 Classification by Decision Tree

Decision trees are powerful and popular tools for classification and prediction. Decision trees are often used in data mining and classification system because they are easily interpreted, accurate, and fast. One important step in constructing a decision tree is the selection of node attributes so that the tree has a minimum number of branches. Each node in the tree specifies a test of some attribute of the instance, and each branch descending from that node corresponds to one of the possible values for this attribute. An instance is classified by starting at the root node of the tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute. This process is then repeated for the sub tree rooted at the new node. Above these is the principle in building tree. A well known and frequently used over the years is C4.5 [4].

3.1.1 C4.5 Algorithm

The algorithm C4.5 is greedy algorithm that constructs decision trees in a top-down recursive divide-and-conquer manner. C4.5 belongs to the family of decision-tree based algorithms and is an extension to the well known ID3 decision tree induction algorithm [6]. C4.5 is known as a rule post pruning algorithm, because it attempts to avoid overfitting through pruning strategies [8]. It involves the following steps:

1. Build the decision tree from the training data set.
2. Convert the learned tree into an equivalent set of rules by creating one rule for each path from the root node to a leaf node.
3. Prune (generalize) each rule by removing any preconditions that result in improving its estimated accuracy.
4. Sort the pruned rules by their estimated accuracy, and consider them in this sequence when classifying subsequent instances.

C4.5 attempts to avoid the overfitting by building general rules for classification.

Algorithm: Generate _ decision _ tree. Generate a decision tree from the given training data.
Input: The training sample, samples, represented by valued attributes; the set of candidate attributes-list.
Output: A decision tree.
Method:

```
FormTree (T)
1.  ComputeClassFrequency (T)
2.  If OneClass or FewCases
            Return a leaf;
            Create a decision node N;
3.  ForEach Attribute A
            ComputeGain (A);
4.  N.test = Attribute WithBestGain;
5.  if N.test is continuous
            Find Threshold;
6.  ForEach T' in the splitting of T
7.  If T' is empty
            Child of N is a leaf;
    Else
8. Child of N = FormTree (T');
9. ComputeErrors of N;
Return N
```

Figure 1. C4.5 Algorithm

3.2 Attribute Selection Measure or Information Gain Measure

The information gain measure is use to select the test attribute at each node in the tree. Such in

measure is referred to as an attribute selection measure or a measure of the goodness of split. The attribute with the highest information gain (or greatest entropy reduction) is chosen as the test attribute for the current node. This attribute minimizes the information needed to classify the samples in the resulting partitions and reflects the lest randomness or "impurity" in these needed to classify an object and guarantees that a simple (but not necessarily the simplest) tree is found.

Let be a set consisting of s data samples. Suppose the class label attribute has m distinct values defining m distinct classes, C_i (for i= 1,….,m). Let s_i be the number of samples of S in class C_i. The expected information needed to classify a given sample is given by

$$I(s_1, s_2, ..., s_m) = -\sum_{i=1}^{m} p_i \log_2(p_i) \quad (1)$$

Where p_i is the probability that an arbitrary sample belongs to class C_i and is estimated by s_i/ s.

$$E(A) = \sum_{j=1}^{v} \frac{s_{1j}+...+s_{mj}}{s} I(s_{1j}, ..., s_{mj}). \quad (2)$$

From equation 2, the term $\frac{s_{1j}+...+s_{mj}}{s}$ acts as the weight of *j*th subset and is the number of samples in the subset (i.e., having value a_j of A) divided by the total number of samples in S. Note that for a given subset S_j,

$$I(s_{1j}, s_{2j}, ...s_{mj}) = -\sum_{i=1}^{m} p_{ij} \log_2(p_{ij}) \quad (3)$$

Where $p_{ij} = \frac{s_{ij}}{|s_j|}$ and is the probability that a sample in S_j belongs to class C_i from equation 3.

The encoding information that would be gained by branching on A is

$$Gain\ (A) = I(S_1, S_2, ……, S_m) - E(A) \quad (4)$$

The notion of maximum information gain is used in the C4.5 algorithm to determine which attribute to select. Such division in the training data is useless and to avoid this C4.5 uses the Gain Ratio.

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo(A)} \quad (5)$$

SplitInfo (A) is the information due to the split of C on the basis of the value of the categorical attribute A.

$$SplitInfo(A) = -\sum_{i=1}^{v} \frac{|c_i|}{c} \log_2 \frac{|c_i|}{c} \quad (6)$$

The algorithms compute the information gain of each attribute. The attribute with the highest information gain is chosen as the test attribute for the given set S.

4. Classification System Design

In this section, this system can apply to implement the classification system of IT Technicians job roles.

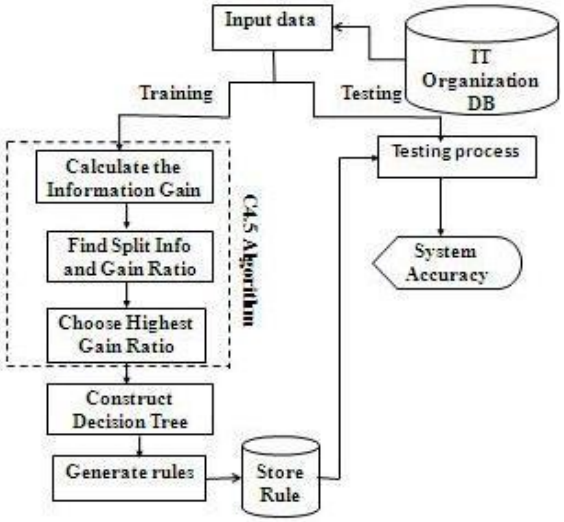

Figure 2. System Flow Diagram

Firstly, this system will be used IT Technicians job roles dataset from the stored database as shown in figure 3.

Training Dataset

New

#	Web Application Server	Software Design	Database	Operating System Platform	Skills	Salary	Experience	Education Level	Class
1.	Web Logic	OLAP	Oracle	Windows	Java	$10000.00	6	Bachelors Degree in Computer Science	Senior Web Developer
2.	Web Logic	UML	Oracle	Windows 2000	J2EE	$10000.00	6	Bachelors Degree in Computer Science	Senior Web Developer
3.	Web Logic	UML	Oracle	Windows	PL/SQL	$10000.00	6	Bachelors Degree in Computer Science	Senior Web Developer
4.	Web Logic	-	Oracle	Windows 2000	Core Java	$10000.00	6	Bachelors Degree in Computer Science	Senior Web Developer
5.	Web Logic	UML	Oracle	Windows XP	JavaScript	$10000.00	6	Bachelors Degree in Computer Science	Senior Web Developer
6.	-	-	ODBC	-	Java	$10000.00	6	Bachelors Degree in Computer Science	Senior Web Developer
7.	Web Logic	UML	ODBC	Windows	J2EE	$10000.00	6	-	Senior Web Developer
8.	Web Logic	UML	SQL Server	Windows XP	C#	$10000.00	-	Bachelors Degree in Computer Science	Senior Web Developer
9.	-	-	ODBC	Windows	Java	$10000.00	7	Bachelors Degree in Computer Science	Senior Web Developer
10.	ISS	-	-	Windows	J2EE	$10000.00	7	Bachelors Degree in Computer Science	Senior Web Developer

Figure 3: sample training data

C4.5 algorithm uses the concept of information gain to make a tree of classificatory decisions with respect to a previously chosen target classification. Information Gain is used to select the test attribute at each node in the tree. Secondly, the dataset is then split on the different attributes. The entropy for each branch is calculated. Then it is added proportionally to get total entropy for the split. The resulting entropy is subtracted from the entropy before the split. The result is the Information Gain, or decrease in entropy. Similarly, we can compute Gain Ratio. The highest gain ratio among the attributes is selected as splitting attribute. In this system highest gain ratio is Software_Design. Thus, Node N is labeled and branches are grown for each of the attribute's value. The tuples are then partitioned to the same class. This process continues until all partition branches have the pure information class. Then, the final decision trees are as followed in Figure 4.

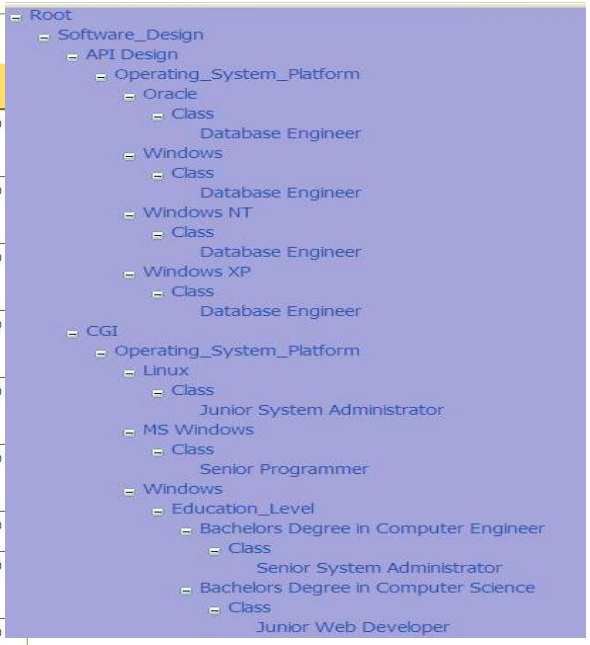

Figure 4. Decision Tree Produced by C4.5 Algorithm

The attribute that yields the largest Information Gain is chosen for the decision node. The notion of maximum information gain is used in the C4.5 algorithm to determine which attribute to select. This system will describes decision tree and extracted rules from the decision tree. And then, rules set will be store in database. In this paper 380 training data set are used and the rule set got 112 rules sets.

Figure.5 shows an example of classification by using decision tree: C4.5 algorithm. Application is the root node from the decision tree, which consists of many decision nodes and leaves. A leaf node specifies a class value (such as Sr. Web Developer, Software Engineer, etc). And then rule can get from the decision tree. And then this system can the query and match this query with decision form rules.

#	Rules
1	IF (Software_Design = 'API Design' AND Operating_System_Platform = 'MS Windows' AND Web_Application_Server = 'ISS') THEN Class = 'Software Engineer'
2	IF (Software_Design = 'API Design' AND Operating_System_Platform = 'Oracle') THEN Class = 'Database Engineer'
3	IF (Software_Design = 'API Design' AND Operating_System_Platform = 'Windows') THEN Class = 'Database Engineer'
4	IF (Software_Design = 'API Design' AND Operating_System_Platform = 'Windows NT') THEN Class = 'Database Engineer'
5	IF (Software_Design = 'API Design' AND Operating_System_Platform = 'Windows XP') THEN Class = 'Database Engineer'
6	IF (Software_Design = 'CGI' AND Operating_System_Platform = 'Linux') THEN Class = 'Junior System Administrator'
7	IF (Software_Design = 'CGI' AND Operating_System_Platform = 'MS Windows') THEN Class = 'Senior Programmer'
8	IF (Software_Design = 'CGI' AND Operating_System_Platform = 'Windows' AND Education_Level = 'Bachelors Degree in Computer Engineer') THEN Class = 'Senior System Administrator'
9	IF (Software_Design = 'CGI' AND Operating_System_Platform = 'Windows' AND Education_Level = 'Bachelors Degree in Computer Science') THEN Class = 'Junior Web Developer'
10	IF (Software_Design = 'CGI' AND Operating_System_Platform = 'Windows' AND Education_Level = 'PhD in Computer Science') THEN Class = 'Senior System Administrator'

Figure 5: Decision tree rules

4.1 IT Jobs Role Dataset

The data were originally collected from IT Technicians job roles dataset from Internet newsgroup by using information extraction such as keyword search method. This data set contains attributes, values, and class labels. This data set contains attributes, values, and class labels. The eight attributes are Web/ Application Server, Software Design, Database, OS Platform, Skills, Salary, Experience and Education Level. The attributes values contain such as Web Logic, ISS, Apache, Tomcat, Web, OLAP, UML, OOAD, Oracle, Windows OS, Java, Core Java, ASP, VB, BS, BE, MS. The target information or class label is IT job role class, examples- Jr. Web Developer, Software Engineer, Programmer, Sr. System Administrator.

Table 1. Attributes and Values of this System

Attribute Name	Values
Web /Application Server	Web Logic, ISS, Tomcat, Apache, Web Sphere, etc.
Software Design	OLAP, UML, OOAD, etc.
Database	Oracle, ODBC, JDBC, etc.
OS Platform	Windows, Windows XP, etc.
Skills	Java, J2EE, J2SE, JSP, etc.
Salary	$10000, $8000, $5000, etc.
Experience	2, 3, 4, above 4 years
Education Level	BS, BE, PhD, MS, etc.

Table.1 shows the attributes used in the computer IT jobs, which consist of: Web/Application Server, Software Design, Database, OS Platform, Skills, Salary, Experience and Education Level.

Finally, enter the user's qualification and this system will check it with extracted rules set from the database and then the system will be determine job role, which roles are suitable with user's qualification, is shown in Figure 6.

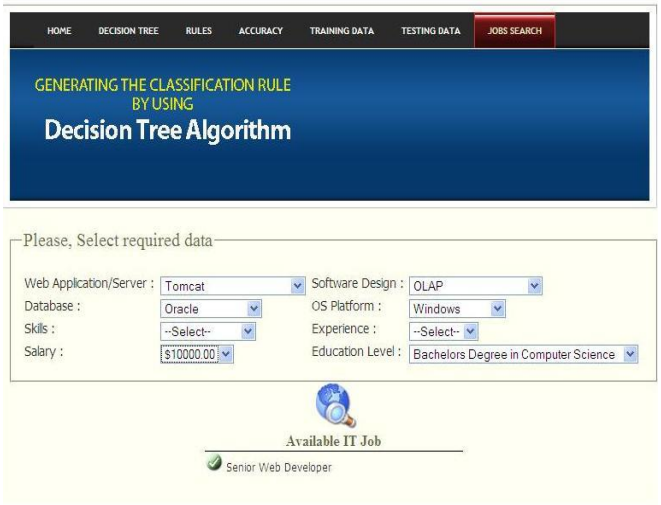

Figure 6. Result of Query matching

5. Conclusion

In this paper, this system implements the user to view the knowledge of data mining. The application is based on the retrieval of IT Technicians job roles data set. The system used C4.5 algorithm to produce rules. This system describes the application of data mining techniques to automated discovery of useful or interesting knowledge from job roles.

In this paper, the system can provide for the users who want to know which job roles are suitable with their qualification. Moreover, this system can support the users to get more precise information without reading the whole job's web sites.

References

[1] Aijun, A "Classification Methods" York University, Canada

[2] D.G., Stork, P.E., Hart and R.O., Duda, Pattern Classification.

[3] Delavari N, Beikzadeh M. R. A New Model for Using Data Mining in Higher Education System, 5th International Conference on Information Technology based Higher Education and Training: ITEHT '04, Istanbul, Turkey, 31 st May-2nd Jun 2004

[4] J.R. Quinlan. *C4.5: Programs for Machine Learning. Morgan Kaufmann, San Mateo,* CA, 1993.

[5] J.R. Quinlan. *Induction of Decision Trees*, Machine Learning, Boston: Kluwer, 1986

[6] L. Owoc Mieczyslaw, Galant Violetta, Gladysz Tomasz, Applying Decision Trees in Classification Tasks

[7] Michael Negnevitsky; Artificial Intelligence A Guide to Intelligence System Second Edition, Addison Wesky, 2005.

[8] Mierle K, Laven K, Roweis G, Mining Student CVS Repositories for Performance Indicators, 2005.

[9] Tom Mitchell; Machine Learning, the McGraw- Hill Companies, Inc, International Edit

[10] Varapron P. et al. Using Rough Set theory for Automatic Data Analysis. 29th Congress on Science and Technology of Thiland. 2003.

[11] Wolfgang Muller and Fritz Wysotzki, Automic construction of decision trees for classification.