Pessimistic decision tree pruning based on tree size

Yishay Mansour
Computer Science Dept.
Tel-Aviv University
Tel-Aviv, ISRAEL
mansour@math.tau.ac.il

Abstract

In this work we develop a new criteria to perform pessimistic decision tree pruning. Our method is theoretically sound and is based on theoretical concepts such as uniform convergence and the Vapnik-Chervonenkis dimension. We show that our criteria is very well motivated, from the theory side, and performs very well in practice. The accuracy of the new criteria is comparable to that of the current method used in C4.5.

1 Introduction

The phenomena of overfitting the data is well known in machine learning, and refers to the case that the learned hypothesis is so closely related to the training examples such that its generalization capabilities would be penalized. Overfitting would usually occur when the class of hypotheses used is as complex as the given training sample. For this reason we would like, in many cases, to limit the hypothesis we generate to be "less complex" than the training sample.

In decision trees the overfitting phenomena can occur when the size of the tree is too large compared to the number of training examples. As an extreme case, for a given training sample consider an arbitrary decision tree where each leaf contains at most a single example from the training sample. By setting the value of the leaf to be the label of the example that reaches it, we are guarantee to have no error on the training sample. However, the generalization capabilities of such a tree are very doubtful. For this reason we are interested not only in trees whose error on the training sample is small, but also those whose size is small compare to the sample size.

There are two general methods, in decision tree induction, to bound the size of the tree. The first is to have some criteria, called *Early Stopping* criteria, that would be used in order to decide whether or not to stop growing a given node. Such an early stopping criteria limits the size of the tree and avoids trying to grow unnecessary sub-trees. The risk in using an early stopping criteria is that by not growing a sub-tree we might be missing some real structure of the problem that could be discovered only later.

The second method is growing the initial tree until it fits the sample perfectly. Then we use this tree as our initial starting point and start *pruning* it. Generally, pruning out performs early stopping, since it allows us to discover some structure that is not immediate. There are a few different methods of performing the pruning and they all output at the end of the pruning process a tree which is a pruned version of the original tree.

There are two different methodologies for performing pruning. The first is of generating a sequence of pruned trees, and later to chose one of them. This idea is the core idea behind the Minimal Cost-Complexity Pruning proposed by Breiman et. al. [BFOS84]. They proposed using dynamic programming to find a sequence of trees that minimizes a combination of the error on the training sample and the tree size. Later, an independent sample is used to chose the best tree out of the sequence. In [BB94] a simple dynamic programming algorithm is given to find the smallest pruned decision tree that has a given error rate. Here again, one can generate a sequence of trees and use an independent sample to chose the best tree in the sequence.

The second methodology for pruning tries to directly prune the original decision tree. The pruning is done by replacing the sub-tree rooted at a node by either a leaf or by a sub-tree rooted at one of its children.

The approach of Reduced Error Pruning, proposed by Quinlan [Qui87], uses an independent sample to test the accuracy of each sub-tree compared to the case when it is pruned. Mingers [Min87] suggested the Critical Value Pruning, which performs the decision about pruning using on the "information-gain" that was achieved in growing the tree. (For each node it computes the maximum "information-gain" in its subtree, and prunes a node if this value is less than a certain threshold.)

Under the category of pessimistic pruning we include all the pruning criterias that try to bound the true error of a sub-tree based on its training error. We perform the pruning at a node when the reduction in the error due to the sub-tree seems statistically insignificant. The algorithmic benefit of the pessimistic pruning methods is that they require only a single pass over the tree to generate the pruned tree (compared to the quadratic time of the dynamic programming based methods).

The main question now is how do we decide if the change in the error is significant. We try to estimate (or at least to upper bound) the true error of a sub-tree, based on the number of examples it classifies, the number of errors, the number of classes, and the size of the tree.

One method that falls into our category of pessimistic pruning is *Minimum error* pruning [Nib87]. This method tries to estimate the true error using the following formula,

$$\frac{e + (r-1)}{n + r}$$

where r is the number of classes, n the number of examples, and e is the number of errors. (The theoretical justification for the formula is simple. Assuming a uniform prior over the classes, the above is the expected error over the posterior distribution, given that in n examples there where e errors.)

Pessimistic pruning of Quinlan [Qui86, Qui93] uses a complex method based on the number of errors and the size of the training sample. However, the theoretical justification of this method is rather weak, or as Quinlan says in his book "... the reasoning should be taken with a large grain of salt" [Qui93]. This method of pessimistic pruning is the method that is used currently in C4.5.

An excellent survey and empirical comparison of the different pruning methods was done by Mingers [Min89], which also includes a survey of all the different methods.

Our starting point can be motivated as follows. Assume you have a sub-tree that classifies 200 training examples and makes no error. We would very much like to differentiate the case that the sub-tree is of a small size (say 3) or a huge size (say 100). In the former case we expect that the true error would be very small, while in the latter case the true error might be huge. This is a most compelling argument why the size of the decision tree should be incorporated in the pessimistic estimate.

Our basic proposal it to estimate the error of a sub-tree on the training sample by the following quantity,

$$\frac{e}{n} + c_1 \cdot \sqrt{\frac{k \log d + c_2}{n}},$$

where n is the sample size that reaches the sub-tree, e is the number of incorrect classifications done by the sub-tree, k is the size of the sub-tree, d the number of attributes and c_1 and c_2 are some constants. We show, using basic theoretical arguments, that the above expression is an upper bound on the true error of the sub-tree.

When considering pessimistic pruning one should question how "pessimistic" is it. The main point to keep in mind is that the generated decision tree is not arbitrary, but rather the one that reduces the error on the training examples. For example, say we have a set of functions all with the same true error. On any given set of examples, we expect to have some function, from the set of functions, whose error on this set of examples is lower than the true error. This is why our pessimistic estimates are really realistic estimates in many cases.

We have performed empirical tests to compare the current method of C4.5 and our new proposed method. We start with an artificial data on which we show that our criteria is superior to the current method of C4.5 (even when modifying $C4.5$'s confidence parameter). We then continue and perform an empirical evaluation of six datasets. Since our theory was design for boolean output, we concentrated on datasets where the output is boolean. The comparison shows that the accuracy of the new criteria is comparable to that used in C4.5. (However, it enjoys being theoretically sound.)

We show that in our method we can use the parameter c to control the size of the pruned trees. A larger value of c will cause more pruning, while a smaller value would limit the pruning. (This is similar to the confidence parameter build in C4.5, which also controls the amount of pruning.)

From our empirical results it seems that very significant pruning generates much smaller decision trees and maintains the accuracy. In [AHM95] the accuracy of C4.5 was compared to an algorithm that generates decision trees of depth two. There it was observed that decision trees of depth two achieve remarkably good results compared to C4.5, although in some cases the limitation to depth two forced reduced accuracy. In this work we observed that our criteria can be used to perform extreme pruning of the decision tree while maintaining its accuracy. This might lead one to suspect that the complex decision trees, that are in many cases generated by the default setting of C4.5, have their predictive power concentrated in a small sub-tree. We may hope that by extreme pruning we will recover that sub-tree.

This work is organized as follows. In Section 2 we derive the theoretical bounds. Section 3 describes our criteria in detail. In Section 4 we describe the empirical results.

2 Theoretical Background

In this section we derive from basics the bounds that justify our criteria. More elaborate techniques may give superior bounds, but we did not try to peruse that direction.

We are interested in the maximum deviation of the error when the hypothesis is a binary decision tree with k nodes. Given this we will be able to bound the true error by the error on the sample plus the maximum deviation. The first technical lemma that we derive bounds the number of possible decision trees.

Lemma 2.1 *The number of binary decision trees over d binary inputs with boolean output and $k = 2k' - 1$ nodes is bounded by, is*

$$O(2^k) \cdot d^{k'-1} \cdot 2^{k'} = O((4d)^k) = O((4d)^k).$$

Proof: In a binary decision tree, each internal node has degree two (otherwise it is redundant). Therefore, a binary decision tree with $k = 2k' - 1$ nodes, has k' leaves and $k' - 1$ internal nodes. Each leaf can be label by either zero or one, therefore the number of assignments to the leaves is at most $2^{k'}$. Each internal node can be labeled by any of the inputs, therefore there are at most $d^{k'-1}$ ways to label the internal nodes. The number of different binary trees can by bounded by $O(2^k)$ (proof omitted). Multiplying the three quantities derives the lemma. □

The next step is to bound the deviation of a fixed function from its expectation. This can be derived by using the Chernoff inequality.

Lemma 2.2 (Chernoff) *Let $x_1, \ldots, x_n$ be independent identically distributed random variables, where $x_i \in \{0, 1\}$ and $Prob[x_i = 1] = p$. Let $\hat{p}_n = \sum_{i=1}^n x_i/n$, then:*

$$Prob[|\hat{p}_n - p| \geq \lambda] \leq 2e^{-2\lambda^2 n}$$

By combining the two we can derive the following theorem,

Theorem 2.3 *Let $\mathcal{H}$ be the class of binary decision trees over d binary inputs with boolean output and at most k nodes. Let f be any boolean function and D be any probability distribution. For $h \in \mathcal{H}$ let $e(h, f) = Prob_{x \sim D}[h(x) \neq f(x)]$. Given $z_1, \ldots, z_n$ let $\hat{e}(h, f)$ be the fraction of z_i's for which $f(z_i) \neq h(z_i)$.*

$$Prob_{z_1,\ldots,z_n \sim D} \quad [\exists h \in \mathcal{H} \ : \ |e(h,f) - \hat{e}(h,f)| > c\sqrt{\frac{k \log d + \log 1/\delta}{n}} \ ] \leq \delta$$

for some constant c.

What the above theorem is saying can be interpreted as follows. For any decision tree with k nodes, when tested on n examples, the deviation between the observed error (on the sample) and the true error is bounded by $O(\sqrt{\frac{k \log d}{n}})$, where we take δ to be a constant. Such a result, in the theory terminology, is a uniform convergence result, since the error of *all* the hypotheses in $\mathcal{H}$ converge simultaneously. This means that if a given decision tree h with k nodes has an error $\hat{e}(h, f)$ on a set of n examples, its true error, $e(h, f)$, with high probability, is bounded by,

$$e(h, f) < \hat{e}(h, f) + c\sqrt{\frac{k \log d + \log 1/\delta}{n}}$$

We would like to stress that Theorem 2.3 can be applied to *any* node in the tree, and not only to the root. This is due to the nature of uniform convergence results, that derives worst case bound on the error, and the bound holds for any distribution. Therefore, no matter how we chose the path to the node (even adversarially), the bound would still hold. This is due to the observation, that no matter how we restrict the distribution (by the path to the node, in this case), when we consider only examples that have this property, they

algorithm	*database*					
	`australian`	`diabetes`	`german`	`heart`	`adult`	`Ionosphere`
No-Pruning	143.2 ± 12.3	56.2 ± 9.0	370.4 ± 15.0	61.0 ± 7.2	8742.6 ± 207.9	17.8 ± 4.2
`c4.5 -c 1`	5.2 ± 3.5	10.2 ± 6.2	2.6 ± 2.6	10.0 ± 4.0	41.6 ± 5.8	12.6 ± 1.9
`c4.5 -c 10`	13.6 ± 8.5	42.2 ± 6.1	47.0 ± 10.4	27.6 ± 5.9	180.8 ± 10.2	17.0 ± 3.2
`c4.5 -c 25`	39.8 ± 17.0	49.8 ± 9.4	110.4 ± 18.7	43.0 ± 7.2	675.0 ± 133.2	17.8 ± 4.2
`c = 0.5`	3.0 ± 0.0	3.4 ± 0.6	1.0 ± 0.0	3.8 ± 0.3	22.0 ± 3.2	3.0 ± 0.0
`c = 0.2`	6.4 ± 5.4	31.0 ± 8.0	47.0 ± 24.8	14.0 ± 6.0	50.4 ± 14.9	6.2 ± 1.0
`c = 0.15`	48.8 ± 4.6	47.4 ± 11.5	149.0 ± 14.0	35.0 ± 7.6	214.4 ± 55.9	10.2 ± 2.6
`c = 0.1`	69.8 ± 17.0	51.8 ± 11.0	240.2 ± 8.7	46.8 ± 8.2	768.6 ± 140.7	14.6 ± 2.9
`c = 0.05`	98.4 ± 16.3	55.4 ± 8.3	344.2 ± 21.0	58.2 ± 5.8	3368.6 ± 95.5	17.8 ± 4.2

Figure 1: The average tree size of each of the criterias.

are drawn independently and identically from the restricted distribution.

In this section we used very basic techniques to bound the deviations. More involved techniques may be used, and they could be based on the Vapnik Chervonenkis (VC) dimension [VC71]. For decision trees we where not able to determine the exact VC dimension as a function of their size, but we can show that it is between $\Omega(k)$ and $O(k \log d)$, for a binary decision tree over d inputs with at most k nodes (proof omitted). Generally, the smaller the VC dimension is the smaller the maximum deviation would be.

3 Our Criteria

Our criteria builds directly on the theory developed in Section 2. We would like, given an observed error on the training data, to derive a "pessimistic" estimate of the true error. The "pessimistic" indicates that we would like our estimate to be an upper bound of the true error. As we seen in the previous section, the theory suggests that the deviation can be bounded by $O(\sqrt{\frac{k \log d}{n}})$, where k is the size of the tree, d the number of attributes and n is the number of examples that the tree classifies.

Our pessimistic estimate would simply add $O(\sqrt{\frac{k \log d}{n}})$ to the observed error, and this would be our pessimistic estimate of the error. This would guarantee that with high probability our estimate is an upper bound of the true error. One issue that needs to be resolve is the constant that hides in the big O notation. Rather than trying to derive the exact theoretical constant, we kept it as a free parameter of our program, denoted by `c`. In our experiments we used different values for `c`. By controlling `c` one can control the size of the resulting tree, the larger `c` the more likely we are to prune.

To be more specific, assume we chose some value for `c`. Assume we are given a sub-tree of size k, that classifies n examples (each with d attributes), e of which it classifies incorrectly. If we prune the sub-tree, and replace it by a leaf, the leaf will make ℓ errors out of the n examples. Our test whether to prune the sub-tree is the following:

$$\frac{\ell}{n} \leq \frac{e}{n} + \mathtt{c}\sqrt{\frac{k \log d + \log 20}{n}}.$$

(In C4.5 the pruning process also includes comparing a sub-tree to the sub-trees of its children. We perform a similar test for that case.) Note that the modification in the error is based only on the tree size and the sample size and not on the number of errors, in contrast to the current method of C4.5 that uses e and n but not k.

4 Empirical Results

We have implemented our new criteria for pessimistic pruning in C4.5 and compared it with the existing method of C4.5. (We used C4.5 version 8.) First we did the comparison on an artificial data set, this experiment demonstrates the benefit of incorporating the tree size into the pruning criteria.

We chose the following artificial data. Each examples has 100 binary attributes. The classification is done as follows. With probability 0.9 we chose a random classification and with probability 0.1 we chose the value of the first attribute. The distribution over the inputs is uniform (i.e. each attribute is independent, and has equal probability of being zero or one). Clearly, the best classifier is simply to use the first attribute, and it has error 0.45.

We chose 10000 random examples, and ran C4.5 with

algorithm	*database*					
	`australian`	`diabetes`	`german`	`heart`	`adult`	`Ionosphere`
No-Pruning	$20.3 \pm 7.2\star$	27.0 ± 5.0	31.8 ± 3.0	22.6 ± 4.1	$15.2 \pm 0.2\star$	14.0 ± 7.2
`c4.5 -c 1`	$16.4 \pm 4.9\dagger$	24.6 ± 5.6	30.6 ± 1.7	22.6 ± 4.1	$14.4 \pm 0.2 \star \dagger$	12.8 ± 7.4
`c4.5 -c 10`	$15.9 \pm 5.5\dagger$	27.5 ± 4.0	29.2 ± 2.4	23.7 ± 4.1	$13.9 \pm 0.2\dagger$	14.2 ± 7.4
`c4.5 -c 25`	$17.1 \pm 5.7\dagger$	26.4 ± 5.4	30.1 ± 2.1	21.1 ± 3.9	$13.9 \pm 0.2\dagger$	14.0 ± 7.2
`c=0.5`	$15.8 \pm 5.6\dagger$	27.5 ± 3.0	30.0 ± 2.4	$26.7 \pm 4.9 \star \dagger$	$15.1 \pm 0.1\star$	$20.5 \pm 9.2 \star \dagger$
`c=0.2`	$15.9 \pm 5.5\dagger$	25.7 ± 4.4	$28.5 \pm 1.2\dagger$	23.0 ± 5.8	$14.5 \pm 0.2 \star \dagger$	14.8 ± 7.9
`c=0.15`	$16.7 \pm 6.1\dagger$	25.8 ± 5.4	29.4 ± 3.1	21.9 ± 3.4	$14.0 \pm 0.3\dagger$	14.2 ± 7.4
`c=0.1`	18.3 ± 6.0	26.4 ± 5.4	$28.7 \pm 2.8\dagger$	21.9 ± 4.0	$13.9 \pm 0.2\dagger$	12.5 ± 7.2
`c=0.05`	19.7 ± 6.3	27.0 ± 5.0	31.2 ± 2.6	23.0 ± 3.9	$14.5 \pm 0.2 \star \dagger$	14.0 ± 7.2

Figure 2: The average percentage error rate of each of the criterias. We marked by $\star$ significant differences from the standard C4.5 (`c4.5 -c 25`) and by $\dagger$ significant diffrences from the unprunned tree. (The significance level is 0.9 and it uses a difference of proportions test [Die96])

the parameter `-m 1` and `-c 1`. The unpruned tree had size 4567 (and zero error on the training set). We ran C4.5 with the default confidence parameter (`-c 25`) which generated a pruned tree of size 3685 (and only 244 mislabels on the training set). Even when we chose a very low confidence level (`-c 1`) it generated a pruned tree of size 551 (with 3201 errors on the training set). In contrast, when we used our criteria, with `c=0.7`, as the theory recommends, we recovered the *best* tree (of size 3!). (This best pruned tree is also achieved using `c=0.5`.)

We continued and checked the hypotheses generated using 40000 examples as a test set, in order to test the generalization error. For our criteria we know the exact error, forty five percent. We tested the hypotheses generated by C4.5, and they had error 49.5% (for `-c 25`) and 47.1% (for `-c 1`). As expected, the results show that the superfluous nodes only increase the generalization error. Using test of proportions [Die96], a 95% confidence interval around the best error (45%) gives the interval $[44.3, 45.7]$, which implies that the deviations, as expected, are very significant.

We did not limit ourself to artificial data, but considered also a variety of real databases. In order to perform the comparison we chose six databases, four taken from the project StatLog [Sta] and two databases from UCI Repository [MA94]. Since we are interested in functions with boolean output, the databases all have to predict a boolean output. The databases are:

1. `australian` – Australian Credit database. The aim is to decide whether to approve a credit card application. There are 690 examples. Each example has fourteen attributes, six continuous and eight categorical.
2. `diabetes` – Pima Indians Diabetes database. The aim is to predict whether a patient shows signs of diabetes. There are 768 examples. Each example has eight continuous attributes.
3. `german` – German Credit database. There are 1000 examples. Each example has twenty attributes, seven numerical and thirteen categorical.
4. `heart` – Heart Disease database. There are 270 examples. Each example has thirteen continuous attributes.
5. `adult` – US Census Bureau database. The aim is to predict whether the salary of a person is greater or less than 50,000$. There are 48842 examples. Each example contains fourteen attributes (six continuous and eight nominal).
6. `Ionosphere` – Radar data about the free electrons in the ionosphere. The aim is to predict if the structure is "good" or "bad". There are 351 examples. Each example has thirty four continuous attributes.

In the experiment we performed a 5-fold cross validation. (Namely we split the input to 5 equal parts, trained on four parts and tested on the fifth. We did this five times, one for each combination.)

In Figure 1 we have the mean tree size that the various criteria have generated. We considered the `unpruned` tree, the different confidence parameters to C4.5 (1, 10 and 25) and four settings of our criteria (denoted by `c= #`, where `#` is the constant we used for `c`).

It is very clear that in our criteria, as we increase the parameter `c` we are getting smaller and smaller trees, and by very significant factors. Initially, when `c=0.05`, there is very little pruning. As we get the parameter to `c=0.5` the trees that are generated are extremely small. In Figure 3 we plot the average tree size as a function of `c` for the database `adult`.

The error rate of the different criteria appears in Figure 2. In order to test the significance of the results we used the test of proportions [Die96] with confidence 90%. We compared the results to the unpruned tree and the C4.5 with the default setting (i.e., `c4.5 -c 25`. The results show that the setting of our parameter to `c=0.5` is too pessimistic, and it over-prunes. The other inferior criteria is the unpruned tree, here clearly we see that pruning helps. All the others seem competitive with each other.

In Figure 4 we plotted the error as a function of the parameter `c` used in our criteria. As expected, the behavior is as follows. Initially, the unpruned tree, has a large error, then, as more pruning is performed, the error drops to some minimal value (around `c=0.1`, in our case). From this minimum value, as we perform more pruning, the error starts to increase as we prune more and more.

Acknowledgements

I would like to thank Willian W. Cohen and Thomas G. Dietterich for pointing me to the relevant literature.

This research was supported in part by a grant from the Israel Science Foundation.

References

[AHM95] Peter Auer, Robert C. Holte, and Wolfgang Maass. Theory and applications of agnostic PAC-leaning with small decision trees. In *The 12th Internetional Conference on Machine Learning*, pages 21–30. Morgan Kaufmann, 1995.

[BB94] Marco Bohanec and Ivan Bratko. Trading accuracy for simplicity in decision trees. *Machine Learning*, 15:223–250, 1994.

[BFOS84] Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. *Classification and Regression Trees.* Wadsworth International Group, 1984.

[Die96] Thomas G. Dietterich. Statistical tests for comparing supervised classification learning algorithms. manuscript, 1996.

[Koh96] Ron Kohavi. Scaling up the accuracy of naive-bayes classifiers: a decision-tree hybrid. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, page to appear, 1996.

[MA94] P. M. Murphy and D. W. Aha. Uci repository of machine learning databases. `http://www.ics.uci.edu/ mlearn /MLRepository.html`, 1994.

[Min87] John Mingers. Expert systems – rule induction with statistical data. *Jounal of the Operational Research Society*, 38:39 – 47, 1987.

[Min89] John Mingers. An empirical comparison of pruning methods for decision tree induction. *Machine Learning*, 4:227 – 243, 1989.

[Nib87] T. Niblett. Constructing decision trees in noisy domains. In I. Bratko and N. Lavrac, editors, *Progress in Machine Learning–Proceedings of EWSL 87: 2nd European Working Session on Learning*, pages 67–78, Bled, Yugoslavia, May 1987.

[Qui86] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

[Qui87] J. R. Quinlan. Simplifying decision trees. *International Journal of Man-Machine Studies*, 1987.

[Qui93] J. R. Quinlan. *C4.5: Programs for machine learning.* Morgan Kaufmann, 1993.

[Sta] StatLog. Comparative testing and evaluation of statistical and logical learning algorithms for large-scale applications in classification, prediction and control. `ftp:ftp.ncc.up.pt/pub/statlog /datasets`, (See also: Machine Learning, Neural and Statistical Classification, ed. Michie, Spiegelhalter and Taylor).

[VC71] V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its applications*, XVI(2):264–280, 1971.

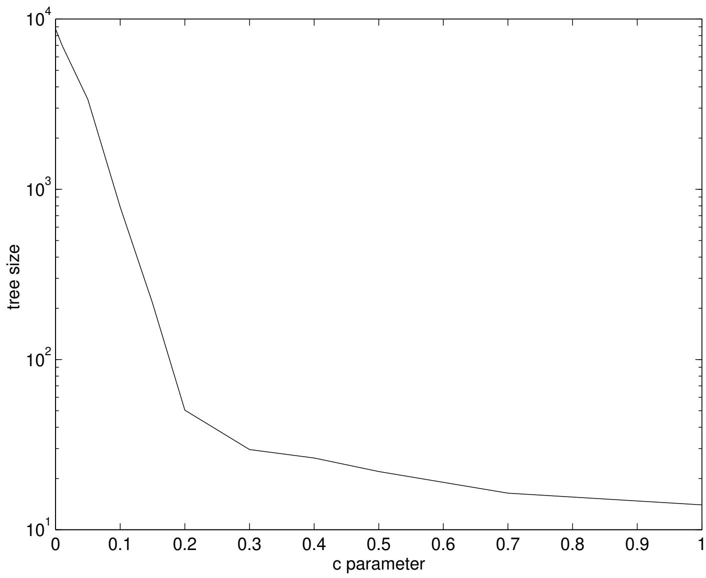

Figure 3: The average tree size as a function of the parameter `c` for the database `adult`.

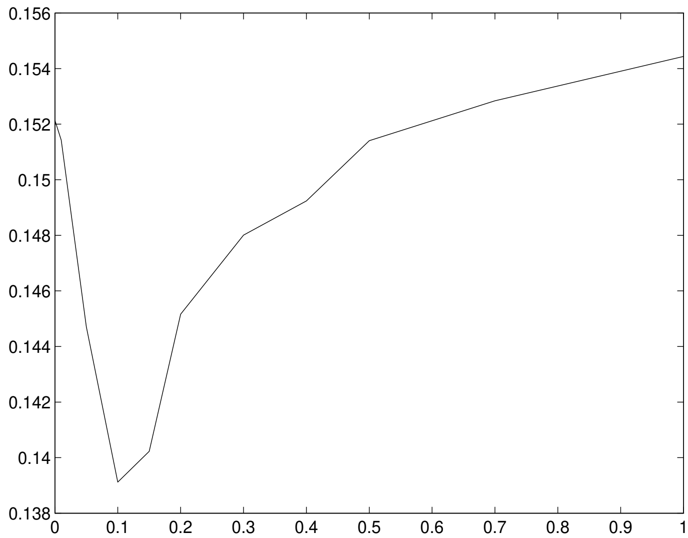

Figure 4: The average error as a function of the parameter `c` for the database `adult`.