Query-Based Learning Decision Tree and its Applications in Data Mining

Ray-I Chang+, Chia-Yen Lo+, Wen-De Su*, Jen-Chieh Wang*

+Department of Engineering Science, National Taiwan University
Taipei, Taiwan, ROC
*Information Management Center, Chung-Shan Institute of Science & Technology
Armaments Bureau, MND
Lung-Tan, Tao-Yuan, Taiwan, ROC

Abstract

Decision tree is one of the most significant classification methods applied in data mining. By its graphic output, users could have an easy way to interpret the decision flow and the mining outcome. However, decision tree is known to be time consuming. It will spend a high computation cost when mining the large scale dataset in the real world. This drawback causes decision tree to be ineligible in processing the time critical applications. In these years, we have introduced the query-based learning (QBL) method to different neural networks for providing a more effective way to learn the large dataset. These neural networks have achieved good clustering and classification results. In this paper, a novel mining scheme called QBLDT (query-based learning decision tree) is proposed to apply the QBL concept in decision tree construction. Experimental results show our proposed method is better than the traditional decision tree in different performance metrics. It makes learning quicker and can achieve better prediction results.

*__Keywords__: Query-Based Learning, Decision Tree, Data Mining.

1. Introduction

Data mining has been widely utilized in many fields, such as in business, medicine, industry, *etc*. In past years, different data mining methods have been proposed. For classification, it has decision tree, neural networks, K-nearest neighbors, *etc*. For clustering, it has K-means clustering, squared error clustering, *etc* [1]. Among these methods, decision tree is mostly chosen because its processing procedure is in graphical flow. It has an easy way to interpret the mining outcome. However, decision tree is known to be time consuming. It needs a lot of time when dealing with large scale real world problem. In this paper, a novel mining scheme called QBLDT (query-based learning decision tree) is proposed to solve this problem. In the early studies of machine learning, Oates [3] suggested that we can apply partially representative training data to achieve certain degree of correctness, and also save time. In our previous researches [4], we have applied query-based learning (QBL) concept to different neural networks. It assumes there is an oracle in the learning loop. The oracle can actively and repeatedly add training samples for better training [2]. Our experiments have achieved good clustering and classification results. It provides a more efficient way to deal with large dataset.

In this paper, we apply QBL concept to improve the decision tree construction. Previous QBL methods usually take "conjugate points" as their learning instances. However, in real world, the conjugate points may be difficult to get and even non-existed. In this paper, we try to examine different sampling method to help decision tree find the better learning points. Our proposed method, called QBLDT (query-based learning decision tree), is better than the traditional decision tree in different performance metrics. It makes learning quicker and can achieve better mining results.

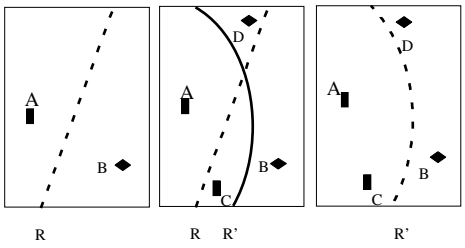
2. Related Works

There are many mining tools in data mining applications and the one we use most is decision tree. A decision tree is a logical model represented as a multi-split tree that shows how the value of a target variable can be predicted by using the values of a set of predictor variables. It uses very often in decision making because of its graphical output, so people can easily understand its flow. There are several splitting algorithms (information gain, GINI, gain ratio, *etc.*) and pruning methods to enhance decision tree's ability.

Figure 1 shows a simple example of the learning process of QBL. Input the initial dataset which contains samples *A* and *B* in two different classes. Following a traditional training algorithm (for example, the back-propagation algorithm), we can obtain a classification boundary *R*. As shown in Figure 1(a), it introduces a dotted line to separate the whole sample space into two parts. Assume that *R'* is the real boundary. According to QBL, we can use the inversion algorithm to select some additional unclear

* *This paper is partially supported by NSC 94-2416-H-002 -015- and 95-2623-7-002-004-D.*

data points that near the boundary for finding a better classification result. As shown in Figure 1(b), we take two points *C* and *D* without knowing their classes. Then, we use "oracle" to give these two additional samples correct learning instances. Obviously, *R* is not a good boundary to correctly classify these four samples. Continuously applying the QBL method, we can fix the classification boundary gradually as shown in Figure 1(c) and find the real boundary finally.



(a) (b) (c)

Figure 1. Learning process of QBL.

Theoretically, we have to take "conjugate points" as learning instances. They are located with the same small distance to the classification boundary but in different side. The intention of selecting conjugate points is to prevent learning instances from bias. But, in real world, the conjugate points may hard to get and even non-existed. We have to use the sampling method to find data points those are near recognized boundary and are also existed in database for learning. In this paper, we will try to use different sampling method to help decision tree to find the better learning points. The features of simple random sampling and stratified random sampling [12] are illustrated as follows.

Simple random sampling:

1. The population consists of *N* objects.
2. The sample set consists of *n* objects.
3. All the samples are randomly picked from the population.

Stratified random sampling:

1. The population consists of *N* objects.
2. Using a certain criterion to divide the population into *x* levels.
3. For each level, randomly pick a suitable number of samples to let the sample set be *n* objects.

A good criterion to divide the population in stratified random sampling should have the property: "*Within the level its homogeneity should be bigger, but between the different levels needs its heterology bigger*".

3. Proposed Methods

In this paper, the target of query is to sample suitable points near the delaminated boundary. Then use those points to induce the real boundary. It can be viewed as a combination of stratified sampling and judgment sampling. As judgment sampling is difficult in decision tree, we use the most common sampling method: simple random sampling and stratified random sampling to carry out a simple QBL in decision tree. We use information gain to be the criterion to divide population in stratified random sampling. After calculating information gain, we choose the attribute with biggest value to become the criterion of node splitting and then build up decision tree. Table 1 shows the operating processes of a simple QBLDT algorithm (that considers only the initial construction of decision tree) with either simple random sampling or stratified random sampling.

Table 1. A simple QBLDT algorithm that considers only the initial construction of decision tree.

Step1. Total data set is D_{all}.
--[Simple Random Sampling]
Randomly choose *x*% data from D_{all} as the training set D_{tr}, and others be the testing data set D_{te}.
--[Stratified Random Sampling]
Estimate attribute A_i with the highest information gain. Then randomly takes *x*% from A_i as the training set D_{tr}. If A_i has *m* value, each of one will occupy (*x*/*m*)%. Others as the testing data set D_{te}.
Step 2. Use D_{tr} to construct decision tree by ID3 algorithm. (We don't restrict the depth of tree and use no pruning method.)
Step 3. Test the decision tree by D_{te}.

Then, we propose an iterative version of QBLDT algorithm that continuously applies the QBL concept to gradually find a better classification boundary. At the first step, we choose an appropriate subset of input data as the training set to construct the initial decision tree, just like traditional one. Then, we test the current decision tree to decide whether adds new data or not. A new decision tree is constructed for the next verification if some new data are added into the training set. The flow of this iterative construction process is shown in Figure 2.

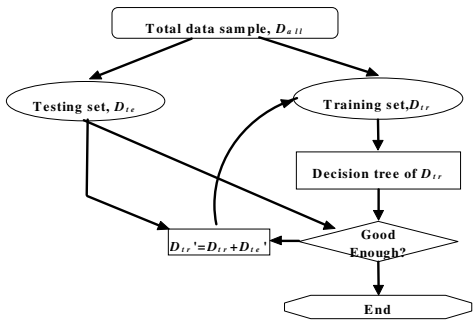


Figure 2. Iterative construction process of QBLDT.

The detail processing steps are illustrated in Table 2. During this procedure, Step 2 can repeat *k* times until the stop criterion is met. Usually, the stop criterion

can be the pre-specified classification result or iteration number.

Table 2. An iterative version of QBLDT algorithm.

Step 1. Total data set is D_{all}. Choose *x*% data from D_{all} as training set D_{tr} (by simple random sampling or stratified random sampling), and others as testing data set D_{te}.
Step 2. Use D_{tr} to construct decision tree by ID3 algorithm.
Step 3. Test the decision tree by D_{te}. Calculate "correct prediction", "no prediction", and "incorrect prediction" data in D_{te}.
Step 4. **IF** the stop criterion is met **THEN stop** **ELSE** choosing *n* no prediction points from D_{te} and making it as a new training data set D_{tr}'; go to Step 2.

According to QBL, we need to choose some meaningful extra learning data to strengthen our decision ability. Notably, after doing the classification, test data will obtain one of these three kinds of results:

"correct prediction, CP" (assigned class is correct),
"no prediction, NP" (can't be assigned to any class),
"incorrect prediction, IP" (assigned class is incorrect).

As shown in Figure 3, each sample point gets a distance from the boundary. Comparing with the no prediction points (triangles), the incorrect prediction points (circles) may have a longer distance. (Based on the QBL concept, we can remove correct prediction points from our consideration.)

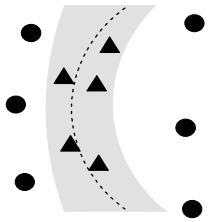


Figure 3. Different Learning points.

Previous experiments show that training QBL neural networks by points those far away from the boundary may cause a worse classification result. If we enforce decision tree to learn those incorrect prediction points, the boundary may locate in wrong place for classification. On the contrary, learning from no prediction points can let boundary make fine tuning. Thus, we may get a better boundary. In this paper, we use "no prediction" points (and their conjugate points, if possible) as extra learning data. We also use experiments to prove this heuristic rule.

4. Experimental Results

We use "Mushroom" and "Nursery" in UCI Machine Repository [5] as our experimental data set. Their attributes are shown in Table 3. In this paper, we let the training set occupy 5% (*x*=5) of the total data set. Rest of it becomes the testing set.

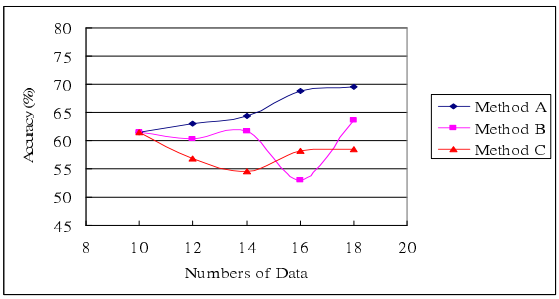
Table 3. Experimental data set.

Dataset	"Total" dataset number	"fold" data number
Nursery	12960 (8 attributes)	200 (60 folds)
Mushroom	8124 (22 attributes)	200 (40 folds)

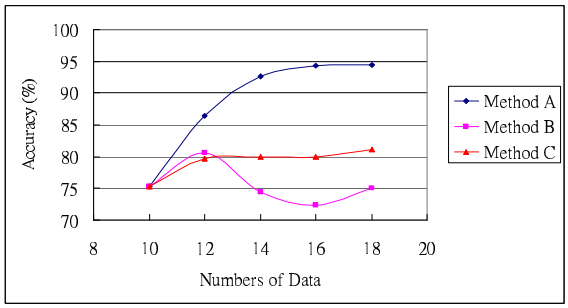
Table 4. Test methods.

	Queried Data Points		
	NP	IP	Random
Simple Random Sampling	A	B	C
Stratified Random Sampling	D		

For easily reading, we make a table to show our proposed methods. As shown in Table 4, its row is the sampling method applied and its column is the source of our queried data points to show where we get new learning data. For example, in method A, we use simple random sampling to get the initial training dataset and the queried data points are from no prediction class. Method C also uses simple random sampling, but the queried data points are selected randomly. As both the initial dataset and the queried data points are obtained by simple random sampling, method C is called SRSDT (Simple Random Sampling Decision Tree) in this paper.



(a)



(b)

Figure 4. Comparing the classification result of decision tree methods that use the same simple random sampling but with different queried data points. Datasets: (a) Nursery. (b) Mushroom.

There are two parameters of QBLDT. One is the number of data chosen for new testing data (D_{te}'), said *n*. The other is the iteration number *k*. We set *n*

= 2 and k = 10, and use all dataset in experiments. In this paper, we first compare the classification result of decision tree methods that use the same simple random sampling but with different queried data points. The obtained results show that method A is better than method C (SRSDT), and the curve tendency is up as shown in Figure 4.

Comparing with method B (with incorrect prediction points), method A (with no prediction points) obviously gets a better classification result in learning. Learning by incorrect prediction points is sometimes better than by random sampling from a certain point of view. Note that, learning by incorrect prediction points makes the growth curve bounced, and even get a worse classification result in some situations. According to our experiments, we decide to choose the extra learning set from no prediction points instead of incorrect prediction points.

For further experiment, we will test if using stratified random sampling can get better classification results. According the previous experiments, learning from no prediction is the best policy. Therefore, we don't test stratified random sampling with incorrect prediction or random points. Figure 5 shows the results.

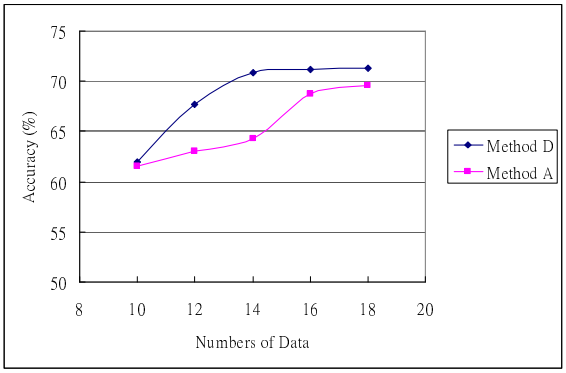


Figure 5. Different query strategy classification results in Nursery dataset.

5. Conclusion

In this paper, we use simple random sampling and stratified random sampling as criterion of sampling suitable data points near the delaminated boundary. Stratified random sampling really has better result in choosing points. Then, we propose QBLDT to strengthen primal decision tree's classification ability with query-based learning. QBLDT gets better prediction outcome than SRSDT.

The initial decision tree may not have a good classification result. But with more queried data added in, curve grows up and shows a better classification result in some cases. However, as the training dataset is just a small subset of the total dataset, we may select different attributes in constructing the initial decision trees. As shown in Figure 6, they imply different results at beginning.

After our calculation information gain of total data in Nursery dataset, we find attribute *health* with the highest information gain in it. So if initial sampling does not find this attribute as delaminated attribute, the effect of stratified will decrease the correctness of classification.

This result motivates us to find a better attribute in our future works. If we choose a better attribute at initial query, the correctness of classification will be higher and also have a stable curve growing tendency.

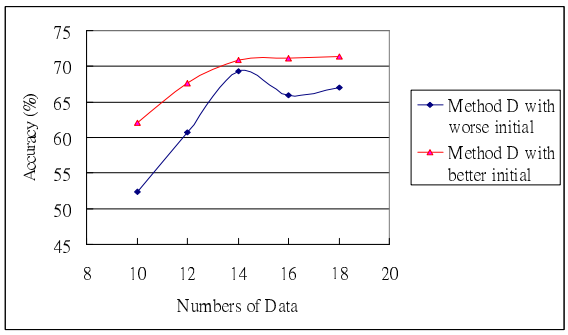


Figure 6. Different initial querying strategy in Nursery dataset.

References

[1] M.H. Dunham, *Data Mining – Introductory and Advanced Topics*, Prentice Hill, 2002.

[2] E.W. Saad, J.J. Choi, J.L. Vian, and D.C. Wunsch, "Query-Based Learning for Aerospace Application," *IEEE Trans. on Neural Networks, vol. 14, no. 6*, pp.1437-1448, 2003.

[3] T. Oates, D. Jensen, "The Effects of Training Set Size on Decision Tree Complexity," *The Fourteenth International Conference on Machine Learning*, pp.254-262, 1997.

[4] L.B. Lai, R.I. Chang, and J.S. Kouh, "Mining Data by Query-Based Error-Propagation," LNCS, Vol. 3610, pp. 1224-1233, 2005.

[5] http://www.ics.uci.edu/~mlearn/MLRepository.htm (UCI Machine Learning Repository)

[6] CI Space, *Decision Trees ver 4.0.1,* http://www.cs.ubc.ca/labs/lci/CIspace/Version4/dTree/

[7] R. Musick, J. Catlett, S.J. Russell, "Decision Theoretic Subsampling for Induction on Large Databases," *International Conference on Machine Learning (San Mateo, CA)*, Morgan Kaufmann, pp. 212-219, 1993.

[8] Selby, R.W. Porter, "Learning from examples-generation and evaluation of decision trees for software resource analysis," *IEEE Trans. on Software Engineering*, pp.1743-1757, 2000.

[9] R.I. Chang, P.Y. Hsiao, "Unsupervised query-based learning of neural networks using selective-attention and self-regulation," *IEEE Trans. on Neural Networks*, vol.8, no.2, pp.205-217, March 1997.

[10] M. Zorman, V. Podgorelec, P. Kokol, M.Peterson, J.Lane, "Decision tree's induction strategies evaluated on a hard real world problem," *IEEE Symposium on Computer-Based Medical Systems*, pp.19-24, 2000.

[11] DTREG, http://www.dtreg.com/dtintro.htm

[12] Statistics Tutorial, http://www.stattrek.com/Lesson3/SamplingTheory.aspx