A Survey of Merging Decision Trees Data Mining Approaches

Pedro Strecht

INESC TEC/Faculdade de Engenharia, Universidade do Porto
Rua Dr. Roberto Frias, 4200-465 Porto, Portugal
pstrecht@fe.up.pt

Abstract. The merging of decision tree models is a topic lacking a general data mining approach that is not domain specific. Existing research address the issue under different motivations and to solve different problems. This paper presents a survey of current approaches of merging decision trees, highlighting what they share in common by presenting a general data mining approach based of the combination of rules. Although its major components and problems are abstracted, illustrative examples from the literature are provided. Possible directions of unexplored issues for future research are also discussed.

Keywords: prediction models, decision tree merging, survey

1 Introduction

Classifiers obtained from decision tree models have the characteristic of not requiring previous domain knowledge or heavy parameter tuning making them appropriate not only for prediction but also for exploratory data analysis. The tree-like representation of knowledge presents itself as intuitive, making models that are usually interpretable by humans [1]. For this reason, decision trees models have been very popular as models in classification problems in various business domains and are still widely used.

The motivation to merge models has its origins as a strategy to deal with building prediction models for distributed data. Distribution can occur naturally, i.e., when the data is initially collected on distributed locations and transportation to form a *monolithic data set* (designation used in literature to refer to a single centralized data set) is costly or unsafe making it not feasible. An example is Bursteinas and Long [2] motivation which is related to data being generated on distributed distant machines connected by low transparency connections. Alternatively, distribution can occur artificially, being a strategy to deal with very large monolithic data sets which would make training a model a very slow task or even impossible due to lack of resources. Data sets exceeding RAM sizes is presented as a factor for distributed data by Andrzejak, Langner and Zabala [3]. Another reason to have artificially distributed data is when it is collected as consequence of a business process. An example is in Strecht, Moreira and Soares [4] research in which student enrolments records are gathered in courses to create models at course level in a university. Merging models appears as a technique to generalize the knowledge contained in those models at university level to provide useful information to help explain the drop out phenomenon.

A *local data set* can be defined as a subset of a larger monolithic data set that is splitted either naturally or artificially. Each local data set provides training examples to create local models. If the number of models is too large, it becomes difficult to generalize knowledge, and have a single model view. There are two major approaches to build generalized models from distributed data:

- *Data compression* in which data in each local data set is compressed into one or more training examples. Another process then gathers all examples and trains a generalized model. Yael and Elad [5] describe this approach in detail. The drawback is that there are no models created at each local data set, which may be required to understand business processes at local level.
- *Model merging* in which a local model is trained in each local data set and then another process combines them to form a generalized model. This has been used to a greater extent with different approaches which can be divided into two main groups: mathematical, in which a mathematical function is used to aggregate models; and data mining in which the models are broken down into parts, combined and re-assembled to form a new model (both are detailed in Section 2).

At first glance, it may seem that merging models is another form of ensemble learning. There are, however, major differences between the two methodologies that help to easily distinguish them, as presented in Fig. 1.

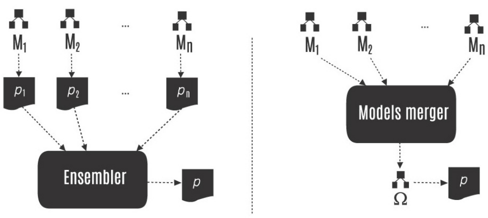


Fig. 1. Ensemble learning vs Models merging

Ensemble learning consists in combining the predictions (p_1, ..., p_n) made by various models (M_1, ..., M_n) into one prediction (p). The ensembler implements a method to combine the predictions, such as bootstrap aggregating (bagging), boosting, or random forests. Models merging consists in combining various models (M_1, ..., M_n) to create a merged model (Ω) which is the only one making a prediction.

The remainder of this paper is structured as follows. Section 2 presents an overview of approaches to merge decision trees. Section 3 describes a general data mining approach with different alternatives of carrying out intermediate steps. Section 4 presents the conclusions and identifies open issues for research.

2 Overview of Approaches for Merging Decision Trees

2.1 Mathematical Approaches

Kargupta and Park [6], motivated by the need to analyse and monitor time-critical data streams using mobile devices, propose an approach to merge decision trees using the Fourier Transform. This mathematical operation decomposes a function of time (a signal) into its frequencies yielding the frequency domain representation of the original signal. According to the authors, mining critical data streams requires on-line learning that produces a series of decision trees models, which may have to be compared with each other and aggregated, if necessary. Transmitting these models over a wireless network is presented as a problem. As the decision tree is a function, it can be represented in frequency domain, resulting in the model *spectra*. Merging models becomes a matter of adding theirs spectra's, a trivial task in frequency domain. If required, the merged model can be transformed back to the decision tree domain by the Inverse Fourier Transform. This approach has not evolved since 2004, when it was first presented. It has been criticized [3] for being difficult to extend (e.g. only binary attributes are considered by the authors) and for the lack of performance measures.

Gorbunov and Lyubetsky [7] address the problem by proposing a mathematical approach to construct a decision tree that is closest on average to a set of trees. The problem is analysed from a theoretical point of view, i.e., it is not presented as a solution to be used in a specific application. Nevertheless, decision trees illustrating the theory of evolution are pointed out as a proof of concept. A complex algorithm is described and exemplified by a case in which ten binary decision trees are combined, resulting in a *super tree* that represents their average. The algorithm is cited afterwards by the authors in another research [8] in the context of molecular biology, therefore suggesting what seems to have been the main motivation for its development. Although developed quite recently, in 2011, it has not been used by other researchers, probably due to its complexity.

2.2 Data Mining Approaches

Provost and Hennessy [9, 10] present an approach to learning and combining rules on disjoint subsets of a full training data. A rule based learning algorithm is used to generate rules on each subset of the training data. The merged model is constructed from satisfactory rules, i.e., rules that are generic enough to be evaluated in the other models. All rules that are considered satisfactory on the full data set are retained as they constitute a superset of the rules generated when learning is done on the full training set. This approach has not been replicated by other researchers.

A more common approach is the combination of rules derived from decision trees. The idea is to convert decision trees from two models into decision rules by combining the rules into new rules, reducing their number and finally growing a decision tree of the merged model. The basic fundamentals of the process are first presented in the doctoral thesis of Williams [11] and over the years, other researchers have contributed by proposing different ways of carrying out intermediate tasks. Table 1 summarizes research examples of this approach, specifying the problem (or motivation) and data sets used.

Table 1. Research examples of combination of rules approaches to merge models

Research	Problem/motivation	Data sets
Hall, Chawla and Bowyer [12]	Train model in a very large data set	Iris, Pima Indians Diabetes
Bursteinas and Long [2]	Mining data distributed on distant machines	UCI Machine Learning Repository
Andrzejak, Langner and Zabala [3]	Train models for distributed data sets and exceeding RAM sizes	UCI Machine Learning Repository
Strecht, Moreira and Soares [4]	Generalize knowledge in course models at university level	Academic data from University of Porto

Hall, Chawla and Bowyer [12, 13] research present as rationale that is not possible do train decision trees in very large data sets because it could overwhelm the computer system's memory by making the learning process very slow. Although a tangible problem in 1998, nowadays, this argument still makes sense as the notion of very large data sets has turned into the big data paradigm. The approach involves breaking down a large data set into n disjoint partitions, then, in parallel, train a decision tree on each. Each model, in this perspective, is considered an independent learner. Globally, models can be viewed as agents learning a little about a domain with the knowledge of each agent to be combined into one knowledge base. Simple experiments to test the feasibility of this approach were done on two datasets: Iris and Pima Indians Diabetes. In both cases, the data sets were split across two processors and then the resulting models merged.

Bursteinas and Long [2] research aims to develop a technique for mining data which is distributed on distant machines, connected by low transparency connections arguing that there is a lack of algorithms and systems which could perform data mining under such conditions. The merging procedure is divided into two scenarios: one for disjoined partitions and one for overlapped partitions. To evaluate the quality of the method, several experiments have been performed. The results showed the equivalence of combined classifiers with the classifier induced on a monolithic data set. The main advantage of the proposed method is its ability to induce globally applicable classifiers from distributed data without costly data transportation. It can also be applied to parallelise mining of large-scale monolithic data sets. Experiments are performed merging two models in data sets taken from the UCI Machine Learning Repository [14].

Andrzejak, Langner and Zabala [3] propose a method for learning in parallel or from distributed data. Factors cited as contributing to this trend include emergence of data sets with exceeding RAM sizes and inherently distributed scenarios such as mobile environments. Also in these cases interpretable models are favoured: they facilitate identifying artefacts and understanding the impact of individual variables. The method is compared with ensemble learning, because in a distributed environment, even if the individual learner on each site is interpretable, the overall model usually is not, citing as example the case of voting schemes. To overcome the problem they propose an approach for merging of decision trees (each learned independently) into a single decision tree. The method complements the existing parallel decision trees algorithms by providing interpretable intermediate models and tolerating constraints on bandwidth and RAM size. The latter properties are achieved by trading RAM and communication constraints for accuracy. The method and the mentioned trade-offs are evaluated in experiments on data sets from the UCI Machine Learning Repository [14].

In all previous presented research examples, decision trees were trained using C4.5 algorithm [15] and accuracy [1] was used as evaluation function of the individual and merged models.

Strecht, Moreira and Soares [4] research on educational data mining starts from the premise that predicting the failure of students in university courses can provide useful information for course and programme managers as well as to explain the drop out phenomenon. The rationale is that while it is important to have models at course level, their number makes it hard to extract knowledge that can be useful at the university level. Therefore, to support decision making at this level, it is important to generalize the knowledge contained in those models. An approach is presented to group and merge interpretable models in order to replace them with more general ones without compromising the quality of predictive performance. The case study is data from the University of Porto, Portugal, which is used for evaluation. The aggregation method consists mainly of intersecting the decision rules of pairs of models of a group recursively, i.e., by adding models along the merging process to previously merged ones. The results obtained are promising, although they suggest alternative approaches to the problem. Decision trees were trained using C5.0 algorithm [16] and F1 [17] was used as evaluation function of the individual and merged models.

3 Combination of Rules Approach to Merge Models

The *combination of rules* approach to merge decision trees models is the most common found in the literature. However, when presented it has always been in the context of a specific problem intertwined with details from the context of business rules. Therefore, there is the lack of a generalized approach that is not restricted to a specific domain. This section proposes such an approach by identifying its key components and the major problems encountered. Aligned with a survey perspective, it also presents the different alternatives to carry out intermediate tasks found in the literature.

Fig. 2 presents the system architecture of this approach which encompasses four main processes: models creation and evaluation, models grouping, models merging and group models evaluation. The local data sets ($D_1, \ldots, D_n$) are assumed to be collected and prepared by a data extraction process which is not part of the methodology. The outputs are group models ($G_1, \ldots, G_k$) and corresponding performance measures ($\theta(G_1), \ldots, \theta(G_k)$). θ denotes a function for evaluation measure (e.g. accuracy or F1). The following sub-sections detail each of the processes.

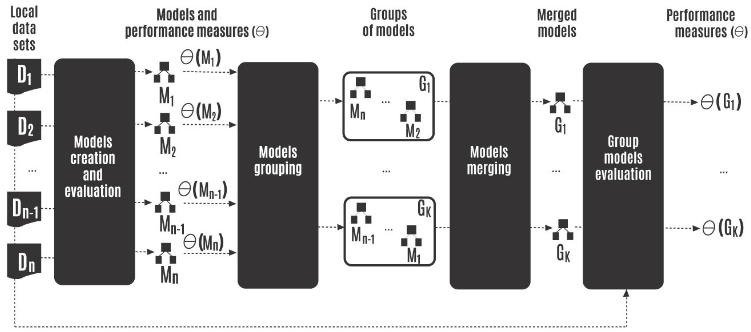


Fig. 2. System architecture of the combination of rules approach to merge models

3.1 Models creation and evaluation

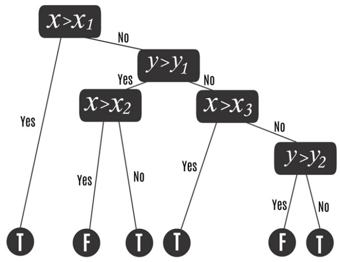
In the first process a decision tree model is created for each local data set. Due to limitations of space, the description of the decision tree induction process is not included. There are several algorithms to create decision trees, the most popular being CART [18] (Classification and Regression Trees) and C5.0 [16] (an evolution of the C4.5 [15] which is an extension of ID3). Although the approach for merging decision trees is not specific to any algorithm, it is recommended that the same algorithm is used to train all individual models. As pre-requisite for merging, it is mandatory to have access to the models themselves. Decision tree algorithms may output the model graphically, through oriented graphs, or textually, as a set of lines. Fig. 3 shows an example of a model in both representations with variables x and y, and classes T and F.

It is worthwhile noting that there may be local data sets for which a model is not created (because of the characteristics within the data). As a consequence, there may be less models than local data sets. For simplicity, however, in Fig. 2 it is assumed that to all n local data sets (D_1, ..., D_n) there is a respective decision tree model (M_1, ..., M_n).

The performance of each model is determined by an evaluation function $\theta(M_i)$. All models should be evaluated using the same experimental set-up and evaluation function.

3.2 Models grouping

In the second process the models are gathered into groups. Although this can be done according to any criteria it is important to make the distinction between two cases of grouping:




```
x > x1: T
x <= x1
:... y > y1
     :... x > x2: F
          x <= x2: T
     y <= y1
     :... x > x3: T
          x <= x3
          :... y > y2: F
               y <= y2: T
```

Fig. 3. Graphical and textual representation of a decision tree model

- *domain-knowledge* in which models are grouped together according to meta-information of the local data sets (e.g. different locations belonging to the same geographical area);
- *data-driven* in which models are grouped together according the characteristics of the models themselves (e.g. measures over the variables used).

Hall, Chawla and Bowyer [12] and Bursteinas and Long [2] do not address the issue of grouping which suggests that the models are all merged in sequence. Andrzejak, Langner and Zabala [3] define a k parameter to study the impact of increasing the number of groups. Each group always has the same number of models, with $k = 1$ being the baseline case. Strecht, Moreira and Soares [4] perform experiments grouping models relating to courses by ten scientific areas (domain-knowledge grouping), by the number of variables and importance of variable (data-driven grouping). The latter by clustering models (with k-means algorithm) using the C5.0 algorithm measure of importance of variable I_v which relates to the percentage of examples tested in a node by that variable in relation to all examples. Finally, a baseline case (similar to the other researchers) was included by forming only one group containing all models.

In Fig. 2, for illustrative purposes, M_n and M_2 belong to the first group (G_1) while M_{n-1} and M_1 are placed in the last (G_k) by some arbitrary criterion.

3.3 Models merging

In the third process the models in each group (or all in sequence if no grouping is performed) are merged together yielding the *group model*, according to the experimental set-up presented in Fig. 4.

A requirement for this process is that each model must be represented as a set of decision rules. This takes the form of a decision table, in which each row is a decision rule. Therefore, the first (M_1) and second (M_2) models are converted to decision tables and merged, yielding the Ω_1 model, also in decision table form. Then the third model (M_3) is also converted to a decision table and is merged with Ω_1 model yielding the Ω_2 model. This process is replicated to all models in the group. The last merged model Ω_{n-1} is converted to decision tree (renamed as group model). Each one of these sub-processes and its tasks are detailed next.

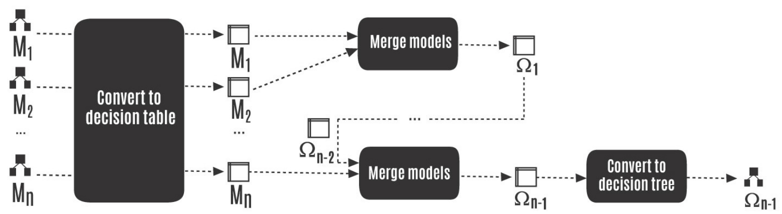
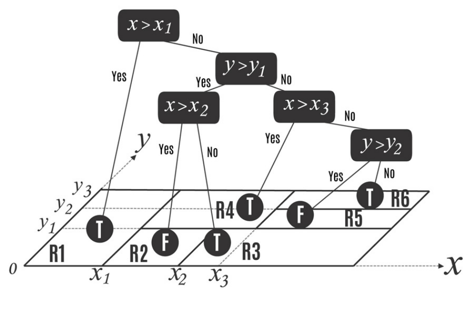


Fig. 4. Experimental set-up to merge all models in a group

Conversion to Decision Table. In the first sub-process, a decision tree is transformed to a set of rules. Each path from the root to the leaves creates a rule with a set of possible values for variables and a class. These have been called "rules set" by Hall, Chawla and Bowyer [12], "hypercubes" by Bursteinas and Long [2], "sets of iso-parallel boxes" by Andrzejak, Langner and Zabala [3], or "decision regions" by Strecht, Moreira and Soares [4]. All these designations arise from the fact that a variable can be considered as a dimension axis in a multi-dimensional space. The set of values (nominal or numerical) is the domain for each dimension and each rule defines a region. It is worth noting that all rules lead to regions that do not overlap and together cover the entire multi-dimensional space.

A *decision table* is the linearisation of a decision tree, being an alternative way of representing it. Fig. 5, in the left, extends the example of Fig. 3 which now includes a two-dimensional space and the corresponding projections as decision regions ($R1$, ..., $R6$). In the right, these are listed in a decision table with columns specifying the class assigned to each region and the set of values of each variable.



Region	Class	Variables	
		X	Y
R1	T	$[0, x_1)$	$[0, y_3]$
R2	F	$[x_1, x_2)$	$[0, y_1)$
R3	T	$[x_2, x_4]$	$[0, y_1)$
R4	T	$[x_1, x_3)$	$[y_1, y_3]$
R5	F	$[x_3, x_4]$	$[y_1, y_2]$
R6	T	$[x_3, x_4]$	$[y_2, y_3]$

Fig. 5. Converting a Decision Tree to Decision Table

Strecht, Moreira and Soares [4] included an extra column referring to the *weight* of a region, which specifies the proportion of examples used by C5.0 to create a region relative to the local data set. In the implementation of the algorithm that was used, this information is included in each branch of the textual representation of decision trees. The weight is, therefore, a measure of region importance. It is used during the merging process as a strategy to avoid very complex merged models. This is done by only keeping in the merged model the most important regions, i.e., the ones created by a larger number of examples in the original models.

Merge Models. In the second sub-process, two models are merged together which encompasses three sequential tasks. These have been given different designations in the literature. Fig. 6 presents them using Strecht, Moreira and Soares [4] terminology of intersection, filtering and reduction, described next.

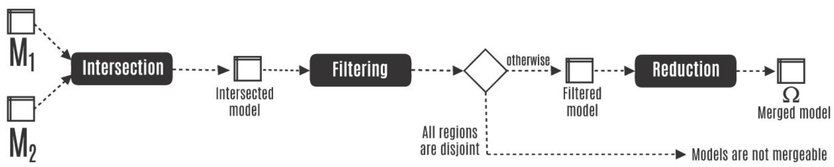


Fig. 6. Sub-process of merging two models

Intersection is a task to combine the regions of two models using a specific method to extract the common components of both, presented in decision table form. The output is the intersected model also in decision table form. The set of values of each region on each model are compared to discover common sets of values across each variable (mapped as a dimension in a multi-dimensional space). The intersection of values of each variable from each pair of regions may have the following outcomes:

– If there are common values, then these are assigned to that variable in the merged region;
– If there are no common values, then they are considered *disjoint regions*, regardless of other variables in which the intersection set may not be empty.

In Hall, Chawla and Bowyer [12] and Strecht, Moreira and Soares [4] approaches, all regions of both models are compared with each other. Bursteinas and Long [2] have a similar method but separate disjoint from overlapping regions. Andrzejak, Langner and Zabala [3] call this operation "unification" and propose a line sweep algorithm to avoid comparing every region of each model. It is commonly based on sorting the limits of each region and then analysing where merging can be done. However, this method only applies to numerical variables.

The class to assign to the merged region is straightforward if the pair of regions have the same class, otherwise the *class conflict problem* arises. Andrzejak, Langner and Zabala [3] propose three strategies to address this problem. The first assigns the class with the greatest confidence, the second, the one with the greater probability and a third strategy, which is the more complex, involves more passes over the data. Hall, Chawla and Bowyer [12] explore the issue in greater detail and propose further strategies, e.g., comparing distances to the boundaries of the variables. However, this approach seems suitable only for numerical variables. Bursteinas and Long [2] use a different strategy by retraining the model with examples for the conflicting class region. If no conflict arises, that class is assigned, otherwise the region is removed from the merged model. Strecht, Moreira and Soares [4] use the weight associated with each region to decide which is the class to be assigned to the merged region. They study both the impact of choosing the class of the region that has the maximum weight and the minimum weight.

Filtering is a task to remove disjoint regions from the intersected merged model yielding the filtered merged model. Andrzejak, Langner and Zabala [3] call this operation "pruning" and developed a ranking system retaining only the regions with the highest relative volume and number of training examples. Hall, Chawla and Bowyer [12] only carry out this phase to eliminate redundant rules created during the removal of class conflicts. Bursteinas and Long [2] mention the phase but do not provide details on how it is performed. Strecht, Moreira and Soares [4] addresses the issue by removing disjoint regions, recalculating the weight of the remaining ones and highlighting the cases where models are *not mergeable* if all regions are disjoint.

Reduction is a task to limit the number of regions in the filtered merged model, to obtain a simpler model. The regions are examined to find out which can be joined into one. This is possible when a set of regions have the same class and all variables have equal values except for one. In the case of nominal variables, reduction consists on the union of values of that variable from all regions. In the case of numerical variables, reduction is performed if the intervals are contiguous. Another consequence of the reduction is that there may exist variables with the same value in all decision regions. The columns for these variables are removed from the table. Both Strecht, Moreira and Soares [4] and Andrzejak, Langner and Zabala [3] perform this operation. It is not mentioned in the researches of Bursteinas and Long [2] and Hall, Chawla and Bowyer [12].

Conversion to decision tree. In the third sub-process, the last merged model of the group (Ω_{n-1}), in decision table form, is converted to the decision tree representation. Andrzejak, Langner and Zabala [3] attempt to mimic the C4.5 algorithm using the values in the regions as examples. One problem with this method is that it is necessary to divide one region in two to perform the splitting, which increases their number, thus making the model more complex. Hall, Chawla and Bowyer [12] do not perform this phase and the merged model is represented as the set of regions. Bursteinas and Long [2] claim to grow a tree but do not describe the method. Strecht, Moreira and Soares [4] grow a tree of the merged model by generating examples provided from each of the decision regions and submitting them as learning examples to the same algorithm used to create the initial models (taking into account the region weights if desired).

3.4 Group models evaluation results

In the fourth process the group models are evaluated using the same evaluation function used to evaluate the original models. Each research evaluates the group models differently. Hall, Chawla and Bowyer [12, 13] compare the accuracy of the merged model with a baseline model trained with all examples. They observed a slight improvement (about 1%) by using the merged model. Andrzejak, Langner and Zabala [3] also use the same baseline case and then compare its accuracy on increasing the number of groups. They observe that creating up to sixteen groups is the limit where the quality of predictions of the merged model still provides a good approximation to the baseline case. Bursteinas and Long [2] compare the classification accuracy of the test set for the combined tree claiming it to be similar to the accuracy generated with the tree induced on the monolithic data set. Strecht, Moreira and Soares [4] method for evaluation is the most different in the literature. As $F1$ was used as evaluation function in each model, $\Delta F1$ is defined as the gain in the predictive performance by using the group model instead of the original model in relation to each local data set. Also, a *merging score* is defined as the number of models that is possible to merge divided by the number of models in a group. They observed that, merging groups across scientific areas yields, in average, an improvement of 3% in prediction quality.

4 Conclusions

The approach of merging models by combining decision rules is the most often found in the literature. However, being specific to each research, it still lacked a general domain-free specification as the one presented in this paper. There is also the lack of a general terminology for concepts that this paper can partially fulfill.

Grouping models is open for further exploration as it is only tackled by two researches. An important issue that is yet to be investigated is the merge order of models within a group. The merging operation is not commutative, therefore it is of great interest that this aspect should be studied in future research as well as its impact in improving the predictive quality. The process of merging models in a group presents great variations in how it is implemented in the literature. While all agree that the most suitable representation to merge models is working with decision tables (although under different designations), the combination of decision rules algorithm (the core of the whole process) is where the major differences are found. The main problem to deal with is class conflict in overlapping rules that has no consensual approach. Efforts to simplify the resulting merged model are always included mainly by attempting to reduce the number of decision rules. The final sub-process of growing a decision tree representation of the merged model also presents challenges and should be further explored in future research. There is still no consistent way to assess the quality of the merged models or which are the best evaluation measures and experimental set-ups. It is also notable the absence of a common baseline case to be used in any study and help comparison of results.

Finally, existing research shows that merging decision trees, despite being an emerging topic, offers interesting prospects which can make it an exciting research area to be further explored on theoretic and application perspectives.

References

1. J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques.* San Francisco: Morgan Kaufmann, 2011.
2. B. Bursteinas and J. Long, "Merging distributed classifiers," in *5th World Multi-conference on Systemics, Cybernetics and Informatics*, 2001.
3. A. Andrzejak, F. Langner, and S. Zabala, "Interpretable models from distributed data via merging of decision trees," *2013 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, Apr. 2013.
4. P. Strecht, J. Mendes-Moreira, and C. Soares, "Merging Decision Trees: a case study in predicting student performance," in *Proceedings of 10th International Conference on Advanced Data Mining and Applications*, pp. 535–548, 2014.
5. B.-H. Yael and T.-T. Elad, "A Streaming Parallel Decision Tree Algorithm," *Journal of Machine Learning Research*, vol. 11, pp. 849–872, 2010.
6. H. Kargupta and B. Park, "A fourier spectrum-based approach to represent decision trees for mining data streams in mobile environments," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, pp. 216–229, 2004.
7. K. Y. Gorbunov and V. a. Lyubetsky, "The tree nearest on average to a given set of trees," *Problems of Information Transmission*, vol. 47, pp. 274–288, Oct. 2011.
8. V. a. Lyubetsky and K. Y. Gorbunov, "Fast algorithm to reconstruct a species supertree from a set of protein trees," *Molecular Biology*, vol. 46, no. 1, pp. 161–167, 2012.
9. F. J. Provost and D. N. Hennessy, "Distributed machine learning: scaling up with coarse-grained parallelism," in *Proceedings of the 2nd International Conference on Intelligent Systems for Molecular Biology*, vol. 2, pp. 340–7, Jan. 1994.
10. F. Provost and D. Hennessy, "Scaling up: Distributed machine learning with cooperation," in *Proceedings of the 13th National Conference on Artificial Intelligence*, pp. 74–79, 1996.
11. G. J. Williams, *Inducing and Combining Multiple Decision Trees.* PhD thesis, Australian National University, 1990.
12. L. Hall, N. Chawla, and K. Bowyer, "Combining decision trees learned in parallel," *Working Notes of the KDD-97 Workshop on Distributed Data Mining*, pp. 10–15, 1998.
13. L. Hall, N. Chawla, and K. Bowyer, "Decision tree learning on very large data sets," *IEEE International Conference on Systems, Man, and Cybernetics*, vol. 3, pp. 2579 – 2584, 1998.
14. C. Blake and C. Merz, "UCI Machine Learning Repository," 1998.
15. J. Quinlan, *C4.5: Programs for Machine Learning.* San Mateo, CA: Morgan Kaufmann, 1992.
16. M. Kuhn, S. Weston, N. Coulter, and R. Quinlan, "C50: C5.0 Decision Trees and Rule-Based Models. R package version 0.1.0-16," 2014.
17. N. Chinchor, "MUC-4 Evaluation Metrics," in *Proceedings of the 4th Message Understanding Conference (MUC4 '92)*, pp. 22–29, Association for Computational Linguistics, 1992.
18. Breiman, Friedman, Olshen, and Stone, *Classification and Regression Trees.* Chapman and Hall/CRC, 1984.