Technical Note STAN-CS-TN-94-12

Cross-Validated C4.5: Using Error Estimation for Automatic Parameter Selection

George H. John
Computer Science Department
Stanford University
Stanford, CA 94305
gjohn@cs.Stanford.EDU

Oct 5, 1994

Abstract

Machine learning algorithms for supervised learning are in wide use. An important issue in the use of these algorithms is how to set the parameters of the algorithm. While the default parameter values may be appropriate for a wide variety of tasks, they are not necessarily optimal for a given task. In this paper, we investigate the use of cross-validation to select parameters for the C4.5 decision tree learning algorithm. Experimental results on five datasets show that when cross-validation is applied to selecting an important parameter for C4.5, the accuracy of the induced trees on independent test sets is generally higher than the accuracy when using the default parameter value.

1 Introduction

Cross-validation (Weiss & Kulikowski 1991, Bailey & Elkan 1993) is a powerful tool for error estimation which has unfortunately been under-utilized in the machine learning community. We investigate its use in the C4.5 decision tree algorithm (Quinlan 1993) as the evaluation component of an exhaustive search over a space of parameter values.

In supervised classification learning, one is given a *training set* of labeled instances. Each training instance is described by a vector of attribute (feature, measurement variable) values $\vec{x}$ and a categorical class label y (output, dependent variable). The job of a supervised classification learning algorithm is to induce some function $\widehat{f}$ which approximates the mapping $X \mapsto Y$.

A common choice for the representation of $\widehat{f}$ is a decision tree (Hunt, Marin & Stone 1966, Morgan & Messenger 1973, Breiman, Friedman, Olshen & Stone 1984). The process of building a decision tree is a recursive partitioning of a training set. CART (Breiman et al. 1984), C4.5 (Quinlan 1993) and other decision tree algorithms use a *stopping* parameter, which sets some lower bound on the number of elements that must be in a set in order for that set to be partitioned further. In C4.5 this is called the `MINOBJS` or m parameter. When a node contains fewer than m instances, it is not split further but rather made into a leaf labeled with the majority class.

Quinlan (1993) suggests that the C4.5 user should manually experiment with different values of the `MINOBJS` parameter, and the accompanying software includes a utility program to partially automate this task, using 10-fold cross-validation. An obvious extension is to fully automate this search and evaluation inside the learning algorithm itself, and this is what we describe in this paper.

To find the best setting of m we waged an all-out war against our datasets, trying twenty different settings of m, using five trials of 20-fold cross-validation repeated five times to estimate the value of each m setting, running C4.5 a total of ten thousand times on each dataset. This is the "speak softly, carry a small stick, but hit your opponent with it 10,000 times" approach. The remaining sections describe the use of cross-validation to select a parameter for a learning algorithm and our experimental results.

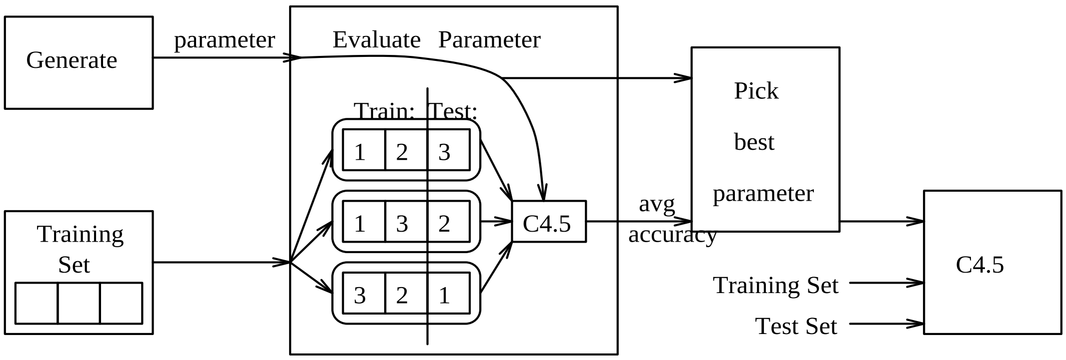

Figure 1: Cross-validation methodology (3-fold CV)

2 Searching for the Best m Parameter

Figure 1 describes the method for using cross-validation as the objective function in an exhaustive search for the best value of the m parameter. We tried all values from 1 to 10, 12, 15, 20, 25 and every tenth value from 30 to 100. To evaluate $m = 1$, say, we first break the training set into k blocks (three in the figure, although we use 20 in the experiments). We then repeatedly train `C4.5 -m1` using two out of the three blocks as training data and using the remaining block as a test set to estimate the accuracy of the induced tree. This gives three accuracy estimates, which are averaged to give the 3-fold cross-validation estimate of the accuracy when using $m = 1$. This is repeated for all other potential values of m. At the end we pick the value $\widehat{m}^*$ with highest estimated accuracy and use this value to train C4.5-$\widehat{m}^*$ on the entire training set and use an unseen test set to measure the resulting accuracy.

A problem with using cross-validation as an error estimate is that it has high variance. Running the above procedure several times will give markedly different estimates for the value of a given possible m parameter. One remedy used in our experiments is stratified CV, where an effort is made to ensure that the distribution of classes in each block is roughly the same. This only helps a little, so we overcame variance by running the entire CV algorithm five times and estimating the mean of the resulting set of estimates. We use a 20% *trimmed mean*, removing the top and bottom 20% of the values and then taking the mean of the rest. A trimmed mean is a more robust estimator (Hubel 1977) of the mean of a set of data if the data is suspected to contain outliers. As a general rule of thumb, we found that this procedure generally allows us to get a good (low variance) estimate of error with only half as many runs of CV as were needed when using the usual mean estimate. Additionally, since the m parameter depends heavily on the training set size, we used 20-fold cross-validation (where the training set is broken into 20 blocks) so that each training set used in cross-validation would contain 95% of the instances in the original training set. The best estimated value of m is multiplied by 20/19 as a final correction for training set size.

3 Experiments

We show results for four datasets taken from the UC Irvine repository (Murphy & Aha 1994) and one artificial dataset of our own construction. The Irvine datasets are the Wisconsin breast cancer database, the Pima Indians diabetes database, the credit database, and the artificial monks2 data. Our "Financial" dataset is so named because it is an attempt at constructing a dataset with characteristics one would find in many real financial domains. We arbitrarily divided each database into a training and test set. For a better comparison we should have used cross-validation for this final evaluation step as well, but we chose to use a single training and test set to avoid confusing the role of cross-validation in this work. We then ran C4.5, CVC4.5, and C4.5* on each dataset, and results are given in Table 1. C4.5 results are from running the C4.5 algorithm with default parameters and estimating the accuracy of the pruned tree on the unseen test set. CVC4.5 results are from using cross-validation to pick the optimal m parameter value and then using the test set to estimate the accuracy of the pruned trees. Results shown are from five runs. Variance in size was

Table 1: Experimental Results.

			C4.5		CVC4.5		small CVC4.5		C4.5*	
Dataset	Train	Test	Acc	Size	Acc	Size	Acc	Size	Acc	Size
Breast	466	233	94.4	19	94.34±1.10	13	94.4	5	95.7	31
Pima	512	256	68.4	99	72.74±1.23	8	72.3	3	74.6	9
Credit	490	200	81.0	39	81.29±0.04	41	81.5	28	85.0	9
Monk2	169	432	65.0	31	65.20±1.04	22	67.1	1	67.1	1
Financial	250	150	78.7	45	81.34±1.32	35	78.7	19	82.7	55

large but is not shown. Small-CVC4.5 results are from picking the smallest induced tree from among the 5 CVC4.5 runs. C4.5* results are optimistic upper bounds – we "cheated" and used the test set itself instead of cross-validation to pick the best value of the m parameter, then used the same test set to evaluate the induced tree.

The CVC4.5 results are almost uniformly better than the C4.5 results, but for a few bad runs which brought up the averages in the CVC4.5 column. Small-CVC4.5 is uniformly better than C4.5 on all domains tested. Even when Small-CVC4.5 and C4.5 have the same accuracy, Small-CVC4.5 is often smaller.

4 Conclusion

We have described a simple method of improving the C4.5 algorithm by using cross-validation to select the m parameter of the tree construction algorithm. This yields better accuracy and smaller trees than C4.5 run with the default parameter settings. We are not claiming to be at the forefront of machine learning with our new CVC4.5 algorithm; rather we intended to demonstrate the power of cross-validation used as a parameter selection tool. We hope that our results will inspire machine learning researchers to look more closely at cross-validation and related statistical error estimation techniques such as the bootstrap (Efron & Tibshirani 1986) as tools for parameter selection.

Acknowledgments

This material is based upon work supported under a National Science Foundation Graduate Research Fellowship. We would like to thank J. R. Quinlan for publishing the source code to C4.5, which allowed us to implement our ideas quite easily. Thanks to Ronny Kohavi for pointing out the importance of the `MINOBJS` parameter in C4.5.

References

Bailey, T. L. & Elkan, C. (1993), Estimating the accuracy of learned concepts, *in* R. Bajcsy, ed., "Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence", Morgan Kaufmann, pp. 895–900.

Breiman, L., Friedman, J., Olshen, R. & Stone, C. (1984), *Classification and Regression Trees*, Chapman & Hall, New York.

Efron, B. & Tibshirani, R. (1986), "Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy", *Statistical Science* **1**(1), 54–77.

Hubel, P. J. (1977), *Robust Statistical Procedures*, Society for Industrial and Applied Mathematics, Pittsburgh, PA.

Hunt, E. B., Marin, J. & Stone, P. J. (1966), *Experiments in Induction*, Academic Press, New York.

Morgan, J. N. & Messenger, R. C. (1973), *THAID: a sequential analysis program for the analysis of nominal scale dependent variables*, University of Michigan.

Murphy, P. M. & Aha, D. W. (1994), "UCI repository of machine learning databases", Available by anonymous ftp to `ics.uci.edu` in the `pub/machine-learning-databases` directory.

Quinlan, J. R. (1993), *C4.5: Programs for Machine Learning*, Morgan Kaufmann.

Weiss, S. M. & Kulikowski, C. A. (1991), *Computer Systems that Learn*, Morgan Kaufmann, San Mateo, CA.