Behavior of Consolidated Trees when using Resampling Techniques

Jesús Mª Pérez, Javier Muguerza, Olatz Arbelaitz, Ibai Gurrutxaga, Jose I. Martin

Dept. of Computer Architecture and Technology, University of the Basque Country
M. Lardizabal, 1, 20018 Donostia, Spain

Abstract. Many machine learning areas use subsampling techniques with different objectives: reducing the size of the training set, equilibrate the class imbalance or non-uniform cost error, etc. Subsampling affects severely to the behavior of classification algorithms. Decision trees induced from different subsamples of the same data set are very different in accuracy and structure. This affects the explanation of the classification; very important in some domains. This paper presents a new methodology for building decision trees. The final classifier is a single decision tree, so that it maintains the explaining capacity of the classification. A comparison in error and structural stability of our algorithm and the C4.5 algorithm is done. The decision trees generated using the new algorithm, achieve smaller error rates and structurally more steady trees than C4.5 when using subsampling techniques.

1 Introduction

Many machine learning areas use subsampling techniques with different objectives. The first example could be the construction of a multiple classifier able to obtain larger accuracy in the classification [3],[9],[11],[6],[1]. Another application of subsampling can be the reduction of the size of the data base, so that it becomes usable for the corresponding machine learning algorithm [17],[4]. Probably one of the most important applications of subsampling is to use it in order to equilibrate the class distribution, in databases with class imbalance [17],[12]. There are many areas where cases of one of the classes can be difficult to obtain —medicine, fraud detection, etc.—, this leads very often to class imbalance in the data set which in general, does not even coincide with the natural distribution or the case distribution of the data expected in reality. Another similar case is the one of data sets with non-uniform cost, so that the cost of errors is not the same for the whole confusion matrix. The use of subsampling techniques to make some errors become more important than others can be a way of introducing such a cost in the learning algorithm when the algorithm does not take into account the cost-matrix in the induction process —oversampling or undersampling— [8].

Mª Pérez J., Muguerza J., Arbelaitz O., Gurrutxaga I. and I. Martin J. (2004).
Behavior of Consolidated Trees when using Resampling Techniques.
In *Proceedings of the 4th International Workshop on Pattern Recognition in Information Systems*, pages 139-148
DOI: 10.5220/0002665601390148
Copyright © SciTePress

In all the mentioned cases the initial data set is subsampled to build one or several subsamples. These will be given to the learning algorithm in order to build a classifier. This subsampling affects severely the behavior of the classification algorithms [12]. Decision Trees (DT) are not an exception. Decision trees induced from different subsamples of the same data set, are very different in accuracy and structure [7]. This last feature needs to be taken into account because in many domains it is as important as the first one. There are areas such as illness diagnosis, fraud detection in different fields, customer's behavior analysis (marketing), etc. where the algorithm used to make the classification has to provide an explanation about the decision made. In this kind of problems, the coming actions will depend on the explanation given by the classifier. This makes important the fact that the structure of the induced classifier does not vary excessively due to the subsampling.

This paper presents a new methodology for building decision trees. The meta algorithm can be used when subsampling has to be done due to any of the reasons mentioned before. A comparison in error and structural stability of our algorithm and the C4.5 algorithm [15] is presented in this work. The results will show that the decision trees generated using the new algorithm, achieve smaller error rates than C4.5. The classification is made in a different way bagging and boosting do it —where the final classification is made based on a set of base classifiers (for example DT)—. In our system the final classifier is a single decision tree, so that it maintains the explaining capacity of the classification. The structural analysis presented in this work proves that the new algorithm has a more steady behavior than C4.5. It is more steady, less complex —except in the cases where it manages to situate in a better position in the learning curve [10]— and it obtains a smaller error rate in most of the analyzed domains —giving this way a better quality to the explanation—.

The paper proceeds with the description of our new methodology for building decision trees, Section 2. In Section 3, the description of the data set and the experimental set-up is presented. The results of our experimental study are discussed in Section 4. Section 5 is devoted to comment further work. Finally, Section 6 summarises the conclusions.

2 Consolidated Trees' construction algorithm

Consolidated Trees Construction algorithm (CTC), is based on resampling techniques [13],[14]. This technique is radically different from bagging and boosting; the consensus is achieved at each step of the trees' building process and only one tree is built. All the trees being built (using the base classifier: C4.5 in our case) from the different subsamples, make proposals about the variable that should be used to split in the current node. The decision about which variable will be used to make the split in a node of the consolidated tree (CT) is accorded among the different trees. The decision is made by a voting process level by level. Based on this decision, all the trees (each one associated to a different subsample) are forced to use the same variable to make the split. The process is repeated iteratively until some stop criterion is fulfilled.

The algorithm starts extracting from the original training set a set of subsamples (*Number_Samples*) to be used in the process. The subsamples are obtained based on the desired resampling technique (*Resampling_Mode*). The consolidation of nodes, in the CT's construction process, is made step by step the following way:

- The variable that would be used to make the split at this level of the CT tree is suggested by each tree.
- The number of trees that propose to make an split is counted and the decision is made depending on the established criteria, *Crit_Split* (ex: absolute majority, ...).
- If the decision is to split, the most voted variable is selected.
- *Crit_Branches* criteria is used to decide the branches the node to split will have.
- The accorded split is forced (variable and stratification) in every tree.

The used subsampling technique and the number of subsamples used in the tree's building process are important aspects of the algorithm [16]. There are many possible combinations for the *Resampling_Mode*: size of the subsamples —100%, 75%, 50%, etc; of the original training set —, with replacement or without replacement, stratified or not, etc. The options presented in this work are 75% and 50% without replacement.

Related to the *Crit_Branches* criteria, if the variable to split is continuous, the cutting point has to be determined (ex: using the mean or the median of the values proposed for that variable). But, when the variable to split is discrete, a set of categories for each branch has to be selected (ex: a branch for each category, using heuristics such as C4.5's subset option, etc.). Both kind of trees have been pruned using the pruning algorithm of the C4.5 R8 software, to situate both systems in a similar zone in the learning curve. We can not forget that developing too much a classification tree leads to a greater probability of overtraining.

Once the consolidated tree has been built, its behavior is similar to the behavior of a single decision tree. Section 4 will show that the trees built using this methodology, have better discriminating capacity and they are less complex.

3 Experimental methodology

Twenty databases of real applications have been used for the experimentation. Most of them belong to the well known UCI Repository benchmark [2], widely used in the scientific community. The Segment domain has been used for experimentation in two different ways: taking into account the whole set of data (*segment2310*) and respecting the training/test division of the original data set. The *Faithful* database is a real data application from our environment, centered in the electrical appliance's sector. In this case, we try to analyze the profile of the customers during the time, so that a classification related to their fidelity to the brand can be done. In this kind of applications, the use of a system that provides explanation is very important; it is nearly more important to analyze and explain why a customer is or is not faithful, than the own categorization of the customer.

The CTC methodology has been compared to the C4.5 tree building algorithm Release 8 of Quinlan [15], using the default parameter settings. The methodology used for the experimentation is a 10-fold stratified cross validation [10].

In order to compare the behavior of the two algorithms the same procedure has been used in each of the folds of the cross-validation: 100 subsamples have been extracted, always without replacement and with sizes of 75% and 50% (*Resampling_Mode*). These subsamples have been used to build both kinds of trees, CT and C4.5. Being the aim of this work just the analysis of the effect of subsampling in different kinds of trees, the generated subsamples maintain the class distribution of the original training set (stratified samples).

In each case we have calculated the error and the complexity of the trees. The complexity is estimated as the number of internal nodes of the tree. A structural distance among the trees that are being compared has been defined. This structural measure is based on a metric that takes into account the variability in the nodes of a set of trees (the measure is a pair to pair comparison among all the trees of the set). If two nodes will be counted as common nodes, they have to coincide in the variable used to make the split, the stratification and the position in the tree. The value appearing in the results presented in Section 4, is *Common*. It refers to the nodes of the tree, that coincide in the whole set of compared trees. This value is calculated starting from the root and covering the whole tree, level by level. So, the measure takes into account the rank of importance the algorithm has given to each one of the variables appearing in the tree —the nearer the variable is from the root of the tree, more importance has it in the classification— when making the comparison. A more formal definition of the structural metric can be found in [14]. This metric quantifies the homogeneity of a set of trees. If the difference among the trees comes from the subsample used to induce them (*Resampling_Mode* and *Number_Samples*), we are measuring the influence of the different resampling modes in the induction algorithm.

From a practical point of view, *Common* quantifies the explaining capacity and stability of the classification tree, whereas the error would quantify the "quality" of such an explanation. Evidently it is not enough to have a greater value in *Common* to have greater explaining capacity; it is compulsory to have a similar or smaller error rate.

In order to evaluate the improvement achieved in a given domain by using the algorithm CTC compared to the algorithm C4.5, we calculate the relative improvement or relative difference for the error (R.Dif) and the *Common*. For every result we have tested the statistical significance [5],[6] of the differences of the results obtained with the two algorithms using the paired t-test with significance level of 95% and 90%.

4 Results

This section is devoted to present the results of different comparisons made among the two algorithms (C4.5 and CTC). For the CTC, all the possible trees without repeating none of the 100 subsamples have been built. This has lead to different number of instances of CTs when varying the parameter *Number_Samples (N_S)*: 5 (20 trees), 10 (10 trees), 20 (5 trees), 30 (3 trees), 40 (2 trees) and 50 (2 trees).

The kind of problems mentioned in the introduction can be faced undersampling ($C4.5_{100}$) or oversampling ($C4.5_{union}$) the original dataset. Both options have been compared in the experimentation to CTC. The comparison with C4.5 trees induced directly from the whole fold ($C4.5_{not\ resampling}$) is also done and presented later.

4.1 CT versus C4.5$_{100}$ and C4.5$_{union}$

100 trees (one with each subsample) have been built for the C4.5 and the averages of the obtained results (C4.5$_{100}$) are compared to CTs. The graphics in Fig. 1 show the mean of the error and *Common* obtained with both algorithms for all the databases. The horizontal axe represents the different values of the parameter *N_S* we have tried (5, 10, 20, 40 and 50). Continuous lines show the results (error and *Common*) obtained with *Resampling_Mode* 75%, and dotted lines the ones obtained with 50%.

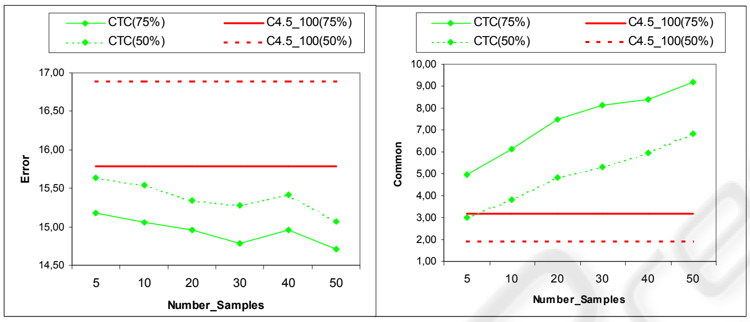

Fig. 1. Average results of the Error (left) and the Common (right) for CTC and C4.5$_{100}$

It can be observed that the results obtained with CTC are better in error than the ones obtained with C4.5$_{100}$, for both values of the parameter *Resampling_Mode*: 75% and 50%. CTC algorithm has also greater explaining capacity and is more steady than C4.5$_{100}$. The results with 75% are in average better than the ones achieved with 50%.

The analysis of the previous results shows that the behavior of CTC is better than the behavior of C4.5 when subsampling has to be used for building decision trees.

The difference in behavior among the two algorithms can be due to the amount of information of the original training set “seen” by each one. The CTC building process is based on a set of subsamples. A CT “sees” more information than a C4.5 tree. This has lead us to design another comparison, where the way the subsamples have been used to induce the C4.5 trees, has been changed. In this experimentation as many C4.5 trees as CTs will be built. The sample used to induce each one of the C4.5 trees (C4.5$_{union}$) will be the union of the subsamples used to build the corresponding CT. So, in this experimentation the information “seen” by both algorithms is the same. Fig. 2 and Table 2 show that results for C4.5$_{union}$ are considerably worse than for CTC. Moreover, when *Resampling_Mode* is 75%, the error rates get with C4.5$_{union}$ are even greater than the ones obtained with C4.5$_{100}$.

Even if the mean of the results get for every data set with C4.5$_{union}$ and C4.5$_{100}$ are worse than the ones get with CTC, it is important to underline that there are some domains where the behavior of the C4.5$_{union}$ (or C4.5$_{100}$) is better than the behavior of CTC. In Table 1 it can be observed that for the domain *Sick-E* the best results are achived by the C4.5$_{100}$. Looking to Table 2 we can say that the best results for the domain *Soybean-L* are achived by C4.5$_{union}$.

Going back to the information Fig. 1 and Fig. 2 provide us, we can observe that the behavior of the CTC algorithm related to both aspects, error and *Common*, improves when the value of *N_S* increases. Even though for the whole range studied, the results for CTC (error and structure) are better than the ones obtaines for any of the versions of C4.5 Although the best average results have been obtained when *N_S* = 50, taking into account the trade-off among the obtained results and the computational cost of the CTC building process, it seems reasonable to use the value 30 for the *N_S* parameter.

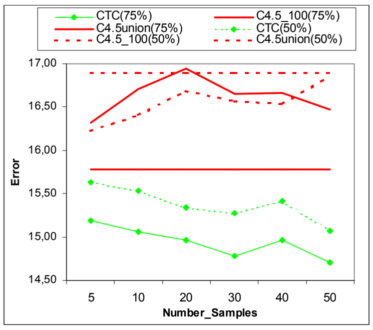

Fig. 2. Average results of the Error for CTC and C4.5$_{union}$ (C4.5$_{100}$ is also added as reference)

Table 1. Results of Error and *Common* for every domain. For each domain, CTC (*N_S* = 30) and C4.5$_{100}$, with *Resampling_Mode* = 75% and 50% appear

	Resampling_Mode = 75%						Resampling_Mode = 50%					
	Error			Common			Error			Common		
	CTC	C4.5$_{100}$	R.Dif	CTC	C4.5$_{100}$	R.Dif	CTC	C4.5$_{100}$	R.Dif	CTC	C4.5$_{100}$	R.Dif
Breast-W	5,89	6,30	-0,06	2,90	1,71	0,70	5,70	7,05	-0,19	2,53	1,12	1,27
Heart-C	22,33	23,92	-0,07	6,87	1,43	3,80	22,57	25,51	-0,12	3,13	0,71	3,41
Hypo	0,76	0,79	-0,04	4,17	2,67	0,56	0,95	0,87	0,10	2,83	2,11	0,34
Lymph	18,27	21,12	-0,14	8,90	2,22	3,00	19,09	23,06	-0,17	5,30	0,69	6,69
Credit-G	26,97	27,75	-0,03	14,00	2,34	4,99	27,53	28,81	-0,04	5,67	1,19	3,75
Segment210	10,80	11,98	-0,10	5,03	2,09	1,41	12,21	14,16	-0,14	4,67	1,70	1,75
Iris	4,23	6,38	-0,34	2,67	2,05	0,30	5,12	6,57	-0,22	2,00	1,62	0,23
Glass	29,18	32,04	-0,09^T	7,37	2,65	1,78	27,30	35,11	-0,22	5,73	1,48	2,87
Voting	3,36	4,18	-0,20	4,60	2,19	1,10	3,75	4,82	-0,22	3,80	1,37	1,77
Hepatitis	19,30	20,45	-0,06	3,17	0,84	2,76	19,28	20,29	-0,05	2,80	0,33	7,59
Soybean-L	14,86	16,10	-0,08	16,20	4,62	2,50	14,90	20,50	-0,27	9,60	2,16	3,44
Sick-E	2,34	2,18	0,07	8,90	4,77	0,86	2,49	2,42	0,03	5,07	3,31	0,53
Liver	34,50	36,11	-0,04	9,27	1,18	6,84	35,45	37,32	-0,05	4,90	0,50	8,89
Credit-A	15,20	14,86	0,02	5,77	2,11	1,73	15,48	15,22	0,02	5,13	1,52	2,37
Vehicle	27,40	28,58	-0,04	16,87	7,13	1,37	28,85	30,18	-0,04	9,93	3,48	1,85
Breast-Y	26,36	28,11	-0,06	2,20	0,71	2,11	28,42	29,11	-0,02	2,67	0,51	4,19
Heart-H	21,96	21,61	0,02	5,27	1,37	2,85	23,59	21,62	0,09	4,73	0,90	4,26
Segment2310	3,23	3,96	-0,18	22,20	10,39	1,14	3,51	5,07	-0,31	14,97	6,61	1,26
Spam	7,29	7,68	-0,05	13,87	4,50	2,08	7,74	8,50	-0,09	8,47	2,06	3,12
Faithful	1,47	1,52	-0,03	2,67	6,58	-0,59	1,54	1,69	-0,09	2,00	5,23	-0,62
Average	14,79	15,78	-0,07	8,14	3,18	2,06	15,27	16,89	-0,10	5,30	1,93	2,95

Table 1 shows the results of the comparison of CTC (with *N_S* = 30) and C4.5$_{100}$. In the case of 75%, the table shows that in 17 domains out of 20, the error is smaller for CTC than for C4.5$_{100}$. The differences are statistically significant in 9 databases —8 with a confidence level of 95% (marked with) and 1 with a 90% (marked with ^{T})—. In the databases where results for C4.5$_{100}$ are better, the differences are never significant. In this situation, it is worth the comparison of the explaining capacity of the different classifiers. The data show that the CTs achieve a higher explanation level than $C4.5_{100}$ (in average 8,14 compared to 3,18) and besides, it is steadier. There is an exception in *Faithful* database. This happens because the complexity of $C4.5_{100}$ trees is an order of magnitude larger, but, error is smaller for CTC and the difference is significant. From this point of view of the comparison, statistically significant differences are obtained in every database. The analysis for 50% gives us similar results. The mean obtained for both algorithms tends to slightly increase the error and decrease de *Common*. Anyway the results obtained with CTC are better than those obtained with $C4.5_{100}$, with significant differences for the error in 9 of de data sets and in all but two for the *Common*.

Table 2. Results of Error for every domain. For each domain, CTC (*N_S* = 30) and $C4.5_{union}$, with *Resampling_Mode* = 75% (left) and 50% (right) appear.

	Error					
	Resampling_Mode = 75%			Resampling_Mode = 50%		
	CTC	$C4.5_{union}$	R.Dif	CTC	$C4.5_{union}$	R.Dif
Breast-W	5,89	7,10	-0,17	5,70	7,25	-0,21
Heart-C	22,33	27,38	-0,18	22,57	28,16	-0,20
Hypo	0,76	1,26	-0,40	0,95	1,25	-0,24
Lymph	18,27	25,00	-0,27	19,09	23,07	-0,17
Credit-G	26,97	31,90	-0,15	27,53	31,60	-0,13
Segment210	10,80	10,41	0,04	12,21	10,41	0,17
Iris	4,23	6,02	$-0{,}30^{T}$	5,12	5,79	-0,12
Glass	29,18	29,00	0,01	27,30	28,82	-0,05
Voting	3,36	4,59	-0,27	3,75	4,82	-0,22
Hepatitis	19,30	20,92	-0,08	19,28	21,85	-0,12
Soybean-L	14,86	12,91	0,15	14,90	11,92	0,25
Sick-Ed	2,34	2,95	$-0{,}21^{T}$	2,49	2,88	-0,13
Liver	34,50	36,95	-0,07	35,45	35,98	-0,01
Credit-A	15,20	18,19	-0,16	15,48	17,77	$-0{,}13^{T}$
Vehicle	27,40	26,93	0,02	28,85	27,04	0,07
Breast-Y	26,36	35,50	-0,26	28,42	34,07	$-0{,}17^{T}$
Heart-H	21,96	22,85	-0,04	23,59	25,19	-0,06
Segment2310	3,23	3,23	0,00	3,51	3,51	0,00
Spam	7,29	7,77	-0,06	7,74	7,67	0,01
Faithful	1,47	2,35	-0,37	1,54	2,42	$-0{,}36^{T}$
Average	14,79	16,66	-0,14	15,27	16,57	-0,09

Table 2 shows the error related comparison among $C4.5_{union}$ and CTC.

In some previous studies [13],[14], we made a third comparison among C4.5 and CTC algorithm. In this case the CTs have been built using the same methodology described in this section. But, the C4.5 trees have been built directly from the training data belonging to each fold of the 10-fold cross-validation, without using any resampling technique. In this experimentation, the C4.5 algorithm uses all the existing information in each fold, that is, the same amount or more information than the CTC will see. Next subsection presents a summary of the obtained results.

4.2 CT versus $C4.5_{not\text{-}resampling}$

We can not forget that the comparison presented in this subsection does not solve the problem described in the introduction of this article; the cases where subsampling is compulsory due to different reasons —class imbalance, sample's size, etc.—. The aim of including these results in the paper is to add one more comparison where both algorithms induce the decision trees parting from the same information. In this case C4.5$_{\text{not resampling}}$ will use all the information of the fold but without repeated cases which makes it different from C4.5$_{\text{union}}$.

The C4.5$_{\text{not resampling}}$ reaches in this experiment the best results from the three possibilities —C4.5$_{100}$, C4.5$_{\text{union}}$ and C4.5$_{\text{not resampling}}$— (see Fig. 3).

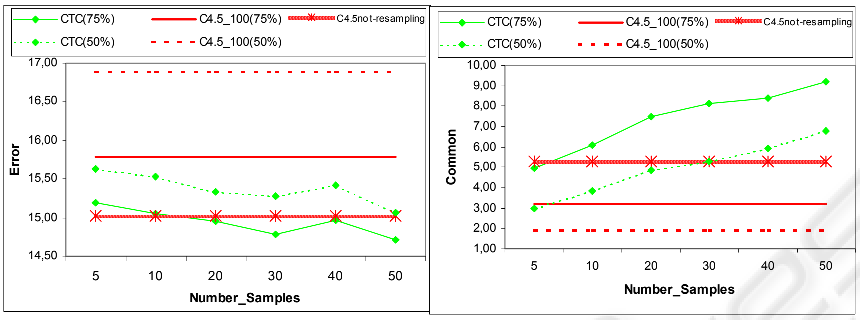

Fig. 3. Average results for Error (left) and *Common* (right) for CTC and C4.5$_{\text{not resampling}}$

The error rate tends to be smaller in CTC for both values of *Resampling_Mode* as the value of *N_S* parameter is increased. The differences with the error in C4.5$_{\text{not resampling}}$ are not statistically significant in any of the domains. Moreover, from the structural point of view, the behavior of CTC is better.

Therefore, we can say that decision trees induced with the CTC meta-algorithm, when *Resampling_mode*e is 75% have a lower error rate than those induced with C4.5$_{\text{not resampling}}$, and they are structurally steadier. As a consequence they provide a wider and steadier explanation, that allows to deal with the problem of the excessive sensitivity decision trees have, to subsampling methods.

5 Future research

The results in error and structural stability lead us to think that the increase of the parameter *N_S* could help to achieve better quality results. Anyway, the graphics show that the CTC is not very sensitive to changes in *N_S*. So, the increase of the computational cost that this parameter can produce must be considered. Analysis of the results obtained for both algorithms with other percentage values for the *Resampling_Mode* parameter can also be interesting.

We have observed that decision trees induced with the CTC algorithm tend to converge to a single tree beyond a specific value of the *N_S* parameter (about 30). Fig. 4 shows the convergence for *Breast-W* domain. The value appearing in Fig. 4 (*%Common*) is calculated by normalizing the *Common* value in respect to the trees' size. The curves represent the values of *%Common* in each one of the folds of the cross-validation. The use of different subsamples to build a C4.5 tree has in the structure the same effect mentioned in Section 4.1 for the CTC; the value of *Common* increases when *N_S* is augmented. But, Fig. 4 shows that the CTs converge to an unique tree after a certain value of *N_S* (left), whereas C4.5 trees (right) show a greater structural variation. This study will reduce the possible values *N_S* parameter can take.

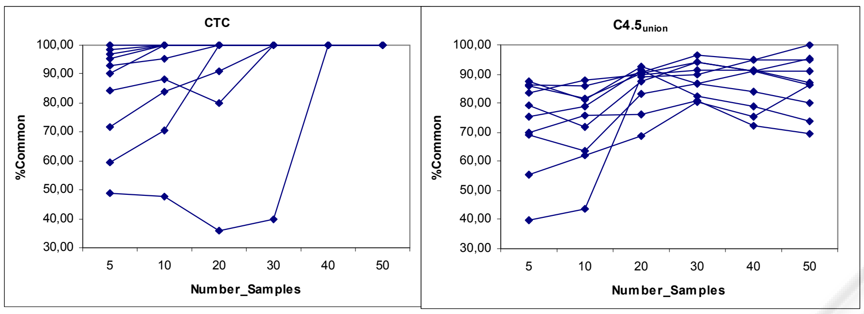

Fig. 4. Structural convergence of the CTC and the C4.5$_{\text{union}}$ for the *Breast-W* domain

The CTC meta-algorithm provides a way to deal with the need of resampling the training set, either for a class imbalance problem or a size problem. Anyway, we are working in quantifying the influence that changes in the class distribution can have in the CTC algorithm. It would also be interesting the comparison of the results obtained with other techniques that use resampling in order to improve the accuracy of the classifier, such as bagging, boosting, etc. We can not forget that this techniques lack the explaining capacity the CTC has. We also have in mind to use different parallelization techniques (shared memory and distributed memory computers), to accelerate the building process of CTC.

6 Conclusions

In this work we have presented a new decision tree construction methodology (CTC) able to afford a problem that standard decision trees suffer: unsteadiness when small changes in the training set happen. There are two usual situations where the training set has to be modified: class imbalance and large data sets.

In the proposed methodology the needed subsampling technique is introduced inside the induction algorithm, because subsampling is inherent in the CTC algorithm. The behavior of the CTC algorithm has been compared to C4.5 in three different ways, and for twenty databases. The obtained decision trees achieve smaller error rates and larger structural stability, so a steadier explanation level. This is essential for some specific domains (medical diagnosis, fraud detection, etc.)

Acknowledgments

The work described in this paper was partly done under the University of Basque County (UPV/EHU) project: 1/UPV 00139.226-T-14882/2002. We would like to thank the company Fagor Electrodomesticos, S. COOP. for permitting us the use of their data (*Faithful*) obtained through the project BETIKO. The *lymphography* domain was obtained from the University Medical Centre, Institute of Oncology, Ljubljana, Yugoslavia. Thanks go to M. Zwitter and M. Soklic for providing the data.

References

1. Bauer E., Kohavi R.: An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants, Machine Learning, Vol. 36, (1999) 105-139
2. Blake, C.L., Merz, C.J.: UCI Repository of Machine Learning Databases, University of California, Irvine, Dept. of Information and Computer Sciences. http://www.ics.uci.edu/~mlearn/MLRepository.html (1998)
3. Breiman L.: Bagging Predictors. Machine Learning, Vol. 24, (1996) 123-140.
4. Chan P.K., Stolfo S.J.: Toward Scalable Learning with Non-uniform Class and Cost Distributions: A Case Study in Credit Card Fraud Detection, Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining, (1998) 164-168.
5. Dietterich T.G.: Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms, Neural Computation, Vol. 10, No. 7, (1998) 1895-1924.
6. Dietterich T.G.: An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization, Machine Learning, Vol. 40, (2000) 139-157.
7. Drummond C., Holte R.C.: Exploiting the Cost (In)sensitivity of Decision Tree Splitting Criteria, Proceedings of the 17th International Conference on Machine Learning, (2000) 239-246.
8. Elkan C.: The Foundations of Cost-Sensitive Learning, Proceedings of the 17th International Joint Conference on Artificial Intelligence, (2001) 973-978.
9. Freund, Y., Schapire, R. E.: Experiments with a New Boosting Algorithm, Proceedings of the 13th International Conference on Machine Learning, (1996) 148-156.
10. Hastie T., Tibshirani R. Friedman J.: The Elements of Statistical Learning. Springer-Verlang (es). ISBN: 0-387-95284-5, (2001).
11. Ho T.K: The Random Subspace Method for Constructing Decision Forests, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-20, N. 8, (1998) 832-844.
12. Japkowicz N.: Learning form Imbalanced Data Sets: A Comparison of Various Strategies, Proceedings of the AAAI Workshop on Learning from Imbalanced Data Sets, Menlo Park, CA, (2000)
13. Pérez J.M., Muguerza J., Arbelaitz O., Gurrutxaga I.: A new algorithm to build consolidated trees: study of the error rate and steadiness, accepted in the conference on Intelligent Information Systems, Zakopane, Poland, (2004)
14. Pérez J.M., Muguerza J., Arbelaitz O., Gurrutxaga I.: Consolidated Tree Construction Algorithm: Structurally Steady Trees, submitted to the. 6th International Conference on Enterprise Information Systems, Porto, Portugal, (2004).
15. Quinlan J. R.: C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers Inc.(eds), San Mateo, California.
16. Skurichina M., Kuncheva L.I., Duin R.P.W., 2002. Bagging and Boosting for the Nearest Mean Classifier: Effects of Sample Size on Diversity and Accuracy, Lecture Notes in Computer Science Vol. 2364. Multiple Classifier Systems: Proc. 3th. Inter. Workshop, MCS 2002., Cagliari, Italy, (1993) 62-71.
17. Weiss G.M., Provost F.: Learning when Training Data are Costly: The Effect of Class Distribution on Tree Induction, Journal of Artificial Intelligence Research, Vol. 19, (2003) 315-354.