Knowledge-Based Systems 79 (2015) 51–67

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Coverage-based resampling: Building robust consolidated decision trees

Igor Ibarguren *, Jesús M. Pérez, Javier Muguerza, Ibai Gurrutxaga, Olatz Arbelaitz

Department of Computer Architecture and Technology, University of the Basque Country UPV/EHU, Manuel Lardizabal 1, 20018 Donostia, Spain[1]

ARTICLE INFO

Article history:
Received 3 July 2014
Received in revised form 23 December 2014
Accepted 24 December 2014
Available online 9 January 2015

Keywords:
Comprehensibility
Consolidated decision trees
Class imbalance
Resampling
Inner ensembles

ABSTRACT

The class imbalance problem has attracted a lot of attention from the data mining community recently, becoming a current trend in machine learning research. The Consolidated Tree Construction (CTC) algorithm was proposed as an algorithm to solve a classification problem involving a high degree of class imbalance without losing the explaining capacity, a desirable characteristic of single decision trees and rule sets. CTC works by resampling the training sample and building a tree from each subsample, in a similar manner to ensemble classifiers, but applying the ensemble process during the tree construction phase, resulting in a unique final tree. In the ECML/PKDD 2013 conference the term "Inner Ensembles" was coined to refer to such methodologies. In this paper we propose a resampling strategy for classification algorithms that use multiple subsamples. This strategy is based on the class distribution of the training sample to ensure a minimum representation of all classes when resampling. This strategy has been applied to CTC over different classification contexts. A robust classification algorithm should not just be able to rank in the top positions for certain classification problems but should be able to excel when faced with a broad range of problems. In this paper we establish the robustness of the CTC algorithm against a wide set of classification algorithms with explaining capacity.

1. Introduction

In data mining, a classification problem occurs when an object needs to be assigned to a predefined group or class based on a number of observed attributes related to that object [1].

Class imbalance has been considered one of the main problems in data mining in recent years [2–6]. The class imbalance problem occurs when at least one of the classes (minority class/es) is underrepresented in the original training sample compared to the remaining classes. The imbalance can be either intrinsic (directly related to the nature of the data, such as the diagnosis of rare diseases) or extrinsic. Extrinsic imbalance can be caused by limitations in the data collection process [7]. Class imbalance is present in several real problems, such as medical diagnosis [8], insurance fraud detection [9], customer churn prevention [10], traffic incident detection [11] and DNA sequencing [12].

Class imbalance has a detrimental effect on classification algorithms that maximize overall accuracy [3]. In the presence of class imbalance, such algorithms might build a trivial classifier that classifies all examples as majority class, obtaining a high overall accuracy but misclassifying all minority class examples (which is usually the class of interest). This is the case with the well known C4.5 decision tree algorithm [13] and its pruning mechanism. This mechanism iteratively deletes leaf nodes by looking for the deletion that maximizes accuracy gain until no deletion increases accuracy. In the presence of class imbalance, the deleted branches are usually responsible for correctly classifying the minority class examples [14]. Class imbalance can also amplify the effects of other classification problems such as concept complexity [15], high dimensionality combined with small sample size [3] or small disjuncts [16].

The CTC (Consolidated Tree Construction) algorithm [17] was proposed for an insurance fraud detection problem where class imbalance was present [9]. CTC creates a set of subsamples from a training sample and builds a decision tree from each subsample in a similar manner to Bagging [18] but applying the ensemble process when building the tree by voting on the split on each of the tree's nodes. Abbasian et al. [19] recently coined the term "Inner Ensembles" for similar procedures and suggested extending it to other algorithms, such as Bayesian networks and K-means. Unlike ensemble algorithms, the final model of the CTC algorithm is a simple decision tree understandable by humans. The mining of understandable patterns is a current trend in data mining, as highlighted in a recent special issue of a high-ranking journal in the field of

[1] http://www.sc.ehu.es/aldapa/.

* Corresponding author.
E-mail addresses: igor.ibarguren@ehu.es (I. Ibarguren), txus.perez@ehu.es (J.M. Pérez), j.muguerza@ehu.es (J. Muguerza), i.gurrutxaga@ehu.es (I. Gurrutxaga), olatz.arbelaitz@ehu.es (O. Arbelaitz).

http://dx.doi.org/10.1016/j.knosys.2014.12.023
0950-7051/

artificial intelligence [6]. Consolidated trees are more stable and less complex than the C4.5 trees they are based on. Consolidated trees change far less when induced from different training samples and are thus more stable [20]. The complexity of the trees, represented as the amount of internal nodes, is smaller in consolidated trees. These features are important because, as Turney [21] and Domingos [22] pointed out separately "engineers are disturbed when different batches of data from the same process result in radically different decision trees. The engineers lose confidence in the decision trees, even when we can demonstrate that the trees have high predictive accuracy." and "a single decision tree can easily be understood by a human as long as it is not too large".

In the work presented in this paper a novel resampling methodology is proposed and applied to the CTC algorithm. This methodology uses the notion of *coverage*, the minimum percentage of instances from any class of the training sample present in the subsample set with a different class distribution, to determine the amount of subsamples needed. Thus, instead of setting a fixed amount, the number of subsamples is determined by the data set's class distribution, the subsample type and the chosen coverage value. The greater the class imbalance present in the training set, the more subsamples are necessary to achieve the same coverage. The results achieved by CTC using this new resampling strategy are compared to those published by Fernández et al. [23]. They proposed a taxonomy of sixteen rule-based evolutionary algorithms, dividing them into 3 main categories and 5 families. The discriminating ability of the algorithms was tested in three different contexts: a set of 30 standard (mostly multi-class) data sets, 33 two-class imbalanced data sets and the same two-class data sets preprocessed with SMOTE to balance the class distribution. All the data sets were taken from the KEEL repository.[2] For each of the three contexts an intra-family comparison was performed and the best ranking algorithms of each family of the taxonomy were compared, along with a fixed set of six classical non-evolutionary classification algorithms. All twenty-two algorithms used in their work (whether rule-based or not) are explanatory, which makes them natural rivals to CTC. This makes that experiment an ideal environment to test CTC with the coverage-based resampling strategy.

The main contribution of this paper is the use of the notion of coverage. Depending on the difficulty of the problem (defined by the class distribution in the data set) and the characteristics of the subsamples to be created, the coverage determines the adequate number of samples to build consolidated trees. In previous works, CTC has never been used with data sets with such high degree of class imbalance and such small size. Coverage-based resampling ensures that the number of samples does not fall short of representing all classes to a minimum degree, independently of the class distribution. Furthermore, we have generalized this strategy in the context of multi-class data sets, where class imbalance is also present but usually not studied. In the analysis performed in this work in three classification contexts, a coverage value of 99% has been determined to be the most adequate for the CTC algorithm. Also, although applying SMOTE had previously never improved CTC's performance in a significant manner, the combination of coverage-based resampling with the use of SMOTE has been able to do so.

In this work we want to establish CTC's robustness by showing that it ranks in the top positions for different classification contexts compared against a wide range of algorithms, all with explaining capability. The significance of CTC's performance compared to its competitors is backed up by performing rigorous statistical testing following the guidelines established in the field of machine learning research [24–26].

[2] http://sci2s.ugr.es/keel/datasets.php.

The rest of the paper is organized as follows. Section 2 gives an overview of the related work in the fields of class imbalance, tree and rule induction algorithms and the CTC algorithm. Section 3 presents the coverage-based resampling and states the hypothesis of this work. Sections 4 and 5 respectively describe the experimental setup and the analysis of results. Finally, Section 6 gives this work's conclusions and details future work.

2. Related work

This section reviews the latest developments in decision tree and rule induction methods and techniques to solve the class imbalance problem. The last subsection reviews the research on the CTC algorithm that has led to the experiments presented in this paper.

2.1. Tree and rule induction methods

In machine learning, sometimes the reason why a classifier makes a decision is as important as the accuracy of the decision itself. This is especially true for domains where the classifier works as a decision support system for humans, such as medical diagnosis and fraud detection. Some classification algorithms have a white box nature, where the decision of the classifier can easily be interpreted by a human. Decision trees and rule sets are classifiers of this type.

A classification tree, also known as a decision tree, is a set of conditions organized in a hierarchical structure. Instances are classified by navigating them from the root node down to a leaf, according to the outcome of the tests along the path [27].

Some of the earliest decision tree algorithms were CHAID [28] and CART [29]. Later Quinlan's ID3 [30] and its successor C4.5 [13] were published. Decades later, C4.5 is still considered one of the top algorithms in machine learning [31]. These algorithms differ from each other in a number of ways, such as: the type of attributes they can handle, their split criteria when making decisions, the type of split made and the presence of a pruning mechanism. Oblique decision trees [32] build decision trees where each split is made based on more than one variable.

Rule induction algorithms formulate rules that aim to describe the concept of interest as a set of conditions for the attributes that describe the examples. Some rule induction methods such as C4.5-Rules [13] and PART [33] derive rules from decision trees. In other rule induction algorithms rule sets are formed from scratch by sequentially building rules using a separate-and-conquer strategy: Each rule covers a portion of the training sample, the examples covered by that rule are removed from the training sample and the next rule is built [34]. A rule set is able to separate examples belonging to the class of interest from the rest. Algorithms such as IREP [35] build rules from scratch. Cohen [36] proposed Ripper, which shows an improvement over IREP's discriminating capacity without sacrificing much computational efficiency.

Rule-induction algorithms can use several techniques to build rules. One of these techniques is evolutionary algorithms. Rule-based techniques that make use of evolutionary algorithms are genetics-based machine learning (GBML) algorithms. In the past, GBML systems were classified into the Michigan and Pittsburgh categories. However not all evolutionary rule-based algorithms fall into those categories and recently a new taxonomy was proposed based on the representation of the chromosome of the associated evolutionary algorithm [23]. This taxonomy defines five categories, divided into three families. The first family encompasses those algorithms where the chromosome is a rule and has three subcategories that differ in their approach: the classic Michigan approach, the iterative rule learning approach and the genetic