Computer and Information Science
www.ccsenet.org/journal.html
Vol. 1, No. 2
May 2008

Research on Decision Forest Learning Algorithm

Limin Wang
College of Computer Science and Technology
Jilin University
Changchun 130012, China
E-mail:wanglim@jlu.edu.cn

Xiongfei Li
College of Computer Science and Technology
Jilin University
Changchun 130012, China
E-mail: lxf@jlu.edu.cn

Abstract

Decision Forests are investigated for their ability to provide insight into the confidence associated with each prediction, the ensembles increase predictive accuracy over the individual decision tree model established. This paper proposed a novel “bottom-top” (BT) searching strategy to learn tree structure by combining different branches with the same root, and new branches can be created to overcome overfitting phenomenon.

Keywords: Decision Forest, BT, Overfitting

1. Introduction

Decision tree based methods of supervised learning represent one of the most popular approaches within the AI field for dealing with classification problems. They have been widely used for years in many domains such as pattern recognition, data mining, signal processing, etc. The overfitting phenomenon is a persistent problem in using decision trees for classification. In existing decision tree based classification approaches a fully trained tree is often pruned to improve the generalization accuracy even if the error rate on the training data increases (Krzysztof, 2002). In the last decade multiple approaches have been studied to overcome this problem. Promising results were achieved using ensembles of multiple classifiers, which is under the assumption that "two (or more) heads are better than one." The decisions of multiple hypotheses are combined in ensemble learning to produce more accurate results. This type of learning algorithms are called Decision Forests include Random Forests (RFs) (Lariviere et al, 2005), Random Split Trees (RSs) (Breiman, 2001), and Bootstrap Aggregating (Bagging) (Pino-Mejias et al, 2008). These trees can be formed by various methods (or by one method, but with various parameters of work), by different sub-samples of observations over one and the same phenomenon, by use of different characteristics. This strategy is based on the observation that, decision tree classifiers can vary substantially when a small number of training samples are added or deleted from the training set.

With the ability of Decision Forests to increase predictive accuracy over the individual model established, the ensembles are investigated for their ability to provide insight into the confidence associated with each prediction. Test samples invariably include new variation that was not in the training set and the ability of the model to accurately predict these samples may be limited. Decision tree inducers are unstable in that resultant trees are sensitive to minor perturbations in the training data set. Largely for this reason, decision trees are widely applied as the base learners in ensemble learning schemes. Some ensemble algorithms have implemented modified decision tree inference algorithms in order to generate diverse decision forests.

2. Details of Some Related Ensemble Schemes

Bootstrap aggregation (Bagging) is a common way of introducing variation into individual trees in a forest. When unstable learning algorithms are applied as base inducers, a diverse ensemble can be generated by feeding the base learner with training sets re-sampled from the original training set. Each decision tree in a bagged decision forest is generated from a bootstrapped sample of the original training dataset. The subsample is formed by random independent selection of samples from initial sample. The probability of selection is identical to each sample. The volume of sub-sample is set beforehand. After the construction of a tree by way of analysis of given sub-sample, the selected observations return into initial sample and the process repeats the given number of times. This process generates a forest where each tree has been trained on a slightly different dataset which hopefully reduces the trees tendency to overfit the training set.

AdaBoost algorithm is referred to as an arcing (adaptive re-sampling and combining) algorithm. AdaBoost iteratively alters the probability over instances in the training set while performing the re-sampling. It works very well when the data is noise free and the number of training data is large. But during construction of the second tree, more attention is given to those objects which have a bigger error, with the purpose to reduce it. When noise is present in the training sets, or the number of training data is limited, AdaBoost does not perform as well as Bagging. The fundamental difference between bagging and AdaBoost is that while bagging is non-deterministic, AdaBoost is deterministic and iterative.

Random forests are a recent addition to the set of available decision forest methods and essentially extend bagging to include bagging the columns (variables) of the data matrix as well as the rows (samples) of the data matrix. Each decision tree in a random forest is trained on a distinct and random data set re-sampled from the original training set, using the same procedure as bagging. This additional randomization step generates significantly more diversity in the forest and additionally provides a significant speed improvement in tree construction time as compared to a Bagged model. While selecting a split at each internal node during the tree growing process, a random set of features is formed by either choosing a subset of input variables or constructing a small group of variables formed by linear combinations of input variables. Random forests have achieved “right”/“wrong” predictive accuracy comparable to that of AdaBoost and much better results on noisy data sets. Breiman also claimed and showed that AdaBoost is a form of random forest (algorithm).

3. Construction of Forests

We proposed a new approach of forest generation based on “bottom-top” (BT) searching strategy. In each stage of BT searching procedure, a branch is found to best fit current test sample, so these branches with the same root can construct a decision tree and more complex structures are created. The proposed method can help to build the shortest tree with maximal accuracy possible and thus, the tree does not need to be pruned to obtain good generalization.

Traditionally, attribute selection measure $Gain(T, A_k)$ generally based on information theory, serves as a criterion in choosing among a list of candidate attributes at each decision node, the attribute that generates partitions where samples are distributed less randomly, with the aim of constructing the smallest tree among those consistent with the data.

$$Gain(T, A_k) = E(T) - E_{A_k}(T) \tag{1}$$

where

$$E(T) = -\sum_{i=1}^{n} \frac{n(C_i, T)}{|T|} \log_2 \frac{n(C_i, T)}{|T|} \tag{2}$$

and

$$E_{A_k}(T) = \sum_{v \in D(A_k)} \frac{|T_v^{A_k}|}{|T|} E(T_v^{A_k}) \tag{3}$$

$n(C_i,T)$ denotes the number of samples in the training set T belonging to the class C_i, $D(A_k)$ denotes the finite domain of the attribute A_k and $T_v^{A_k}$ denotes the cardinality of the set of objects for which the attribute A_k has the value v.

To classify a new sample, having only values of all its attributes, we start with the root of the constructed tree and follow the path corresponding to the observed value of the attribute in the interior node of the tree. This process is continued until a leaf is encountered. Finally, we use the associated label to obtain the predicted class value of the instance at hand.

Correspondingly, given a test sample and in which attribute A_k has the value A_k (p), then $T_v^{A_k}$ in Eq.(3) is redefined as $T_V^{(Ak(\mathrm{p}))}$.

The precise algorithm proposed in this manuscript goes as follows:

– Run BT searching procedure for the whole training data, branches with different roots are created.

– Combine these branches with the same root to construct decision forest.

– Discard the branch that do not correspond to a state in any of cross validation parts (the idea is that such search states are not common and would not generalize well.

– If test sample does not match any branch, build a new branch for it.

References

Krzysztof Gra and bczewski, Wl odzisl aw Duch. (2002). *Heterogeneous Forests of Decision Trees*. In proceedings of the International Conference on Artificial Neural Networks. 504-509.

Lariviere Bart, Van Den Poel Dirk. (2005). Predicting customer retention and profitability by using random forests and regression forests techniques. *Expert Systems with Applications*. 29(2). 472-484.

Breiman, L. (2001). Random forests. Machine Learning. 45(1). 5-32.

Pino-Mejias Rafael, Jimenez-Gamero. (2008). Reduced bootstrap aggregating of learning algorithms. *Pattern Recognition Letters*. 29(3). 265-271.