Applied Soft Computing 12 (2012) 3615–3626

Contents lists available at SciVerse ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Inducing decision trees with an ant colony optimization algorithm

Fernando E.B. Otero*, Alex A. Freitas, Colin G. Johnson

School of Computing, University of Kent, UK

ARTICLE INFO

Article history:
Received 16 September 2011
Received in revised form 22 May 2012
Accepted 23 May 2012
Available online 30 June 2012

Keywords:
Ant colony optimization
Data mining
Classification
Decision tree

ABSTRACT

Decision trees have been widely used in data mining and machine learning as a comprehensible knowledge representation. While ant colony optimization (ACO) algorithms have been successfully applied to extract classification rules, decision tree induction with ACO algorithms remains an almost unexplored research area. In this paper we propose a novel ACO algorithm to induce decision trees, combining commonly used strategies from both traditional decision tree induction algorithms and ACO. The proposed algorithm is compared against three decision tree induction algorithms, namely C4.5, CART and *c*ACDT, in 22 publicly available data sets. The results show that the predictive accuracy of the proposed algorithm is statistically significantly higher than the accuracy of both C4.5 and CART, which are well-known conventional algorithms for decision tree induction, and the accuracy of the ACO-based *c*ACDT decision tree algorithm.

1. Introduction

One of the most studied data mining tasks in the literature is the classification task [15,29]. In essence, the classification task consists of learning a predictive relationship between input values and a desired output. Each example (data instance or record) is described by a set of features (attributes)—referred to as predictor attributes—and a class attribute. Given a set of examples, a classification algorithm aims at creating a model, which represents the relationship between predictor attributes values and class values (labels), and which is able to predict the class label of a new (unseen) example based on the values of its predictor attributes.

Classification problems can be viewed as optimisation problems, where the goal is to find the best function (model) that represents the predictive relationships in the data. A classification problem can be formally specified as:

- Given: $\{(e_1, c_{e_1}), \ldots, (e_n, c_{e_n})\}$ pairs representing the training data D, where each e_i denotes the set of predictor attributes' values of the i-th example ($1 \leq i \leq n$, where n is the total number or examples), and each c_{e_i} denotes the class label associated with the i-th example out of m different class labels available in the set C.
- Find: a function $f: D \rightarrow C$ that maps each example e_i in D to its correspondent class label c_{e_i} in C.

The main goal of a classification algorithm is to build a model that maximises the predictive accuracy—the number of correct predictions—in the test data (unseen during training), although in many application domains the comprehensibility of the model plays an important role [15,8,19]. For instance, in medical diagnosis the classification model should be validated and interpreted by doctors; in credit scoring the classification model should be interpreted by an expert, improving their confidence in the model; in protein function prediction the classification model should be interpreted to provide useful insights about the correlation of protein features and their functions and ultimately improve the current biological knowledge about protein functions. In these domains, it is crucial to produce comprehensible classification models.

Ant colony optimization (ACO) [11–13] algorithms involve a colony of ants (agents), which despite the relative simplicity of their individuals' behaviours, cooperate with one another to achieve an unified intelligent behaviour. As a result, the colony produces a system capable of performing a robust search to find high-quality solutions for optimisation problems with a large search space. In the context of the classification task in data mining, ACO algorithms have the advantage of performing a flexible robust search for a good combination of predictor attributes, less likely to be affected by the problem of attribute interaction [10,17].

Decision trees are widely used as a comprehensible representation model, given that they can be easily represented in a graphical form and also be represented as a set of classification rules, which generally can be expressed in natural language in the form of IF-THEN rules. Most ACO algorithms for classification in data mining have focused on extracting classification rules [22,18]. In this paper, we propose a novel ACO algorithm for the induction of decision trees. The proposed algorithm—called Ant-Tree-Miner (ant colony

[*] Corresponding author.
E-mail addresses: F.E.B.Otero@kent.ac.uk (F.E.B. Otero), A.A.Freitas@kent.ac.uk (A.A. Freitas), C.G.Johnson@kent.ac.uk (C.G. Johnson).

1568-4946/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.asoc.2012.05.028

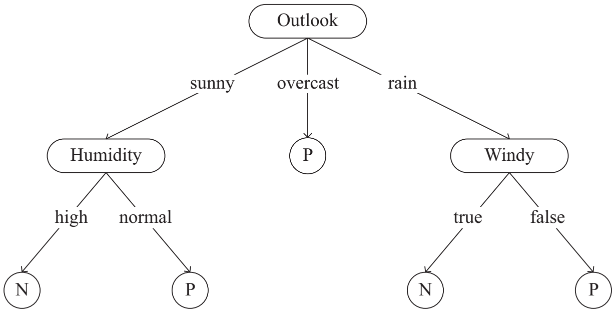

Fig. 1. An example of a decision tree, adapted from [30]. Internal nodes (including the root node) are represented by attribute names and branches originating from internal nodes correspond to different values of the attribute in a node; leaf nodes are represented by different class labels. In this example there are three attributes: Outlook {sunny, overcast, rain}, Humidity {high, normal} and Windy {true, false}; and two class labels {N, P}.

optimization-based decision tree induction)—is compared against two well-known decision tree induction algorithms, namely C4.5 [31] and CART [6], and the ACO-based *c*ACDT algorithm [5] in 22 publicly available data sets in terms of both predictive accuracy and size of the induced decision trees.

The remainder of this paper is organised as follows. Section 2 presents the background of this paper, discussing the top-down strategy commonly used to induce decision trees, an overview of ant colony optimization (ACO) and the related work on ACO algorithms for induction of tree structures. The proposed algorithm is described in Section 3. The computational results are presented in Section 4. Finally, Section 5 concludes this paper and presents future research directions.

2. Background

2.1. Top-down induction of decision trees

Decision trees provide a comprehensible graphical representation of a classification model, where the internal nodes correspond to attribute tests (decision nodes) and leaf nodes correspond to the predicted class labels—illustrated in Fig. 1. In order to classify an example, the tree is traversed in a top-down fashion from the root node towards a leaf node, moving down the tree by selecting branches according to the outcome of attribute tests represented by internal nodes until a leaf node is reached. At this point, the class label associated with the leaf node is the class label predicted for the example.

A common approach to create decision trees automatically from data is known as the divide-and-conquer approach, which consists of an iterative top-down procedure of selecting the best attribute to label an internal node of the tree. It starts by selecting an attribute to represent the root of the tree. After the selection of the first attribute, a branch for each possible (set of) value(s) of the attribute is created and the data set is divided into subsets according to the examples' values of the selected attribute. The selection procedure is then recursively applied to each branch of the node using the corresponding subset of examples—i.e., the subset with examples which have the attribute's value associated with the branch—and it stops for a given branch when all examples from the subset have the same class label or when another stopping criterion is satisfied, creating a leaf node to represent a class label to be predicted. The divide-and-conquer approach represents a greedy strategy to create a decision tree, since the selection of an attribute at early iterations cannot be reconsidered at later iterations—i.e., the selection of the best attribute is made locally at each iteration, without taking into consideration its influence over the subsequent iterations.

The problem of inducing a decision tree following the divide-and-conquer approach is divided into smaller problems of selecting an appropriate attribute given a set of examples. Several heuristics for choosing attributes have been used in the literature [23,24,33]. The CART algorithm [6] uses the Gini Index as a measure of impurity of an attribute, where the lower the impurity the better is the attribute. Quinlan [30] introduced the ID3 algorithm, which selects attributes based on the information gain measure—a measure derived from the entropy measure commonly used in information theory [7]. ID3 is the precursor of the well-known C4.5 algorithm [31]. More recently, a distance-based measure has been used in the Clus algorithm [3,36].

The C4.5 algorithm, probably the best known decision tree induction algorithm, employs an entropy-based criterion in order to select the best attribute to create a node, called the information gain ratio. In essence, the entropy measures the impurity of a collection of examples relative to their values of the class attribute, where higher entropy values correspond to more uniformly distributed examples, while lower entropy values correspond to more homogeneous examples (more examples associated with the same class label). The entropy of a collection of examples S is given by

$$\text{Entropy}(S) = \sum_{c=1}^{m} - p_c \cdot \log_2 p_c \,, \tag{1}$$

where p_c is the proportion of examples in S associated with the c-th class label and m is the total number of class labels. Using the entropy measure, the information gain of an attribute A corresponds to the expected reduction in entropy achieved by dividing the training examples into T subsets, where T is the number of different values in the domain of attribute A, and is defined as

$$\text{Gain}(S, A) = \text{Entropy}(S) - \sum_{v=1}^{T} \frac{|S_v|}{|S|} \cdot \text{Entropy}(S_v), \tag{2}$$

where $|S_v|$ is the number of examples in the subset of S for which the attribute A has the v-th value in the domain of A and $|S|$ is the number of examples in S. The calculation of the information gain ratio includes a penalty for attributes that divide the training examples into very small subsets, called the split information, computed as

$$\text{Split Information}(S, A) = \sum_{v=1}^{T} - \frac{|S_v|}{|S|} \cdot \log_2 \frac{|S_v|}{|S|}. \tag{3}$$

Such a penalty is necessary since attributes that divide the training examples into very small subsets are likely to have a high information gain just because the entropy of each subset is artificially small, given that the very small number of examples in each subset can be easily associated with a single class label without indicating a good generalisation ability. An extreme case would be an attribute that has a different value for each training example. This attribute would have a high information gain, since if we divide the training examples by its values, we would have subsets with examples associated with the same class label, even though the size of each subset is one. Clearly, this attribute represents a poor predictor (no generalisation ability) and would not be useful to classify unseen examples.

Finally, the information gain ratio of an attribute A is derived from the Gain and Split Information measures, and is given by

$$\text{Gain}\quad\text{Ratio}(S, A) = \frac{\text{Gain}(S, A)}{\text{Split}\quad\text{information}(S, A)} \tag{4}$$

At each step of the top-down procedure, the C4.5's selection favours the attribute that maximises information gain ratio, which corresponds to the attribute that provides the larger gain in terms of