Expert Systems with Applications 42 (2015) 4859–4866

Contents lists available at ScienceDirect

Expert Systems with Applications

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

Finding the best diversity generation procedures for mining contrast patterns

Milton García-Borroto [a,*], José Fco. Martínez-Trinidad [b], Jesús Ariel Carrasco-Ochoa [b]

[a] Instituto Superior Politécnico José Antonio Echeverría, Calle 114 # 11901, Marianao, La Habana C.P. 19390, Cuba
[b] National Institute of Astrophysics, Optics, and Electronics, Luis Enrique Erro No. 1, Sta. María Tonanzintla, Puebla C.P. 72840, Mexico

ARTICLE INFO

Article history:
Available online 26 February 2015

Keywords:
Understandable classifiers
Contrast patterns
Ensemble diversity
Deterministic procedures

ABSTRACT

Most understandable classifiers are based on contrast patterns, which can be accurately mined from decision trees. Nevertheless, tree diversity must be ensured to mine a representative pattern collection. In this paper, we performed an experimental comparison among different diversity generation procedures. We compare diversity generated by each procedure based on the amount of total, unique, and minimal patterns extracted from the induced tree for different minimal support thresholds. This comparison, together with an accuracy and abstention experiment, shows that Random Forest and Bagging generate the most diverse and accurate pattern collection. Additionally, we study the influence of data type in the results, finding that Random Forest is best for categorical data and Bagging for numerical data. Comparison includes most known diversity generation procedures and three new deterministic procedures introduced here. These deterministic procedures outperform existing deterministic method, but are still outperformed by random procedures.

1. Introduction

A supervised classifier predicts the class of a query object based on a model built using a training sample. Although an accurate prediction is an important component of the classifier quality, the lack of comprehensibility of classification results may cause a reluctance to use certain classifiers. For example, when credit has been denied to a customer, the Equal Credit Opportunity Act of the US requires the financial institution to provide the reasons for rejecting the application; indefinite or vague reasons for denial are illegal (Martens, Baesens, Gestel, & Vanthienen, 2007).

A pattern is an expression defined in some language that describe some properties of a collection of objects. An important characteristic of a pattern is its *support*, defined as the ratio of objects described by the pattern with respect to the whole object collection. To differentiate between useful and random patterns, a minimal support threshold (μ) is frequently used. If the support of a given pattern is below μ, the pattern is discarded as useless. A pattern that appears significantly more in a group or class than in the remaining groups or classes, capturing existing contrasts, is named *contrast pattern*. Most understandable classifiers are based on contrast patterns (Dong, 2012). Contrast patterns contained in a query object can be used to find its class, and also provides an explanation of the classification in terms that are easy to understand in the user language. Examples of contrast pattern based classifiers are emerging patterns, decision trees, and rules.

Contrast patterns can be mined from a set of decision trees (Boulesteix, Tutz, & Strimmer, 2003; García-Borroto, Martínez-Trinidad, Carrasco-Ochoa, Medina-Pérez, & Ruiz-Shulcloper, 2010; Quinlan, 1987). Extracting patterns from the tree can be better than directly generating the patterns (like rule miners or emerging pattern miners (Novak, Lavra, Webb, Lavrac, & Webb, 2009)) for two reasons (Quinlan, 1987). First, the local discretization performed by decision tree miners with numeric features avoids doing an a priori global discretization, which might cause information loss. Second, even longer paths of decision trees contain small proportion of candidate attributes, which reduces significantly the search space of potential patterns.

The extraction of contrast patterns from a set of decision trees has some advantages over keeping the trees, because there is no obvious way to integrate trees into more complex structures that are more accurate than their components (Quinlan, 1987). Additionally, keeping all independent classifiers requires a significantly larger amount of memory (Lin et al., 2014). On the other hand, creating the collection of extracted patterns from all the trees, reducing it through a filtering procedure, and obtaining an accurate model, is a simple procedure (García-Borroto et al., 2010). Additionally, it is usually simpler to understand classification results using the

* Corresponding author.
E-mail addresses: milton.garcia@gmail.com (M. García-Borroto), fmartine@ccc.inaoep.mx (J.F. Martínez-Trinidad), ariel@ccc.inaoep.mx (J.A. Carrasco-Ochoa).

http://dx.doi.org/10.1016/j.eswa.2015.02.028
0957-4174/

reduced collection of matched patterns than to understand the collective vote of individual trees.

Diversity is the most important property of classifier ensembles, based on the rationale that a collection of nearly identical classifiers cannot outperform any of their components (Didaci, Fumera, & Roli, 2013). There are different concepts of diversity, each one reflecting intuitive notions, and so there are many ways to measure it. However, diversity has not being fully understood yet (Wang & Yao, 2013). Most measures of classifier diversity are based on estimating the independence of the errors performed by the components of the ensemble (Lysiak, Kurzynski, & Woloszynski, 2014). In order to outperform a single decision tree, extracting contrast patterns from a collection of decision tree is meaningful only if the tree collection is diverse (García-Borroto et al., 2010). Nevertheless, using diversity measures based on the classification output for a task where decision trees are discarded and only some patterns per tree are selected can be meaningless.

Some interesting questions arise here: What is the best method for generating diversity, in order to extract more and better contrast patterns? How can we measure the tree diversity considering that the trees are discarded and only some of their contained patterns are interesting? Which method is more efficient for mining patterns, in order to reduce the number of induced trees? What strategy is better for generating diversity in the trees for mining patterns among selecting features, selecting objects, or selecting splits? This paper presents answers to these question, comparing existing and newly introduced diversity generation procedures for mining contrast patterns. It proposes to evaluate diversity by the amount of total, unique, and minimal patterns extracted from the collection of decision trees using different minimal support thresholds. This evaluation allows a deeper understanding about the characteristics of the resultant set of patterns mined per method. As a result, the paper presents some ideas about the behavior of different strategies for generating diversity.

Most methods for generating diversity in decision trees are based on random samplings of features and/or instances. In this paper, three new deterministic methods for generating tree diversity are introduced. Deterministic methods are interesting because many subsets selected by random procedures might contain few, if any, informative features (Ye, Wu, Huang, Ng, & Li, 2013). In general, random procedures does not guarantee that the randomly selected features have the necessary discriminant information (Harandi, Ahmadabadi, Araabi, Bigdeli, & Lovell, 2010; García-Pedrajas & Ortiz-Boyer, 2008).

Summarizing, this paper presents the following results. First, it compares a representative collection of methods for generating tree diversity, comparing their results based on diversity and accuracy. The paper suggest the bests methods according to database characteristics. These results can guide the development of future mining methods, both random and deterministic. Second, it introduces a new method for measuring diversity for the task of using decision trees to extract contrast patterns. Third, the paper introduces three new deterministic methods for generating diversity. Although these methods outperform the existing method LCMine, they are outperformed by some random methods. Finally, up to our knowledge, this is the first paper to study the influence of the feature type in the behavior of each diversity generation procedure. This result can help future researchers to improve actual procedures by using feature type information.

The paper consists of five sections. Section 2 contains materials and methods used throughout the paper. A revision of the theory and state of the art regarding diversity appears in Section 2.1. Section 4 presents the main results of the paper and a detailed discussion. Conclusions and future work are presented in Section 5.

2. Material and methods

In this section, materials and methods are presented. Contents in this section allow readers to evaluate the work performed and provide sufficient details to allow the work to be reproduced. It contains two subsections. Section 2.1 introduces the existing diversity generation procedures while Section 2.2 provides details about the experiments performed, algorithms, and methods for analyzing their results.

2.1. Diversity generation procedures

In this section, the diversity generation procedures analyzed in this paper are presented. Although most of them are components of a more complex method, other components are disregarded. Nevertheless, the same name of the ensemble method is kept.

Bagging (Breiman, 1996) creates diversity by training each classifier with a bootstrap replicate of the training set. Bootstrap replicates are built by randomly sampling the training set, with replacement, until an equal number of instances than the training set is obtained. As decision trees are unstable classifiers (i.e., small changes in the training sample lead to significantly changes in the model), Bagging can be used with decision trees.

Random Forest (Breiman, 2001) creates diversity by selecting a Random Subset of attributes at each node in the decision tree. The best feature of the selected subset is then used to build the node. Although in the original paper each forest is built based on an bagged version of the training set, this process is not considered in this paper to avoid hidden dependencies in the result. The feature subset is taken with size $\log_2|Features|$, which is frequently used (Breiman, 2001; Kocev, Vens, Struyf, & Deroski, 2013). Using this reduced feature subset allows Random Forest to have less computational cost and less correlation between generated trees than other procedures (Xia, Du, He, & Chanussot, 2014). The success of Random Forest can be also explained because injecting randomness at the level of nodes tends to produce higher accuracy models (Dollar & Zitnick, 2013).

Random Subspaces (Ho, 1998) selects a random feature subset to build each decision tree. After testing different sizes of the subset, $\frac{|Features|}{2}$ returns the best results, and is selected for the experiments. As Random Forest also uses feature subspaces, the name *Random Subset* is used in the remaining sections.

Randomized C4.5 (Dietterich, 2000) randomly selects one of the best 20 candidate splits at each node of the decision tree. Since other methods also build randomized trees, the name *Random Split* is used in this paper.

LCMine (García-Borroto et al., 2010) is the only existing method for generating diversity that is deterministic. Diversity in LCMine is generated using a set of vectors ranging from $\vec{v} = (1, 1, \ldots, 1)$ to $\vec{v} = (k_1, k_2, \ldots, k_m)$, each one of a different decision tree. For a given vector, a tree is generated picking the best v_l candidate split for each node at level l. Usually, $i < j \Rightarrow k_i \geqslant k_j$, in order to allow higher diversity in upper nodes, where there is often a larger amount of good splits, while reducing the diversity in lower nodes, where there are usually fewer good splits. The vector $\vec{v} = (k_1, k_2, \ldots, k_m)$, where $m \in \{1, tree\_depth\}$, is known as the *level-diversity vector*.

As an example, consider the level-diversity vector $(5, 4, 3, 2)$. LCMine generates $5 * 4 * 3 * 2 = 120$ decision trees, starting from $(1, 1, 1, 1)$ to $(5, 4, 3, 2)$. For example, the tree for vector $(2, 1, 3, 2)$ is built using the second best split for the root node, the best split for nodes at the second level, the third best split for nodes at the third level, and the second best split at the fourth level of the decision tree. Any level deeper than the fourth level (m) uses the best candidate split.