Classifying Continuous Data Set by ID3 Algorithm

Kietikul JEARANAITANAKIJ
Department of Computer Engineering
Faculty of Engineering
King Mongkut's Institute of Technology Ladkrabang
Bangkok, Thailand
kjkietik@kmitl.ac.th

Abstract— **This paper presents a modified version of the ID3 algorithm. The goal is to build the decision tree for classifying the continuous data set. An example in the training data set composes of some input features (attributes) and one predicate output. A proper feature ordering produces a shallow decision tree, which spends a logarithm time in classifying a data set. The original ID3 algorithm calculates the information gains of the features and arranges those features by descending order of the information gains. As a result, the decision tree selects a feature which has the biggest information gain at the top level. The algorithm repeats the feature ordering process for the rest of the features until there is not any unclassified example in the training data. However, one problem of the original ID3 algorithm is that it cannot classify the continuous feature in the data set. In order to serve a continuous feature, the ID3 algorithm is modified by quantizing the continuous feature into intervals and performing the classification process within those intervals. The modified algorithm is tested with a standard data set. The experimental results show a relationship between the number of intervals and the error rate on a standard real-world problem.**

Keywords- ID3, classification, continuous feature, information theory , decision tree

I. INTRODUCTION

Decision-tree learning (or decision-tree induction) involves in using a set of training data to generate a decision tree that correctly classifies the training data itself. If the learning process works, this decision tree will then correctly classify new input data as well [1]. The best-known decision tree induction algorithm is the ID3, which was developed by Quinlan [2] in the 1980s. The ID3 algorithm builds a decision tree in a top-down manner. The nodes are selected by choosing features that provide the most information about the data and turning those features into questions. The most important point of the ID3 is how the features are chosen. It is possible to produce a decision tree by selecting the features in an arbitrary order, but this does not necessarily construct the most efficient decision tree. The ID3 finds information gains of all branching features which have not been chosen. Then, a feature which has the greatest value of information gain is selected to be the top of the decision tree. The number of examples in the training set plays an important role on the information value. The more examples we have, the more accuracy of the information value.

The ID3 algorithm finds the smallest-depth decision tree that correctly classifies the training data. A classification process involves in determining the least essential input feature. Another well-known classification algorithm, (i.e., CART [3]), select the most suitable feature at each branching node to grow the decision tree. CART uses node impurity as the criterion for node splitting. A feature node that classifies the data set into equally of both positive and negative outputs is considered being the most impure feature. This feature is brought to the bottom of the decision tree or, in the worst case, should be eliminated.

In a real-world problem domain, there are plenty of input features in the data sets which can lead the ID3 algorithm into a long period of classification process. Therefore, determining the input features which has no contribution to the predicate output is very important. The approach proposed in this paper is the extension of the ID3 classification algorithm. Normally, the ID3 classification algorithm takes input feature, which is discrete and finite, to calculate the information gain. Then, the ID3 generates a decision tree with the optimal depth. However, the classification process cannot directly apply the ID3 to the data set which composes of continuous features. The reason is that a huge number of computations may occur in the information gain calculation. For example, an input feature A contains a real value in the range of (-1, +1). There will be numerous ways to classify a feature A with the value in that range. Therefore, this paper invents a modified version of the ID3 algorithm so that it can calculate an information gain on the continuous data set. The goal is to investigate the variation of the error rate when the degree of the feature quantization is changed.

II. MODIFICATION OF THE ID3 ALGORITHM

The ID3 builds a decision tree from a fixed set of examples. The resulting tree is used to classify the unseen data set. An example contains some features and belongs to an output class. A leaf node of a decision tree is an output class, whereas a non-leaf node is a decision node. In addition, a decision node is a feature branching with each branch is connected to another decision node. The ID3 uses information gain to help in deciding which feature goes into a decision node. The advantage of learning a decision tree is that a program can extract knowledge from an expert.

The main point in the ID3 algorithm is selecting which feature to test at each node in the tree [4, 5]. The algorithm selects the feature that is the most useful for classifying examples and moves it to the upper part of the decision tree. One suitable measure of usefulness is the expected amount of information provided by the feature, where this paper uses the

term in the mathematical sense first defined by Shannon and Weaver [6]. Information theory measures information content in bits. One bit of information is adequate to answer a yes/no question. Given a definition of a statistical property called information gain that measures how well a given feature separates the training examples according to their target classification. The one with the highest information (information being the most useful for classification) is selected. In order to define information gain precisely, it is necessary to define a measure commonly used in information theory called entropy that characterizes the (im)purity of an arbitrary collection of examples. Given a collection S, containing examples with each of the C outcomes, the entropy of S is

$$Entropy\ (S) = \sum_{I \in C} [-\ p(I) \log_2 p(I)], \qquad (1)$$

where $p(I)$ is the proportion of S belonging to class I. Note that S is not a feature but an entire set of examples. Entropy is 0 if all members of S belong to the same class. The range of entropy is 0 ("purity") to 1 ("impurity.") The next measure is an information gain that used to measure the expected reduction in entropy. For a particular feature A, $Gain(S, A)$ means the information gain of sample set S on the feature A and is defined by the following equation:

$$Gain(S, A) = Entropy(S) - \sum_{v \in A} [(|\ S_v\ | / |\ S\ |).Entropy(S_v)], \quad (2)$$

where:

Σ is a summation on each possible value v of feature A,

S_v = subset of S for which feature A has value v,

$|S_v|$ = the number of elements in S_v,

$|S|$ = the number of elements in S.

The ID3 searches through all features of training instances and extracts the feature that best separates the given examples. If the feature perfectly classifies the training sets, the ID3 will stop; otherwise, it will recursively operate. The training examples associated with each leaf node all have zero value in their entropy. The algorithm uses a greedy search. It picks the best feature and never looks back to reconsider the early choices.

From (2), if possible values (v) of feature A are uncountable, for example, feature A has real values in the range of (-1, 1), there will be a huge number of computations taking place in the summation. To avoid this problem, the modified ID3 quantizes a range of value of feature A into k intervals, where k is a constant depending on the range of a particular feature. Therefore, the following condition must be added.

$$|v| = k. \qquad (3)$$

As a result, the possible values of feature A now become countable and the computation of $Gain(S, A)$ can be performed within a restricted time. A group of examples, which have continuous features, will fall into an appropriate interval. The number of intervals can be counted as the possible values of a feature. An assumption is established in that the error rate of the decision tree should depend on the variation of the value k. Moreover, the decision tree may confront with the overfitting problem when the value k increases. The experimental results in the next section support the assumption.

III. Experimental Study

In this section, the experimental results tested under a well-known standard benchmark (i.e., Pima Indian diabetes data set) are described. The purpose of the experiment is to focus on the relationship between the value of k in (3) and the performance of the decision tree. The performance of the decision tree is evaluated in terms of the testing error rate. The data set is originally from the UCI repository of Machine Learning databases and Domain Theories. Although the data set is designed as a benchmark for the sophisticate learning platforms, for example, neural network and genetic algorithm, this paper intends on using it to study the impact of the value k to the testing error rate when the modified ID3 is tested under a real-world data. Therefore, there is no expectation that the error rate of the decision tree learning will be as low as those of the high-level learning platforms.

A. Data Description

The data set used by the ID3 has certain requirements, which are:

1) Data Set Characteristics

- Feature description - the feature must describe each example and fall into a specific interval.
- Predefined classes - an example's features must already be defined, that is, they are not learned by the ID3.
- Discrete classes - classes must be sharply delineated. Continuous classes broken up into vague categories such as a metal being "hard, quite hard, flexible, soft, quite soft" are suspect.
- Sufficient examples - since inductive generalization is used, there must be enough test cases to distinguish valid patterns from chance occurrences.

2) Pima Indian Diabetes Data Set

The data set has been collected by National Institute of Diabetes and Digestive and Kidney Diseases has been used in many published papers. It contains 768 examples. Each example consists of eight-element real-value features; 1) number of times pregnant, 2) plasma glucose concentration a 2 hours in an oral glucose tolerance test, 3) diastolic blood pressure (mm Hg), 4) triceps skin fold thickness (mm), 5) 2-Hour serum insulin (mu U/ml), 6) body mass index (weight in kg/(height in m)2), 7) Diabetes pedigree function, and 8) age (years.) The objective of this data set is to diagnose a binary-valued variable investigated whether the patient shows signs of diabetes according to the World Health Organization criteria.

B. *Experimental Procedure*

The total number of patterns or examples is randomly divided into two sets. The first 75% of a data set is selected as the training set and the second 25% is the test set. The test set completely remains unseen to the decision tree during training. The size of each data set used in the experiment is given by: 576 examples are used as the training set and 192 examples are used as the test set. Data distribution is 500 examples in output class zero and 268 examples in output class one. The minimum and maximum values of eight features are listed in Table 1. The representations of features one to eight are given as F1 to F8, respectively.

TABLE I. THE DISTRIBUTE ON THE RANGE OF FEATURES

Feature number	Minimum	Maximum
F1	0	17
F2	0.0	199.0
F3	0.0	122.0
F4	0.0	99.0
F5	0.0	846.0
F6	0.0	67.1
F7	0.078	2.42
F8	21	81

C. *Experimental Results*

The experiment is performed on a benchmark problem (i.e., Pima Indian diabetes data set.) For the sake of simplicity, this paper uses the same value of k (a number of intervals of the feature value) for every feature. In fact, the goal is to focus on the relationship between the value of k and the performance of the decision tree. Therefore, the value of k is set up for the experimental purpose. A practical decision tree, indeed, should have a different value of k in each feature. The experiments are performed on different value of k, e.g., 3, 6, 9, 12, 15, 18, and 20, and then compare the outputs. There are 20 iterations on each value of k in order to get the accuracy. Testing error rate of all values of k on the Pima Indian diabetes problem is shown in Fig. 1. Note that an error rate is the ratio of misclassified examples to the total examples of the testing set.

A graph in Fig. 1 illustrates the result that is congruent to the assumption in Section 2. An error rate on a small value of k, e.g., 3, gives a bad error rate of classification. The reason which can clarify this result is that the testing example has limited choices of moving into intervals of a feature. Therefore, providing more alternatives for each feature seems to give a smaller number of errors. For k=6, the error rate decreases approximately in linear order. When the value of k increases to 9, the error rate constantly decreases. As the number of intervals increases, the decision tree has more choices for the samples to fall into their appropriate categories and hence decreases the error rate of the decision tree learning. However, at a specific number of k (k=12), the decision tree seems to get into the optimal state. The error rate of the decision tree, which has a value of k exceeds the best point, does not decrease as in the previous k. As a result, the error rates for value k of 15 and 18 are higher than that of 12.

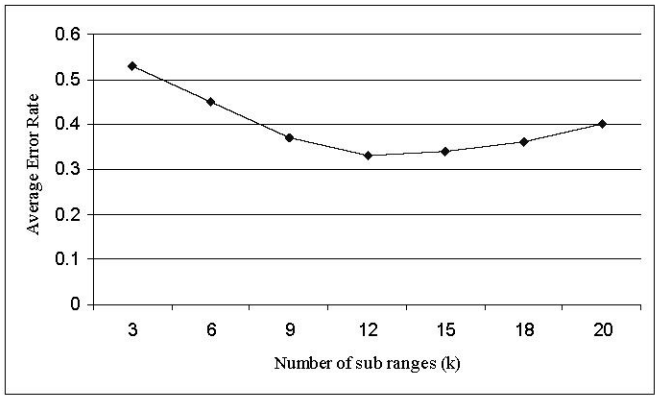

Figure 1. Error rates on different values of k

As the modified ID3 continues increasing the value of k behind the finest point, the error rate gradually increases. The decision tree is tested further, with 20 as the value of k, to confirm the assumption that the error rate will increase for a very large value of k. The result explains the assumption. It indicates that there must be an appropriate value of k for a particular feature. The suitable value of k for this experiment (i.e., the Pima Indian diabetes problem), should be around 12. The reason for the increasing error rate when k exceeds the optimal point is described in the next section.

IV. DISCUSSION

From the result shown in the previous section, the number of intervals, k, on each feature has a high impact on error rate of decision tree learning. The value of k turns into the opposite direction of the error rate. However, this relationship can be hold when the number of intervals does not exceed the optimal point. In addition, increasing too large value of k will lead the decision tree to memorize all the patterns of the training examples. When the value of k reaches the best classification, the decision tree runs into the problem of overfitting. Overfitting usually occurs when there is noise in the training data, or when the training data does not adequately represent the entire space of possible value. In such situation, it is possible for a decision tree to correctly classify all the training data, but to perform less well at classifying unseen data than some other decision trees that perform poorly at classifying the training data. The decision tree, in this case, tries to memories the training data which may contain noises in the data set and then misclassifies the test data. This situation can be described in Fig 2.

The hypothesis separation line in Fig 2 is too overfitting the training data. It draws the hypothesis cover those two noises data (the two white circles at the bottom.) It may perform incredibly well in classifying the training data set, but perform awfully poor in classifying the real and unseen data set.

Therefore, the hypothesis in Fig 2 overfits the training data, and allows itself to be warped by noise in the training data.

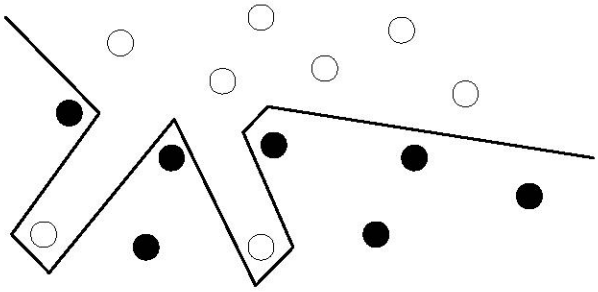

Figure 2. Illustration of overfitting problem

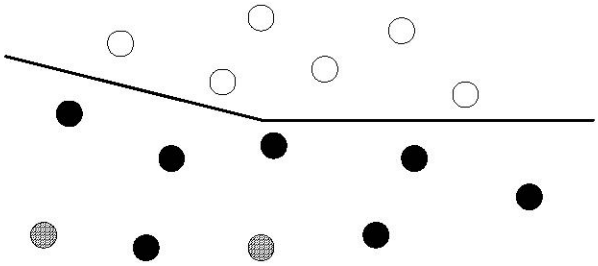

Figure 3. A suitable hypothesis

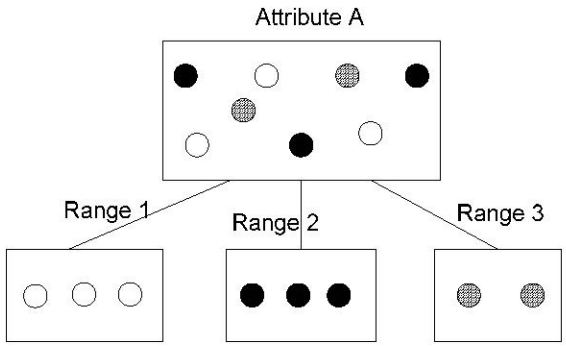

Figure 4. Illustration of overfitting problem when the number intervals increases

A good hypothesis should reasonably and effectively map the full set of data. It may cause some errors, but it reasonably closely represents the trend of the data, as shown in Fig 3. Those two noises at the bottom are considered as distortions in the training data. The hypothesis basically ignores two of them and draws a simple line to separate two groups of training examples. In order to illustrate a more concrete example according to the overfitting problem when the number of k increases, a situation is given in that there are two noises, represented by shaded circles, in the training data as the scenario in Fig 4. Assume that an appropriate value of k for this problem is two. By increasing the value of k, those noises that used to reside in either group 1 or group 2 are split into another new interval which, indeed, gives an ungeneralized hypothesis.

V. Conclusions

This paper has shown a modified version of the ID3 algorithm that classifies the data set which is continuous and bound into a specific range. By dividing a continuous feature into k intervals, the ID3 algorithm will create a decision tree that can work well on the floating point value data set, which cannot be classified in the original ID3. An appropriate value of k depends on the problem type. In addition, the value of k in each feature does not necessarily be the same. A data set expert may wisely pick a different value of k in each range of feature. This is because some features may have a wide range of value while, on the other hand, some have a narrow range. The task of selecting a suitable value of k in each feature should be guided by an expert who thoroughly understands the data set. The investigation in this paper identifies the effect when the value of k increases and describes the overfitting problem, which comes behind the error rate. In many areas of industry and commerce, decision trees are usually the first method tried when a classification method is to be extracted from a data set. One property of decision tree is that it is possible for a human to understand the output of the learning algorithm. However, the ID3 (or any inductive algorithm) may misclassify the data.

Acknowledgement

Thanks to Associate Professor Dr. Ouen Pinngern for valuable comments and suggestions.

References

[1] Ben Coppin, *Artificial Intelligence Illuminated,* Jones and Bartlett Publishers, 2004.

[2] J. R. Quinlan, "Induction of Decision Trees," *Machine Learning*, vol. 1, 1986, pp. 81-106.

[3] L. Breiman, J. H. Friedman, R. A. Olshen, and C.J. Stone, *Classification and Regression Trees,* Belmont, CA: Wadsworth and Brooks, 1984.

[4] Andrew Colin, "Building Decision Trees with the ID3 Algorithm," *Dr. Dobbs Journal*, June 1996.

[5] Paul E. Utgoff, *Incremental Induction of Decision Trees,* Kluwer Academic Publishers, 1989.

[6] Shannon, C. E. and Weaver, W., *The Mathematical Theory of Communication*, University of Illinois Press, Urbana, Illinois, 1949.

[7] Tom Mitchell , *Machine Learning*, McGraw-Hill, pp. 52-81, 1997.