의사결정트리의 분류 정확도 향상

메하리 마르타 레제네*, 박상현*†
연세대학교 컴퓨터공학과
*e-mail : marta_rezene@yahoo.co.uk
† 교신저자

Classification Accuracy Improvement for Decision Tree

Mehari Marta Rezene*, Sanghyun Park*†
Dept. of Computer Science, Yonsei University
† Corresponding Author

Abstract

Data quality is the main issue in the classification problems; generally, the presence of noisy instances in the training dataset will not lead to robust classification performance. Such instances may cause the generated decision tree to suffer from over-fitting and its accuracy may decrease. Decision trees are useful, efficient, and commonly used for solving various real world classification problems in data mining. In this paper, we introduce a preprocessing technique to improve the classification accuracy rates of the C4.5 decision tree algorithm. In the proposed preprocessing method, we applied the naive Bayes classifier to remove the noisy instances from the training dataset. We applied our proposed method to a real e-commerce sales dataset to test the performance of the proposed algorithm against the existing C4.5 decision tree classifier. As the experimental results, the proposed method improved the classification accuracy by 8.5% and 14.32% using training dataset and 10-fold cross-validation, respectively.

Keywords—Preprocessing; Classification Accuracy; Naive Bayes Classifier; Decision Tree

1. Introduction

Classification is a data mining function that describes and distinguishes data classes or concepts. The goal of classification is to predict accurate class labels of instances whose attribute values are known, but whose class values are unknown.

Real-world data tend to be dirty, incomplete, and inconsistent. To build good model for classification algorithms, we should first preprocess the data. Data preprocessing, a critical initial step in data mining work, is often used to improve the quality of training data set, thereby helping to improve the accuracy and efficiency of the subsequent mining process. To do so data cleaning and preparation is the core task of data mining which is dependent on chosen software and algorithms [1].

This paper presents a preprocessing technique for scaling up the decision tree classifier accuracy. We applied naive Bayes classifier on the training dataset to remove troublesome instances. This method able to automatically extract the most valuable training datasets from noisy complex training data before constructing the learning tree for decision making. It will also reduce the risks associated with over-fitting. As good quality data helps the mining purpose to get good performance, this proposed preprocessing technique helps to get better quality training dataset and impressive accuracy results by removing noisy contradictory instances.

The rest of the paper is organized as follows: Section 2 introduces existing works related to decision tree. Section 3 and 4 present our proposed algorithm and the experimental results, respectively. Finally, the conclusion is drawn in Section 5.

2. Related Works

Decision tree is a fast and efficient algorithm of data mining in classification and prediction. After the first decision tree algorithm was proposed [2], numerous algorithms have been proposed by different researchers to improve the classification accuracy of the decision tree. Some of the classic examples of decision tree algorithms are ID3, CART, C4.5, C5.0, CHAID, and SPRINT. C4.5 is based on the improvement of ID3.

In [3], the proposed decision tree algorithm is based on sample selection to improve the classification accuracy and to find the best training dataset. In sample selection, they used iteration process to find the best training set. Using accuracy of the selected sample training as iteration, Information is highly optimized. In [4], the proposed algorithm avoids the over-fitting and complexity problems suffered in the construction of decision trees. This algorithm combines the attribute selection and data sampling processes without the pruning phase.

To overcome over-fitting problem, a post-pruning decision tree algorithm was proposed based on Bayesian theory [5]. Each branch of the decision tree generated by the C4.5 algorithm was validated by Bayesian theorem, and then those branches that do not meet the conditions will be removed from the decision tree. The Bayesian theory puts a post-pruning technique which utilizes two types of verification strategies; i.e., necessity, and sufficiency. This theory was applied to enhanced the accuracy of the given decision tree [6]. Tsallis Entropy Information Metric (TEIM) algorithm was proposed with a new split criterion and new construction method of decision tree [7]. The new split criterion is based on two terms of Tsallis conditional entropy, which is better than conventional split criteria, whereas the new construction method is based on a two-stage approach that avoids local optimum to a certain extent. By combining all the strengths of Tsallis entropy, outstanding performance in accuracy and complexity was achieved by this novel decision tree algorithm, TEIM algorithm.

3. Proposed Algorithm

In this section, we present a naive Bayes classifier and the proposed algorithm.

3.1 Naive Bayes Classifier

A naive Bayes classifier is a simple probabilistic based method, which can predict class membership probabilities [8]. It is easy to use and requires only one scan of the training data for probability generation. It can also handle missing attribute values easily by simply omitting the corresponding probabilities for those attributes when calculating the likelihood of membership for each class. The naive Bayes classifier also requires class conditional independence; i.e., the effect of an attribute on a given class is independent of those of other attributes.

For a given training dataset, $T = \{D_1, D_2, \ldots D_n\}$, each data record is represented as, $D_i = \{d_1, d_2, \ldots d_n\}$. D contains the following attributes, $\{A_1, A_2, \ldots A_n\}$ and each attribute A_i contains the following attribute values, $\{A_{i1}, A_{i2}, \ldots A_{im}\}$. The attribute values can be discrete or continuous. T also contains a set of classes, $C = \{C_1, C_2, \ldots C_z\}$. Each training instance, $D \in T$, has a particular class label C_i. For a test instance, D, the classifier will predict that D belongs to the class with the highest posterior probability, conditioned on D. That is, the naive Bayes classifier predicts that the instance D belongs to the class, C_i, if and only if $P(C_i|D) > P(C_j|D)$ for $1 \le j \le z$, $j \neq i$. The class C_i for which $P(C_i|D)$ is maximized is called the Maximum Posterior Hypothesis and is calculated as in (1). In Bayes theorem, as shown in (1), P(D) is a constant for all classes and therefore, only $P(D|C_i)P(C_i)$ needs to be maximized.

$$P(C_i|D) = \frac{P(D|C_i)P(C_i)}{P(D)} \quad \ldots\ldots\ldots\ldots\ldots\ldots(1)$$

where: P(C|D) - the posterior probability of class (target) given predictor (attribute).

P(C) - the prior probability of class.

P(D|C) - the likelihood which is the probability of predictor given class.

P(D) - the prior probability of predictor.

3.2 Preprocessing Algorithm

We proposed a preprocessing technique to find higher quality data to apply to the C4.5 decision tree classifier. The proposed algorithm is based on the naive Bayes classifier in order to remove noisy instances from the training dataset.

For the training dataset, T, we first applied a basic naive Bayes classifier to classify each training instance, $d_i \in T$. We calculated the prior probability, $P(C_i)$, for each class, $C_i \in T$ and the class conditional probability, $P(A_{ij}|C_i)$, for each attribute value in D. We did the same even if the attribute value was numeric. Then we classified each training instance, $d_i \in T$, using these probabilities. The class, C_i, with the highest posterior probability, $P(C_i|d_i)$, was selected as the final classification for the instance, x_i. Then we removed all the misclassified training instances from the dataset T. In our experiments, these misclassified instances tended to be the troublesome training instances. The presence of such noisy training instances is more likely to cause a decision tree classifier to cause over-fitting, and thus decrease its accuracy.

After removing those misclassified instances from the training dataset, T, we subsequently built a decision tree for decision making using the updated, purely noise free, training dataset T. We present the proposed algorithm as follows:

Preprocessing Algorithm

Input: Training dataset (T) = $\{d_1, d_2, \ldots, d_n\}$

Output: Training dataset with correctly classified instances

Method:

```
1: For each class, $C_i \in T$,
2: Calculate prior probabilities, $P(C_i)$.
3: End For
4: For each attribute value, $A_{ij} \in T$,
5: Calculate class conditional probabilities, $P(A_{ij}/C_i)$.
6: End For
7: For each training instance, $d_i \in T$,
8: Calculate posterior probability, $P(C_i/d_i)$
9: If di is misclassified,
10: Remove $d_i$ from T;
11: End If
12: End For
```

4. Experiments

4.1 Data Source

In this paper, the experimental data was collected from MDM Global (http://www.mdmglobal.co), online sports apparel retailer in Asia. There were 49,871 sales records collected over a period of 27 days from August 1, 2014 to August 27, 2014, inclusive. The data represented in a sequence of time-stamped points.

4.2 Experimental Setup

The experiments were conducted using Windows 7, 64-bit OS, with a 2.5GHz Dual Intel Core i5 Processor and 8GB of RAM. The experiments were performed using Weka 3.8.0, which is an open source data mining software [9] for building models, evaluation, and analysis of the classifier models. Microsoft Excel was used for data preparation, preprocessing and analysis tasks. The proposed preprocessing algorithm was implemented in Java, using Eclipse Java EE IDE 4.5.2 and experiment results were plotted using SigmaPlot 10.0 [10].

To test the proposed algorithm, we used classification accuracy as in (2) and 10-fold cross validation.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad \text{......................(2)}$$

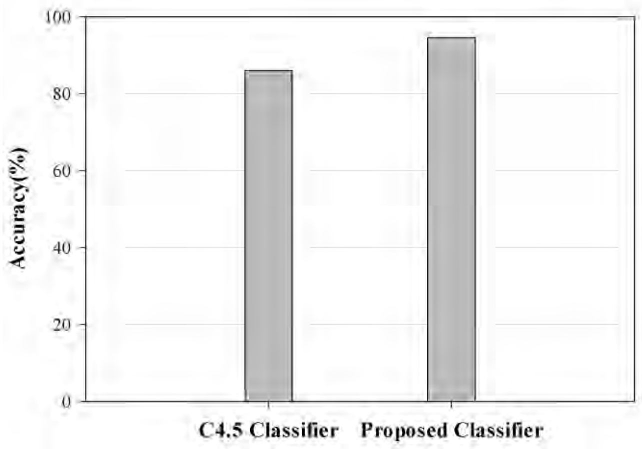
where TP, TN, FP and FN denote True Positives, True Negatives, False Positives and False Negatives, respectively.

The 10-fold cross-validation breaks data into 10 sets of sizes N/10. It trains the classifier on 9 datasets and tests it using the remaining one dataset. This process repeats 10 times and a mean accuracy rate is calculated.

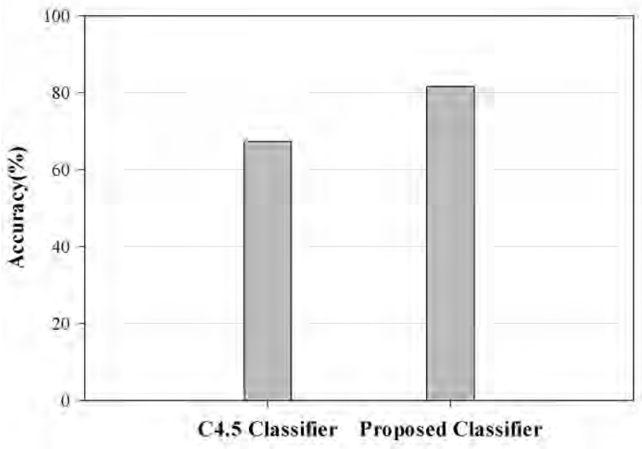
4.3 Results and Discussion

According to the experimental results, the proposed preprocessing technique reduced the size of the training dataset from 49,871 to 35,782 records. The proposed algorithm helps to increase the training dataset quality by removing noisy instances.

We evaluated the performance of the proposed algorithm against the existing C4.5 decision tree classifier based on classification accuracy. Figure 1 shows the classification accuracy for both classifiers based on training dataset. It shows that he proposed decision tree classifier outperformed the existing C4.5 decision tree classifier by 8.5%. Moreover, the result in Figure 2 indicates the proposed decision tree classifier was improved the classification accuracy rate by 14.32% using 10-fold cross validation.



(Figure 1) Classification accuracy with training dataset



(Figure 2) Classification accuracy with 10-fold cross validation

5. Conclusion

In this paper, we have introduced a preprocessing algorithm to improve decision tree classifier performance. The proposed method applied the naive Bayes classifier to remove noisy instances from the training dataset before the decision tree induction. The Experiment evaluation results prove that the efficiency of the proposed decision tree algorithm outperforms the existing C4.5 decision tree classifier. The evaluation results are summarized in Table 1. In future work, we will test more classification performances based on training time, precision, sensitivity, and specificity.

<Table 1> Performance Summary

	Existing C4.5 Decision Tree Classifier	***Proposed Decision Tree Classifier***
Accuracy on Training Dataset (%)	86.02	94.52
Accuracy on 10-fold cross validation (%)	67.26	81.58

Acknowledgment

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP) (NRF-2015R1A2A1A05001845).

References

[1] Galathiya, A. S., A. P. Ganatra, and C. K. Bhensdadia. "Improved Decision Tree Induction Algorithm with Feature Selection, Cross Validation, Model Complexity and Reduced Error Pruning." International Journal of Computer Science and Information Technologies 3.2 (2012): 3427-3431.
[2] Quinlan, J. Ross. "Simplifying decision trees." International journal of man-machine studies 27.3 (1987): 221-234.
[3] Chen, Fucai, Xiaowei Li, and Lixiong Liu. "Improved C4. 5 decision tree algorithm based on sample selection." Software Engineering and Service Science (ICSESS), 2013 4th IEEE International Conference on. IEEE, 2013.
[4] Karabadji, Nour El Islem, et al. "Improved decision tree construction based on attribute selection and data sampling for fault diagnosis in rotating machines." Engineering Applications of Artificial Intelligence 35 (2014): 71-83.
[5] Zhang, Wenchao, and Yafen Li. "A Post-Pruning Decision Tree Algorithm Based on Bayesian." Computational and Information Sciences (ICCIS), 2013 Fifth International Conference on. IEEE, 2013.
[6] Quinlan, J. R. "Induction of Decision Trees, Centre for Advanced Computing Sciences." New South Wales Institute of Technology, Sydney (2007).
[7] Wang, Yisen, Chaobing Song, and Shu-Tao Xia. "Improving decision trees by Tsallis Entropy Information Metric method." Neural Networks (IJCNN), 2016 International Joint Conference on. IEEE, 2016.
[8] Farid, Dewan Md, Nouria Harbi, and Mohammad Zahidur Rahman. "Combining naive bayes and decision tree for adaptive intrusion detection." arXiv preprint arXiv:1005.4496 (2010).
[9] Hall, Mark, et al. "The WEKA data mining software: an update." ACM SIGKDD explorations newsletter 11.1 (2009): 10-18.
[10] SigmaPlot, 10.0, San Jose, CA: Systat, Software Inc. https://systatsoftware.com/