Journal of Machine Learning Research 1 (2000) 1-48 Submitted 2/23; Published 2/23

Decision Trees for Regression and Classification

Brendan Foley BFOLEY3@JH,EDU
Department of Biotechnology
Johns Hopkins University
Baltimore, MD 21202, USA

Editor: Brendan Foley

Abstract

This paper describes the creation and use of Decision Trees for classification and regression. Throughout the paper, several publicly available datasets are used, as noted in the Acknowledgments and References. The general methodology is as follows: we begin creating our tree, selecting the root node to decide on the "most informative" attribute first, as determined by a maximal gain ratio or minimal squared error, for categorical and numerical attributes respectively. After splitting the data, we begin recursively defining nodes, continuing until we have purity in our leaves. After constructing an unpruned tree, we assess the accuracy of classification and the mean square error (MSE) of regression on the tree with the testing set. We then prune the trees using reduced-error pruning and assess their function again.

Keywords: Assignment 3, CART, Decision Trees, Machine Learning

1. Introduction

In this assignment, we have developed a decision tree model for classification and regression tasks. In order to determine the splitting of the root node into its children, we must first decide whether the task is classification or regression. In the case of classification, we utilize a maximal gain-ratio to determine the optimal split based on the data contained in that node. When the task is regression, we select the attribute to split on based on the minimal squared error. Lastly, we allow for the use of pruning, if desired. We implement reduced-error pruning to eliminate excess leaves, given that the resulting smaller tree has entropy within a given threshold, θ.
We predict that we will be able to efficiently construct decision trees for the given datasets. However, datasets that are rich in numeric attributes will take longer to come to a final decision tree, as there is the extra overhead of finding an optimal partitioning value.

2. Experimental Approach

To begin creating our decision trees, we must first find the attribute that either minimizes the squared error, in the case or regression, or maximizes the gain ratio, in the case of classification. The equations for gain ratio are shown in equations 1-5, the equations for squared error are shown in equations 6 and 7.

$$gratio(f_i) = \frac{gain(f_i)}{IV(f_i)} \tag{1}$$

$$gain(f_i) = H(D_\pi) - E_\pi(f_i) \tag{2}$$

$$H(D_\pi) = -\sum_{l=1}^{k} \frac{c_{\pi,l}}{|D_\pi|} \lg \frac{c_{\pi,l}}{|D_\pi|} \tag{3}$$

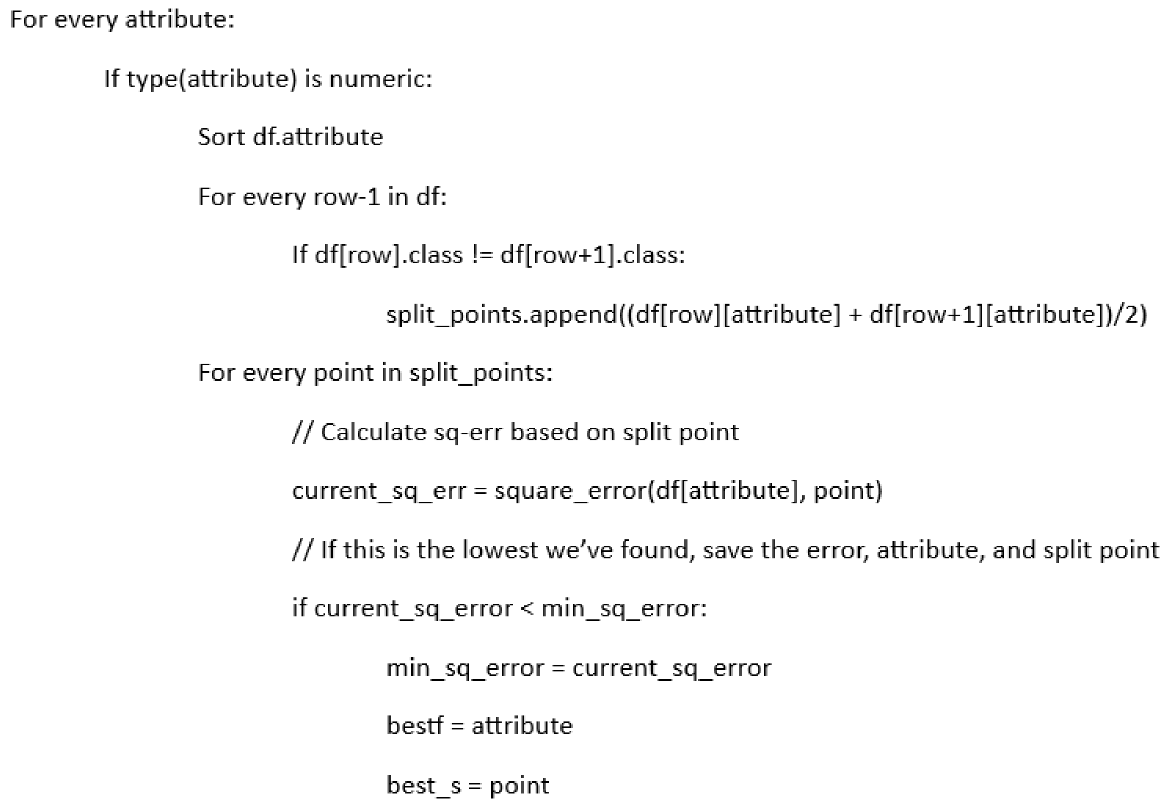
$$E_\pi(f_i) = \sum_{j=1}^{m_i} \frac{|D_\pi^j|}{|D_\pi|} H(D_\pi^j) \tag{4}$$

$$IV(f_i) = \sum_{j=1}^{m_i} \frac{|D_\pi^j|}{|D_\pi|} \lg \frac{|D_\pi^j|}{|D_\pi|} \tag{5}$$

$$Err'_\pi = \frac{1}{D_\pi} \sum_j \sum_l (r^l - r^l_{\pi j})^2 b_{\pi j}(x^l) \tag{6}$$

$$b_{\pi j}(x^l) = \begin{cases} 1 & \texttt{if } x^t \in D_\pi \\ 0 & \texttt{otherwise} \end{cases} \tag{7}$$

To find the optimal attribute to split on, we greedily iterate through each column, calculating the respective statistic. In the case of categorical attributes, we can perform a ”natural” split on the values provided by the attribute. However, on numerical attributes, we must identify various split points and choose the most optimal. The following pseudocode illustrates our process of identifying the optimal split.



```
For every attribute:
    If type(attribute) is numeric:
        Sort df.attribute
        For every row-1 in df:
            If df[row].class != df[row+1].class:
                split_points.append((df[row][attribute] + df[row+1][attribute])/2)
        For every point in split_points:
            // Calculate sq-err based on split point
            current_sq_err = square_error(df[attribute], point)
            // If this is the lowest we've found, save the error, attribute, and split point
            if current_sq_error < min_sq_error:
                min_sq_error = current_sq_error
                bestf = attribute
                best_s = point
```

Figure 1: Pseudocode for finding the optimal split point of a numeric attribute. This is in the case of regression, as we are calculating squared-error, however, the code remains the same for classification, with the exception being our splitting criterion changes to maximal gain ratio.

Once we have the optimal attribute and split point (when appropriate), we then branch our dataset into those meeting the criteria. In the case of categorical attributes, we have n child nodes, where n is the number of values that attribute can take on. When the attribute is numeric, we have 2 child nodes, one where the rows' attribute is $<$ the split value, and one where it is $\geq$ the split value. We store the attribute to split on in the current node, and create the branches recursively. Our node will also keep track of its children.
Proceeding in this fashion, we can grow out our decision tree until we have pure child nodes, or, in the case of pruning, until we see that the threshold value has been met, and our leaves are "pure enough".

3. Results

The following tables report the results of my 5-fold validation runs with, and without pruning. The datasets that are involved in classification have their results reported as percent accuracy, while the datasets using regression have their results reported as mean square error (MSE).

Run #	Accuracy
1	.96
2	.94
3	.95
4	.96
5	.96

Table 1: The results of 5x2 cross validation of the Breast Cancer dataset, before pruning

Run #	Accuracy
1	.66
2	.68
3	.68
4	.68
5	.68

Table 2: The results of 5x2 cross validation of the Breast Cancer dataset, after pruning

Run #	Accuracy
1	.93
2	.93
3	.91
4	.93
5	.93

Table 3: The results of 5x2 cross validation of the Cars dataset, after pruning

Run #	Accuracy
1	.70
2	.66
3	.66
4	.66
5	.66

Table 4: The results of 5x2 cross validation of the Cars dataset, after pruning

Run #	Accuracy
1	.93
2	.91
3	.94
4	.93
5	.91

Table 5: The results of 5x2 cross validation of the House Votes dataset, after pruning

Run #	Accuracy
1	.61
2	.62
3	.95
4	.94
5	.92

Table 6: The results of 5x2 cross validation of the House Votes dataset, after pruning

Run	MSE
1	6.58
2	4.90
3	4.92
4	4.66
5	4.84

Table 7: The results of 5x2 cross validation of the regression on the Abalone dataset, before pruning

Run	MSE
1	10.2
2	8.06
3	8.06
4	8.06
5	8.06

Table 8: The results of 5x2 cross validation of the regression on the Abalone dataset, after pruning

Run #	MSE
1	957.5
2	1244
3	1285
4	1148
5	1304

Table 9: The results of 5x2 cross validation of the regression on the Computers dataset, before pruning

Run #	MSE
1	1003
2	1057
3	1294
4	1061
5	1248

Table 10: The results of 5x2 cross validation of the regression on the Computers dataset, after pruning

Run #	MSE
1	19.30
2	21.90
3	19.48
4	22.49
5	22.45

Table 11: The results of 5x2 cross validation of the regression on the Forest Fires dataset, before pruning

Run #	MSE
1	4.87
2	5.09
3	5.09
4	4.53
5	5.09

Table 12: The results of 5x2 cross validation of the regression on the Forest Fires dataset, after pruning

4. Discussion

The differences between a pruned and unpruned tree are significant. There is a roughly 30% drop in accuracy in classification tasks for the pruned tree. However, regression seems to be more robust to the effects of pruning. Anecdotally the pruned tree is faster, however, I do not have any runtimes to prove the claim at the moment.
There is an interesting case in table 6. There appear to be two trees that, after pruning, have accuracies equal to the unpruned trees. Our guess as to why that is relies on the structure of the dataset. The House Votes dataset has more attributes than the others, and all of the attributes have one of three values : ”y, ?, n”. These low order categorical attributes may make it possible that pruning a tree will result in just as high accuracy as an unpruned one.
This is again observed in tables 8 vs 9. The pruned tree has lower MSE than the unpruned tree.
While Decision Trees are relatively straight forward, we still had some issues to overcome in the of our solution. Throughout the development, regression and numerical attributes were a thorn in our side. Finding the optimal split on numerical attributes took a long time, and in a dataset made up mostly of numerical attributes, such as the abalone dataset, it quickly swells in runtime.
On regression tasks, having mixed types of variables (categorical and numerical) created issues, as we attempt to find the best categorical variable to maximize the gain ratio on, while finding the best numerical variable to minimize squared error on. We introduced bias by allowing for categorical variables to take precedence over numerical variables. Another form of bias introduced was how to split the numerical attributes in regression. We originally employed a median split, but moved to a mean split to help limit some side-effects that we attributed to splitting based on the median. In the end, it appeared to be unrelated, so revisiting the median based split in the future may improve our MSE.
The last hurdle we faced was when a regression tree would get stuck in infinite recursion, thinking that a attribute with identical values (as it had already been split on multiple times) was being split on again, as it appeared that the minimum squared error was that attribute. In the end, this appeared to stem from an issue of squared-error calculation, where I was not using the proper subsets, so the same error value was being returned.

5. Conclusion

By creating decision trees, we are able to have relatively accurate (about 95%) classification, and low mean squared error (MSE) for regression.. The pruning within this experiment could be dialed back, as it seems that the pruning causes too much loss of accuracy for classification. However, it seems to maintain, or improve on, the MSE for regression tasks.

Acknowledgements

All data used for this assignment was obtained from http://archive.ics.uci.edu/ml, from the University of California, Irvine, School of Information and Computer Science. The breast

cancer dataset was obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg

References

1. Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

2. O. L. Mangasarian and W. H. Wolberg: "*Cancer diagnosis via linear programming*", SIAM News, Volume 23, Number 5, September 1990, pp 1 & 18.

3. P. Cortez and A. Morais. A Data Mining Approach to Predict Forest Fires using Meteorological Data. In J. Neves, M. F. Santos and J. Machado Eds., New Trends in Artificial Intelligence, Proceedings of the 13th EPIA 2007 - Portuguese Conference on Artificial Intelligence, December, Guimarães, Portugal, pp. 512-523, 2007. APPIA, ISBN-13 978-989-95618-0-9.