A Significance Measure for Node Splits in C4.5

Nicholas Bidler
Department of Computer Science
California State University
Fullerton, CA 92831, USA
nbidler@csu.fullerton.edu

Shawn X. Wang
Department of Computer Science
California State University
Fullerton, CA 92831, USA
xwang@fullerton.edu

***Abstract*—Decision trees remain one of the most interpretable and computationally efficient classification methods. The ID3 (Iterative Dichotomiser 3) algorithm is a classic algorithm in machine learning used to construct decision trees from a dataset. C4.5 extends ID3 to handle numeric attributes using thresholding and information gain. The C4.5 algorithm recursively partitions data based on maximum information gain, producing trees that progress from the most informative attributes at the root to the least informative at the leaves. While inner nodes are chosen by optimizing entropy-based criteria, leaf nodes do not admit further entropy comparisons, leaving the importance of the *last* split ambiguous. We propose a simple, dataset-agnostic scoring scheme that quantifies the significance of a parent node's final subdivision into its leaves, enabling a direct measure of the utility of the final split. After training a full tree, we record predictions and ground truth at each leaf on a held-out test set, compute per-leaf scores as *accuracy* $\times$ *correct*, and then "merge" sibling leaves back into their parent by concatenating their predicted/actual lists; the same score is computed for the merged node. The final-split significance is measured as the difference between the sum of leaf scores and the merged-parent score; zero indicates no benefit from splitting. Across several public datasets, the measure highlights when small but accurate leaves or imbalanced families meaningfully improve performance, and when splits add little value. The measure is simple to compute, compatible with standard pruning workflows, and can act as a practical heuristic for late-split evaluation. Our experiments on multiple real-world datasets show that this approach can identify and prune non-beneficial splits, improving model interpretability without compromising accuracy. This method offers a statistically aware pruning criterion that complements existing entropy-based splitting strategies.**

***Index Terms*—Decision trees, C4.5, information gain, pruning, split significance, interpretability.**

I. Introduction

Decision trees provide a clear, fast, interpretable, transparent classification and remain competitive in many settings. C4.5, an influential successor to ID3, selects attributes by information gain and supports continuous features via learned thresholds. However, the contribution of the final split above the leaves is difficult to assess with entropy alone. This paper introduces a compact scoring measure to quantify the value of that last subdivision and evaluates its behavior across multiple datasets. Beyond practical utility, a principled view of late splits is relevant for both interpretability and pruning. From an interpretability angle, last-hop decisions often correspond to concrete rules practitioners cite and within the application context; knowing whether such rules matter prevents over-emphasizing inconsequential splits. From an optimization angle, pruning typically removes branches with limited payoff; an explicit late-split score can be used alongside cost-complexity or error-based criteria.

Our contribution is orthogonal: we target the *marginal utility of the final split* using a lightweight, post-hoc score suitable for integration into pruning. Here are hightlights of our key contributions.

- Introduce a **split significance score** for evaluating final splits in C4.5 decision trees.
- Provide a framework to compare leaf accuracy with merged-node accuracy for pruning.
- Demonstrate performance across diverse real-world datasets.

II. Related Work

Classic formulations of ID3 and C4.5 are well established [2], [3]. Numerous decision-tree variants and pruning criteria have been proposed. Comparative studies often find modest accuracy gaps among tree learners, with overfitting mitigated by early stopping or post-pruning .

Both ID3 and C4.5 are based on entropy. The entropy of information of the training data set D is calculated by the following equation (1):

$$Entropy(D) = -\sum_{i=1}^{N} \frac{|C_i|}{|D|} \log_2 \frac{|C_i|}{|D|}. \tag{1}$$

Then, equation (2) below is used to calculate the entropy of information associated with an attribute A:

$$Entropy(D, A) = \sum_{v \in V} \frac{|D_v|}{|D|} Entropy(D_v), \tag{2}$$

where v is a value of the attribute A, and V refers to the domain of A, $D_v = \{x_i \in D \mid A(x_i) = v\}$, and $|D_v|$ denotes the cardinality of D_v.

Finally, the IG of the attribute A in the data set D can be calculated as the following equation (3):

$$IG(D, A) = Entropy(D) - Entropy(D, A). \tag{3}$$

979-8-3315-5397-5/25/$31.00

Recently, Aaboub, *et. al.* [4] published an analysis of the prediction performance of decision tree based algorithms and ID3 is still a top performer for several datasets (Table I).

TABLE I
CLASSIFICATION ACCURACIES (%) ACHIEVED BY THE ID3, CART, PCC-TREE, AND DRDT APPROACHES ON THE ELEVEN DATA SETS.

Data sets	ID3	CART	PCC-Tree	DRDT
Bankruptcy	**98.34**	98.30	91.12	97.52
Breast Cancer	69.94	**69.97**	68.67	67.37
Glass	74.10	**74.51**	67.29	66.35
Hepatitis	**80.13**	78.47	77.50	77.74
Iris	93.49	**93.54**	93.49	88.17
New Thyroid	**91.68**	89.89	89.71	90.09
Somerville	62.03	**62.18**	58.68	60.43
Sonar	**73.13**	71.12	71.39	70.20
Statlog	**79.09**	78.08	74.70	76.93
TAE	46.65	45.85	49.19	**57.64**
Wine	90.28	88.39	**93.19**	82.29
Average	78.08	77.30	75.90	75.88

C 4.5 behaves similarly to ID3 in selecting which attribute to split on but extends ID3 by being able to handle numeric values. When it encounters an attribute that has a numeric value, the values are sorted in ascending order, and makes a note of the halfway point between each pair of values present as a possible "split value" or "split point." For example, a set with values [1, 1, 3] would only have a single possible split at 2, while [1, 1, 2] would have a single split value of 1.5. If each value of the attribute is unique, then for a list of length N, the maximum number of possible breakpoints would be N-1, though this is only a worst-case scenario. For each split point possible, the dataset is split into two subsets and the information gain is calculated. The highest information gain is used to select which value the dataset is subdivided on, and the decision tree node is the numeric attribute, with the two branches being "above split value" and "below split value." Otherwise, C4.5 is very similar to ID3. C4.5 was ranked number 1 most effective data mining algorithm [6].

Decision trees in general are very prone to over-fitting their "training data," so it is common practice to "prune" the tree. Typically, this improves the tree's ability to generalize. These fall into two types: early stopping or removing nodes. Early stopping includes things like simply limiting the depth of the tree, a minimum number of samples required to form a leaf or to split a node, and similar "hard-coded" limits. Removing nodes includes "cost-complexity pruning," which assigns scores to subtrees and removes the subtree with the worst score, removing branches that have the smallest effect on the tree's overall accuracy, and removing leaf nodes with fewer samples than an arbitrary threshold. In most cases, these methods have "hard coded" limits, which are human-readable and easy to implement or alter. However, they are generally insensitive to context. Pruning methods include depth limits, minimum samples per leaf, and post-hoc subtree replacement based on error estimates. Significance assessments for splits have also been explored via statistical tests and regularization.

Prior research on decision tree pruning includes cost-complexity pruning [18] , reduced-error pruning [19] , and statistical pruning [20] . Modern interpretability methods such as LIME [21] and SHAP [22] provide post-hoc explanations but do not guide tree construction or pruning. More recent work explores statistical thresholds and surrogate modeling, yet no existing approach explicitly measures the benefit of final-level splits as proposed here.

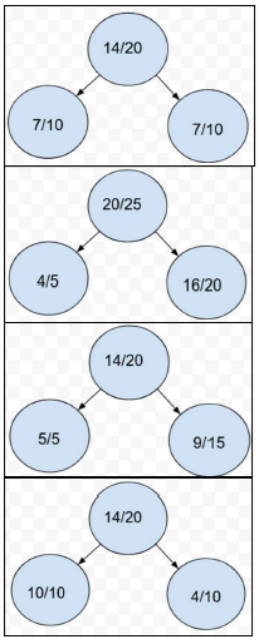


Fig. 1. Node split examples with number of correct predictions/number of total predictions

We complement these lines by offering a directly computable, model-internal indicator that leverages held-out performance without additional modeling assumptions.

III. METHODOLOGY

A. Scoring Per Leaf and Merged Parent

We utilized a score that is accuracy multiplied by the "correct" population. To provide proof of concept, say there exist two nodes and their merged parent node (Figure 1), each one having some number of correct predictions and some number of total predictions. If the two child nodes are perfectly evenly split, trying to differentiate them from their parent is difficult. Compare the weighted averages of a perfectly even split: (0.7 7) + (0.7 7) = 9.8 = 0.7 14 (Figure 1a). Note that the merger results in the same value as the sum of the split values, even if the distribution of the split is lopsided, if the accuracy is the same on both sides: (0.8 4) + (0.8 16) = 16 = 0.8 20 (Figure 1b). With a split that does have some differentiation between the leaves and parent, there is some difference: (1.0 5) + (0.6 9) = 10.4 $\neq$ 9.8 = 0.7 14 (Figure 1c). This seems promising, given one child has a small population but high accuracy. If we keep the lopsided accuracy with altered sizes and compare it to the third example, the pattern seems to hold: (1.0 10) + (0.4 4) = 11.6 $\neq$ 9.8 = 0.7 14 (Figure 1d). We define a

score to be $C = (Accuracy_A \times Correct_A) + (Accuracy_B \times Correct_B) - (Accuracy_M \times Correct_M)$, where A and B are the two child nodes and M is the merged parent node, gives a nice neat constant C that can be compared between clusters or "families" of nodes. On one hand, the two separate nodes were always going to be "more accurate" than an undifferentiated parent. On the other hand, this allows for a "universalizable" measurement across the same tree and trees made from other datasets. The issue of comparing the "significance of the split" has at least a preliminary measurement that can be used to measure the importance of the split between the child nodes and the parent node. It is important to remember that while most "score differences" are less than one, this figure is not a ratio (Figure 1). It is possible to re-use this measure between the two child nodes, but the usefulness of doing so is debatable: (1.0 10) = 10 > 1.6 = (0.4 4) would show that the left node is much more "impactful" than the right node, which would imply a simple relation of "large number = large impact." This may be misleading when comparing a different sample: (1.0 5) = 5 < 5.4 = (0.6 9) or a more extreme example like (0.8 50) = 40 = (0.4 100). These examples show that using this as a measure between two "sibling" nodes would give equal importance to accuracy and the "correct population" which is of debatable use as a measure of importance.

For each leaf ℓ, let acc(ℓ) be accuracy on its test instances and cor(ℓ) the number of correct predictions. The leaf score is

$$S(\ell) = \text{acc}(\ell) \times \text{cor}(\ell). \tag{4}$$

For a branch node p whose children are all leaves, we merge those leaves by concatenating their predicted/actual lists as if p had not been split. The merged score is $S(p)$. The *final-split significance* for that family is

$$\Delta(p) = \sum_{\ell \in \text{children}(p)} S(\ell) - S(p). \tag{5}$$

A value of $\Delta(p) = 0$ indicates no measurable benefit from the final subdivision, while larger values indicate that splitting produced more utility.

Given a fully constructed C4.5 decision tree, we identify branch nodes whose children are all leaves. For each such branch node b, we compute:

$$S_{\text{leaf}} = \sum_{i=1}^{k} n_{\text{correct},i} \times a_i \tag{6}$$

where k is the number of leaf children, $n_{\text{correct},i}$ is the number of correct predictions for leaf i, and a_i is the accuracy of leaf i.

We then merge the prediction records of all k leaves to form a hypothetical unsplit node and compute:

$$S_{\text{merged}} = n_{\text{correct, merged}} \times a_{\text{merged}} \tag{7}$$

The **split significance score** is then:

$$\sigma = S_{\text{leaf}} - S_{\text{merged}} \tag{8}$$

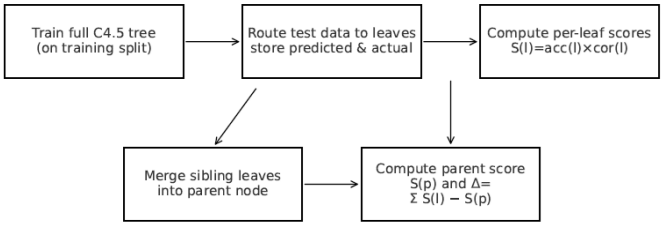


Fig. 2. Workflow for computing the final-split significance. Train a full tree, collect leaf-level predicted/actual on test data, compute leaf scores, merge siblings, compute parent score, and obtain Δ.

B. Rationale and Properties

The score treats accuracy and correct counts symmetrically, rewarding leaves that are both precise and sufficiently populated. Because it is computed on held-out data, it reflects generalization rather than training fit. The metric is additive across families, allowing comparison of different bottom-level branches within the same tree. While not a hypothesis test, it provides a monotone signal that aligns with practical pruning choices.

C. Algorithm

Algorithm 1 Final Split Significance Scoring

```
Build a full C4.5 decision tree on the training data.
for each branch node b whose children are all leaves do
    Compute S_leaf from leaf accuracies and correct counts.
    Merge child records to form a hypothetical parent node.
    Compute S_merged for this merged node.
    Calculate σ = S_leaf − S_merged.
    if σ <= 0 then
        Mark branch b for pruning.
    end if
end for
Return pruned decision tree.
```

D. Implementation Notes

We used a straightforward C4.5-style implementation with entropy/information-gain for attribute selection and threshold search for continuous attributes. Trees are fully expanded (no pre-pruning) before computing Δ. The procedure is compatible with standard pruning and can be iterated to progressively "raise the floor" by re-merging bottom layers. We first train a full C4.5-style tree on a training split and evaluate on a held-out test split. Each leaf stores the sequence of predicted labels (via majority label from training) and the corresponding ground-truth labels encountered during testing. Figure 2 summarizes the workflow.

IV. Datasets and Experimental Setup

We used commonly referenced datasets spanning medical, banking, and environmental domains. We perform random train/test splits and fully expand trees prior to evaluating Δ. All runs strictly use test data for scoring to avoid optimistic

TABLE II
DATASETS SUMMARY (AS USED IN OUR EXPERIMENTS)

Dataset	Attributes	Samples
Pima Diabetes	8	∼768
Sample Bank (tutorial)	10	600
Obesity (UCI)	16	>2000
Algerian Forest Fires	13	∼240
Blood Transfusion	4	∼748
Breast Cancer (Prognostic)	33	< 200

bias. The operational environment's hardware used was a workstation. The software used was Jupyter Version 4.1.2, viewed through the Firefox 125.0.3 web browser. Python version Python 3.9.18. Environment: Conda version 23.1.0

The first dataset used in testing was "diabetes.csv," from Kaggle [13]. This was important, as this data was purely numeric, and if the algorithm had any issues parsing and splitting the dataset it would be immediately apparent. This was followed by "bank-data.csv" from DePaul University [12]. This was selected as an example of a true-ish dataset – it was intended as a testing dataset and had categorical and continuous values, thus allowing for testing of the handling of non-numeric values. The first "true" dataset used in testing was "Estimation of Obesity Levels Based On Eating Habits and Physical Condition," from the UCI Machine Learning Repository [14] This dataset was large, and during testing, revealed an edge case where an entry from the test set simply never occurred in the training set. We chose three more datasets all from the UCI Machine Learning Repository, Algerian Forest Fires [16], Blood Transfusion Service Center [17], and Breast Cancer Wisconsin (Prognostic) [15]. Table II summarizes key characteristics of the datasets.

V. RESULTS

Overall trends. The proposed Δ measure effectively distinguishes useful final splits from negligible ones. Datasets with many samples and fewer attributes (e.g., Diabetes, Blood Transfusion, Obesity) more often exhibit families with nonzero Δ, sometimes ≥ 0.6, reflecting informative last splits. In contrast, settings with few samples but many attributes (e.g., Forest Fires, Breast Cancer) frequently produce few evaluable families and many with $\Delta \approx 0$.

Visualization of the Significance Score Figure 3 illustrates the experimental results for the diabetes dataset. Figure 4 illustrates the results for the bank dataset. Figure 5 illustrates the results for the Obesity dataset. They aggregate representative family-level Δ magnitudes across datasets: the Sample Bank dataset often shows small gains (around 0.25), while Diabetes and Obesity can show larger gains, indicating beneficial late splits. These patterns align with the intuition that richer sample sizes support more informative partitioning at the bottom of the tree.

VI. LIMITATIONS AND FUTURE WORK

The score trades precision for simplicity. It equally weights accuracy and correct counts and evaluates only families whose children are leaves, potentially ignoring higher-level opportunities. Future work includes integrating Δ as a pruning criterion with tunable thresholds, examining statistical alternatives (e.g., tests for set vs. subset comparisons), and exploring relationships between Δ trajectories under repeated merges and node-level entropy. Another direction is to study how Δ correlates with generalization under different train/test splits and to derive confidence intervals around Δ using bootstrap resampling.

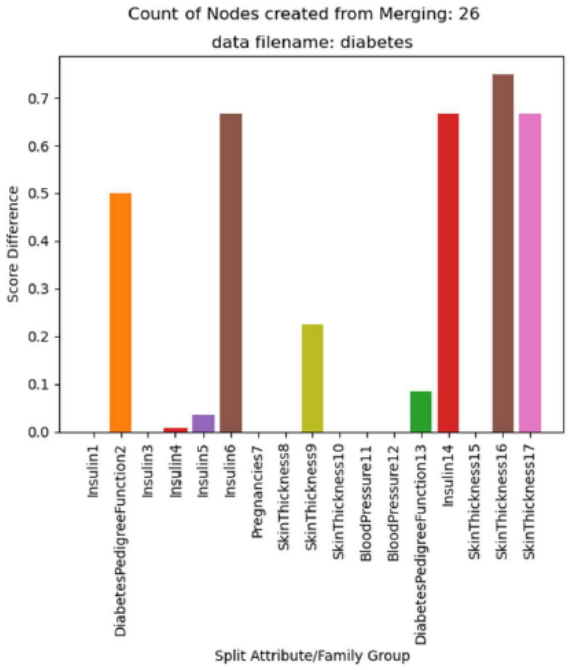
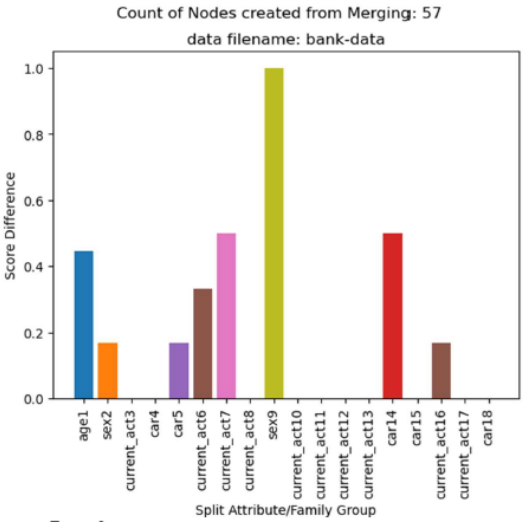


Fig. 3. Representative final-split significance Δ across datasets - diabetes



data.png

Fig. 4. Representative final-split significance Δ across datasets - bank data

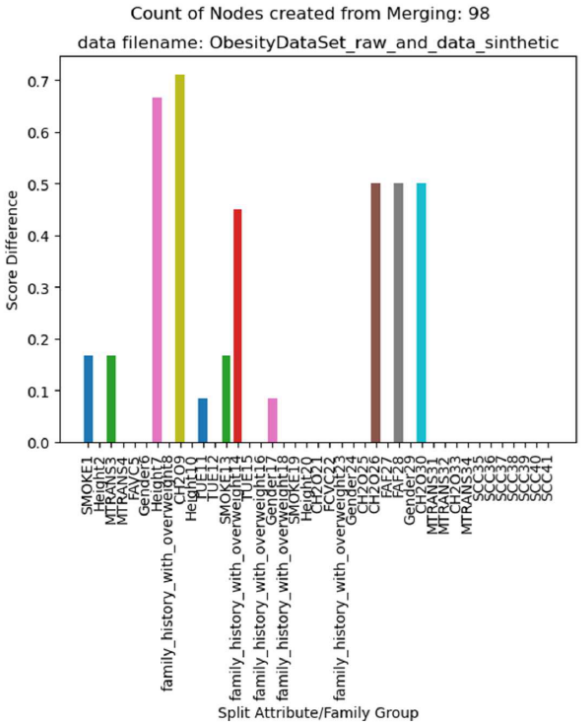


Fig. 5. Representative final-split significance Δ across datasets - obesity

VII. Conclusion

We presented a compact significance measure for the final split in C4.5-style trees. By contrasting summed leaf scores with a merged-parent score on held-out data, Δ identifies when the last subdivision adds practical value. The measure is easy to compute and can complement existing pruning strategies and interpretability analyses. In practice, it can serve as a quick diagnostic: if many bottom families yield $\Delta \approx 0$, a simpler pruned model may suffice without sacrificing accuracy.

Acknowledgment

This work was submitted to the California State University, Fullerton to fullfil the requirements of a graduate research project for a master degree in computer science.

References

[1] K. Lan, *et al.*, "A Survey of Data Mining and Deep Learning in Bioinformatics," *Journal of Medical Systems*, vol. 42, no. 8, 2018, pp. 139–20.

[2] J. Quinlan, "Learning Decision Tree Classifiers," *ACM Computing Surveys*, vol. 28, no. 1, 1996, pp. 71–72.

[3] J. R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[4] F. Aaboub, H. Chamlal, and T. Ouaderhman, "Analysis of the prediction performance of decision tree-based algorithms," in *Proc. Int. Conf. Decision Aid Sciences and Applications (DASA)*, 2023, pp. 7–11.

[5] G. Blanc, *et al.*, "Harnessing the Power of Choices in Decision Tree Learning," *arXiv*, 2023.

[6] X. Wu, *et al.*, "Top 10 Algorithms in Data Mining," *Knowledge and Information Systems*, vol. 14, no. 1, 2008, pp. 1–37.

[7] B. C. Esmer, "C4.5," GitHub repository, 2017. Available: https://github.com/barisesmer/C4.5/

[8] S. I. Serengil, "ChefBoost," GitHub repository, 2023. Available: https://github.com/serengil/chefboost

[9] K. Raczycki, "Scikit-learn C4.5 tree classifier," GitHub repository, 2019. Available: https://github.com/RaczeQ/scikit-learn-C4.5-tree-classifier

[10] Geerk, "C45algorithm," GitHub repository, 2019. Available: https://github.com/geerk/C45algorithm

[11] University of Waikato, "WEKA," 2022. Available: https://waikato.github.io/weka-site/index.html

[12] B. Mobasher, "Sample data set," 2005. Available: http://facweb.cs.depaul.edu/mobasher/classes/ect584/weka/classify.html

[13] M. Akturk, "Diabetes Dataset," Kaggle, 2020.

[14] UCI Machine Learning Repository, "Estimation of Obesity Levels Based On Eating Habits and Physical Condition," 2019.

[15] W. Wolberg, W. Street, and O. Mangasarian, "Breast Cancer Wisconsin (Prognostic)," UCI Machine Learning Repository, 1995.

[16] F. Abid, "Algerian Forest Fires," UCI Machine Learning Repository, 2019.

[17] I.-C. Yeh, "Blood Transfusion Service Center," UCI Machine Learning Repository, 2008.

[18] L. Breiman, *et. al. Classification and Regression Trees (Wadsworth Statistics/Probability)*,1984.

[19] J.R. Quinlan, "Simplifying Decision Trees. International," *Journal of Man-Machine Studies*, 27, pp. 221-234.

[20] F. Esposito, D. Malerba and G. Semeraro, "A Comparative Analysis of Methods for Pruning Decision Trees," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 19, No. 5, 1997, pp. 476-491. doi:10.1109/34.589207

[21] M.T. Ribeiro, S. Singh, C. Guestrin, "Why Should I Trust You?": Explaining the Predictions of Any Classifier, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, Pp. 1135 - 1144 https://doi.org/10.1145/2939672.293977

[22] S.M. Lundberg, Su-In Lee, "A unified approach to interpreting model predictions," *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, PP. 4768 - 4777.