Building semi-supervised decision trees with semi-cart algorithm

Abedinia, Aydin; Seydi, Vahid

International Journal of Machine Learning and Cybernetics

DOI:
10.1007/s13042-024-02161-z

E-pub ahead of print: 01/10/2024

Publisher's PDF, also known as Version of record

International Journal of Machine Learning and Cybernetics (2024) 15:4493–4510
https://doi.org/10.1007/s13042-024-02161-z

ORIGINAL ARTICLE

Building semi-supervised decision trees with semi-cart algorithm

Aydin Abedinia[1] · Vahid Seydi[2]

Received: 30 May 2023 / Accepted: 18 March 2024 / Published online: 24 April 2024

Abstract

Decision trees are a fundamental statistical learning tool for addressing classification and regression problems through a recursive partitioning approach that effectively accommodates numerical and categorical data [1, 2]. The Classification and regression tree (CART) algorithm underlies modern Boosting methodologies such as Gradient boosting machine (GBM), Extreme gradient boosting (XGBoost), and Light gradient boosting machine (LightGBM). However, the standard CART algorithm may require improvement due to its inability to learn from unlabeled data. This study proposes several modifications to incorporate test data into the training phase. Specifically, we introduce a method based on Graph-based semi-supervised learning called "Distance-based Weighting," which calculates and removes irrelevant records from the training set to accelerate the training process and improve performance. We present Semi-supervised classification and regression tree (Semi-Cart), a new implementation of CART that constructs a decision tree using weighted training data. We evaluated its performance on thirteen datasets from various domains. Our results demonstrate that Semi-Cart outperforms standard CART methods and contributes to statistical learning.

Keywords Decision Trees · Semi-Supervised · CART · Generalization · Semi-CART · Distance-based Weighting

1 Introduction

Supervised Learning algorithms have proven effective in significant labeled data scenarios. Conversely, Semi-Supervised Learning algorithms have exhibited a superior ability to generate highly-accurate predictions due to utilizing both labeled and unlabeled data sets. A crucial issue in semi-supervised learning algorithms is effectively utilizing the unlabeled data during the training phase. Even though labeling data incurs prohibitively high costs, is time-intensive, or is infeasible, semi-supervised learning presents a considerable advantage over supervised learning due to the significantly greater access to unlabeled data. It allows for an efficient and cost-effective data acquisition method, making it a preferred choice in varied scenarios. Semi-supervised learning has introduced several algorithms, such as Self-training [3], co-training [4], Pseudo-labeling [5], and Label Propagation [6]. Self-training is a widely used method in semi-supervised learning by iteratively assigning pseudo labels to unlabeled samples. It is used to mitigate the requirement for labeled data, which can be time-consuming and labor-exhaustive to obtain practical tasks [7]. Co-training is a framework for semi-supervised learning that extends from self-training. It works by training two classifiers separately on different views of the data and using the predictions of either classifier on unlabeled instances to augment the training set of the other [4]. Pseudo-labeling is a general approach to SSL[1] that does not rely on domain-specific data augmentations. It generates hard pseudo-labels for unlabeled data and uses them as accurate labels. However, in its original formulation, pseudo-labeling performs relatively poorly due to erroneous high-confidence predictions from poorly calibrated models; these predictions generate many incorrect pseudo-labels, leading to noisy training [8]. Label Propagation is a widely used graph-based approach to SSL. It works by constructing a nearest neighbor graph of the dataset and propagating labels along the data manifold. Many variations of label propagation algorithms have been developed,

✉ Vahid Seydi
v.seydi@bangor.ac.uk

Aydin Abedinia
abedinia.aydin@icloud.com

[1] Department of Computer, Islamic Azad University-South Tehran Branch, Tehran, Iran

[2] Centre for Applied Marine Sciences, School of Ocean Sciences, Bangor University, Menai Bridge, UK

[1] Semi-Supervised Learning.

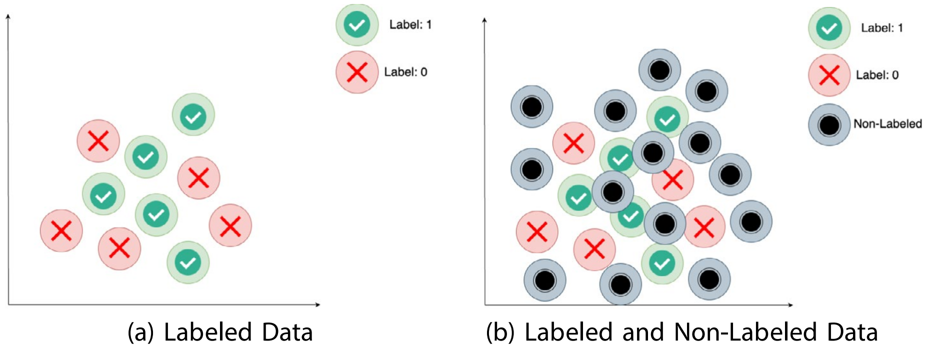


Fig. 1 Training with Labeled and Non-Labeled data

focusing mainly on constructing the similarity matrix and integrating label propagation with other methods [6].

Figure 1 depicts the dataset used for supervised and semi-supervised learning algorithms. Figure 1(a) shows the training dataset used for supervised learning algorithms, representing a relatively small number of points. In contrast, Fig. 1(b) displays the labeled and unlabeled points, illuminating the potential use of semi-supervised algorithms. Semi-supervised learning methods, including self-training, co-training, Moreover pseudo-labeling, rely on datasets similar to that depicted in Fig. 1(b). Utilizing both labeled and unlabeled data, semi-supervised learning techniques offer an advantageous approach for training models by taking advantage of a larger, combined pool of labeled and unlabeled data points unavailable during supervised training approaches. At its core, semi-supervised learning capitalizes on the availability of labeled and unlabeled data, thereby enhancing the accuracy of the developed models relative to those that only rely upon labeled data.

Decision trees are commonly used and effective methods for classifying statistical data in machine learning. Due to their capability of representing complex decision-making processes easily and understandably, decision trees have gained popularity and been broadly adopted in various domains, including finance, healthcare, and social sciences. Multiple decision tree algorithms have been developed for constructing decision trees, including ID3,[2] C4.5, C5, and CART,[3] which is the algorithm that we focused on in our methodology. The CART algorithm uses the GINI Index method to select the best feature and value in each step of building a tree; The GINI index measures the impurity of a set of data. It selects the feature and value that split the data most effectively [9]. The research introduces the Distance-based Weighting method, which proposed calculating the training row weight by computing its distance from the test dataset and removing unsalvageable training rows. The objective is to use high-relevance, high-quality data for model training. We drew inspiration from semi-supervised algorithms and subsequently proposed a new algorithm for constructing a tree utilizing CART, incorporating calculated weights. The study illustrates that the proposed method can effectively enhance the CART algorithm's accuracy.

We introduce Semi-CART, a new algorithm for constructing decision trees based on CART's fundamental principles. Semi-CART utilizes a new Gini index equation during the training process to use calculated weights, allowing for improved feature and value splitting. Notably, Semi-CART outperforms standard CART in terms of accuracy by utilizing better candidates for splitting and predicting unseen datasets. The Semi-Cart is evaluated on thirteen different datasets of various fields. The experimental results demonstrate the superior performance of the proposed method compared to other state-of-the-art algorithms.

The subsequent sections of this paper are structured in the following manner. Section 2 highlights related works, including other approaches for Semi-Supervised learning and decision tree algorithms. Section 3 introduces the proposed method, including a detailed description of the Distance-based Weighting algorithm, the new Gini index equation, and the Semi-Cart algorithm. Section 4 describes the experimental results, including the datasets and evaluation metrics. Section 5 briefly discusses future works, including the possible extension of the proposed method to other machine learning algorithms. Section 6 concludes the paper with discussions of the contributions and limitations of the proposed method.

2 Related works

We conduct a comprehensive literature review on decision trees and semi-supervised learning techniques. We emphasize highlighting the importance of these concepts in developing and refining our proposed algorithms, namely Distance-based Weighting and SemiCart, as they serve as the fundamental principles underlying these approaches.

[2] Iterative Dichotomiser 3.

[3] Classification and Regression Tree algorithm.

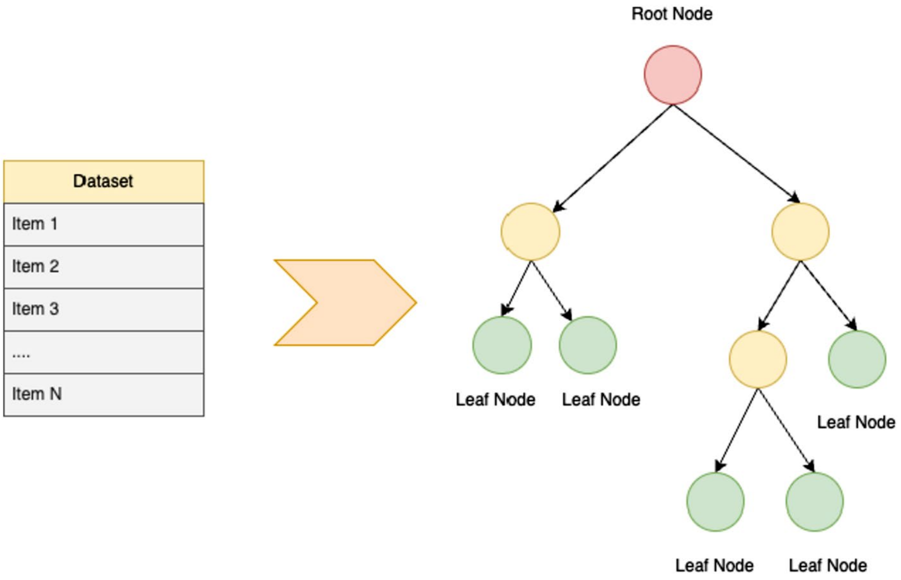


Fig. 2 Dataset to tree structure

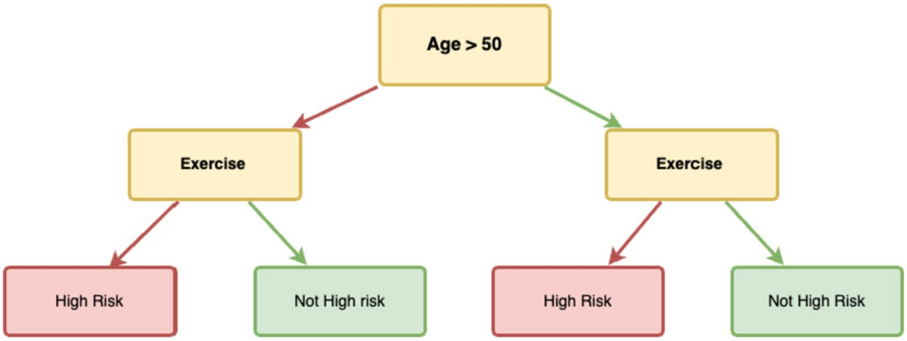


Fig. 3 Simple binary decision tree

2.1 Decision trees

Decision trees have been used in machine learning for several decades and have roots in various fields, such as mathematics and philosophy. Morgan and Sonquist in paper "Problems in the analysis of survey data, and a proposal" introduced decision trees for machine learning in 1963, Their premise was founded on the notion that decision trees can aid in identifying the critical factors influencing a specific outcome and subsequently employ this knowledge in forecasting future events [10]. Figure 2 illustrates the process for constructing a tree data structure using the Tree Builder algorithm. The algorithm receives the dataset as input and generates a tree structure from the root node to the leaf nodes. Each leaf node corresponds to a particular value that can be utilized for making predictions.

Decision tree algorithms are commonly used in machine learning for statistical data classification and regression. These algorithms represent models in a tree structure, where each node in the tree represents a decision or a test on a particular attribute, and the edges represent the outcomes of those tests. The tree structure allows for the straightforward representation of complex decision-making processes, making decision trees popular in various domains. Decision trees can handle numerical and categorical data, making them versatile tools for classification tasks. Decision tree classification is based on decision tree induction, which involves learning decision trees from class-labeled training data. A decision tree is represented as a graphical tree structure in which each outcome of a test and each leaf node (or terminal node) holds a class label. A simple decision tree to detect high risk and not a high risk is depicted in Figure 3 by asking two questions, the first one is age, and the second one is doing exercise.

Quinlan's ID3 algorithm, published in 1986, is considered one of the earliest and most popular decision tree algorithms. The ID3 algorithm recursively splits the dataset based on the attribute that provides the maximum information gain, which is calculated using the entropy measure. The resulting decision tree can be used for classification and can also provide insights into the most important attributes of the classification task [11]. In the paper "Decision Tree Induction: A

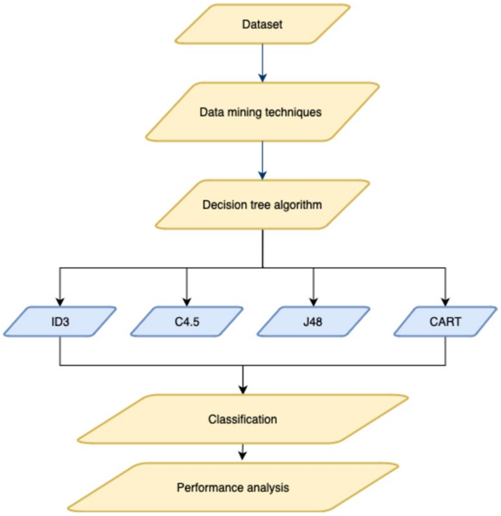


Fig. 4 Methodology of decision trees

Comparative Study of ID3, C4.5, and CART", The authors evaluated the algorithms based on accuracy, tree size, and computational time on datasets from the UCI Machine Learning Repository.[4] They found that their performance varied depending on the dataset characteristics. While ID3 outperformed C4.5 and CART for some datasets, C4.5 or CART was the better choice for others. The authors found that C4.5 produced smaller trees than ID3 and CART, which may benefit interpretability and efficiency. CART was the fastest algorithm in terms of computational time. The study also evaluated the effectiveness of different splitting criteria and found that information gain was the most effective for all three algorithms, although the gain ratio performed similarly for C4.5. The authors noted that the performance of these algorithms depends on the dataset and problem characteristics [9]. As shown in Fig. 4, the algorithms employed for constructing decision trees can handle multiple data types. The ID3 algorithm is designed to work with categorical datasets, whereas the C4.5 algorithm can work with both continuous and categorical datasets [2]. On the other hand, CART can handle nominal attribute datasets, as well as continuous datasets with categorical labels to generate decision trees [12].

2.2 CART

The Classification and Regression Trees (CART) algorithm, first proposed by Leo Breiman in 1984, is a powerful tool for developing decision trees to handle both categorical and numerical data. In the realm of machine learning problems, classification stands as a significant concern. Calculating the most appropriate candidate for splitting data into decision trees is crucial. The CART algorithm uses the statistical measures of Gini impurity and entropy to construct a decision tree. These approaches assist in identifying the most appropriate node splits to be used in building the tree. Specifically, the CART algorithm uses the Gini Index equation to construct a decision tree, as shown in Eq. 1. The GINI Index Eq. 1 measures the impurity of a dataset, thereby enabling the selection of the optimal candidate for splitting. In Eq. 1, "pi" denotes the probability of rows sharing the same label. For instance, in 3/8, "three" represents three rows with the same label, divided by the total count of rows. The binary trees produced by the CART algorithm exhibit several prominent features, such as supporting boosting, pruning, and average speed. Breiman, the founder of the CART algorithm, emphasized that "a key criterion for a good classification procedure is that it not only produces accurate classifiers (within the limits of the data) but also provides insight and understanding of the predictive structure of the data [13].

$$Gini(p) = 1 - \sum_{i=1}^{N} \mathrm{p}^2 i \tag{1}$$

The standard CART algorithm is a widely used decision tree tool, popular due to its ability to handle missing values present in datasets efficiently. However, a drawback of CART is its instability, which results in minor modifications in data leading to significant changes in the resultant decision tree [14]. In response to the instability issue, Guolin Ke introduced two novel techniques, Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB), in the LightGBM paper. GOSS disregards a significant proportion of data instances with low gradients and only uses the remaining samples to calculate the information gain [15]. Our proposed methodology leverages the concept of GOSS and introduces the Distance-based Weighting technique to eliminate redundant rows from training data, as discussed in Sect. 3. The CART algorithm follows several steps in creating a decision tree model. Initially, the dataset is split into two subsets—the training dataset for building the tree and the testing dataset for considering model performance. Next, the algorithm selects the optimal feature to split the dataset based on the GINI index. This process is carried out recursively on the resulting subsets until a stopping criterion, like the maximum depth or the minimum number of samples for splitting a node, is met. The decision tree is then pruned to enhance performance by removing branches with low accuracy contributions. Finally, the decision tree model predicts class labels or continuous.

The CART is popular in boosting algorithms to construct weak learners. Boosted Trees is a machine learning method that combines several decision trees sequentially, with each

[4] https://archive.ics.uci.edu/.

tree considered a weak learner [16]. Similarly, XGBoost, a widely adopted algorithm, uses CART to create decision trees [17]. LightGBM, developed by Microsoft Research, also relies on CART to produce weak learners [15]. The adaptability of the CART algorithm and its ability to support boosting techniques contribute to its popularity among machine learning specialists. Boosting is an ensemble method that blends weak and strong learners to achieve greater accuracy, as emphasized in [15].

2.3 Semi-supervised

Semi-supervised learning is a machine learning technique class that addresses scenarios where labeled data is limited. In contrast, a significant amount of unlabeled data is available. In some scenarios, acquiring labeled data for supervised learning models can be demanding and resource-intensive. Researchers have developed SSL methodologies to address limitations that leverage labeled and non-labeled data to improve model performance. By integrating both data types, the resultant model can make more accurate predictions, especially when dealing with prediction tasks that do not have enough labeled training data. SSL is a regularization technique that leverages the information inherent in unmarked data. The goal of SSL is to minimize the risk of overfitting, which occurs when the machine learning model is excessively intricate and adequately fits the training data. Such scenarios often lead to underperformance when presented with new data [18]. Incorporating a small set of labeled data alongside a larger set of unlabeled data during machine learning model training has enhanced test accuracy. This approach is advantageous because it requires fewer labeled data than Temporal Ensembling [19]. Various approaches are employed within semi-supervised learning, such as Self-training, Co-Training, and Graph-Based methods, with the field continuously advancing through developing novel techniques.

Self-training is a machine learning technique that is widely employed in semi-supervised learning. This approach utilizes labeled data to generate a model capable of assigning a label to unlabeled data. The self-training method trains a model with a small set of labeled data. The trained model is then applied to label the remaining unlabeled data. The new labeled data is added to the labeled data set, expanding the dataset for the next iteration. The iterative procedure is executed repetitively until a desirable degree of precision is attained. However, over-relying on self-directed learning can limit the model's effectiveness, such as the possibility of error propagation and the assumption that the model's predictions on the unlabeled data are consistently precise. These limitations may compromise the model's overall performance and impede its ability to infer accurate predictions. Nevertheless, self-training remains an essential technique in semi-supervised learning, particularly in cases where labeled data is scarce and expensive. These limitations and benefits of self-training are well documented in the literature, as highlighted in [20].

Co-training is a semi-supervised learning technique that involves training two separate models on different data views, each using a small amount of labeled data and then unlabeled data to iteratively improve the models' performance by cross-validating each other's predictions. The approach assumes that each view of the data is conditionally independent given the class labels, which allows the models to learn from each other and reduce the need for labeled data [21]. Co-training is a popular technique employed in semi-supervised learning methods. While it is a useful approach, there may be better fits for some data or learning task types. The careful selection of training sets and views is critical to the success of this method.

Graph-based methods constitute a popular semi-supervised learning technique whereby the data structure of a graph is harnessed to propagate labels from labeled data points to unlabeled ones. By constructing a graph with data points represented as nodes and edges connecting similar data points, the graph-based methods utilize similarity metrics, such as Euclidean distance or cosine similarity, to assign weights to the edges. In practice, the labeled data points function to initialize labels of the nodes, while the labels of unlabeled data points are determined through label propagation. Laplacian regularization and label propagation are some common graph-based methods. The former involves adding a regularization term to the learning objective to penalize estimates that differ from the suppositions of the graph structure. The latter utilizes the Laplacian matrix of the graph to smooth the labels across the graph. Formulating the labels of unlabeled data points entails solving a linear system comprising the Laplacian matrix and the labels of the labeled data points. Nonetheless, the effectiveness of these methods largely depends on the quality of graph connectivity and the similarity metric adopted. Therefore, graph construction and parameter adaptation require careful attention to attain desirable outcomes [22, 23]. Inspired by the Graph-based methods, we developed a distance-based weights method to compute weights and eliminate noisy training instances.

2.4 Semi-supervised decision trees

Decision tree algorithms such as ID3, C4.5, and the CART are supervised machine learning techniques that rely on labeled data for learning and prediction. However, limited labeled datasets can hamper their performance. Combining decision trees with semi-supervised learning strategies has been proposed to enhance performance and mitigate this difficulty. Integrating semi-supervised learning methods can improve the decision tree model's performance by leveraging the untapped information within the unlabeled data. These approaches provide improved classification power and generalization compared to rigidly supervised methods.

The authors of "Semi-supervised Self-training for decision tree classifiers" propose a novel methodology to enhance decision tree classification accuracy in cases where labeled data is scarce. This strategy integrates semi-supervised techniques with self-training to employ the information in a substantial amount of unlabeled data to improve the classifier's classification accuracy. The self-training algorithm is utilized iteratively to add only the most confident predictions of the decision tree classifier to the labeled dataset, consequently amplifying the size of the labeled dataset and overall accuracy of the decision tree classifier. Assessments of this model were conducted on various datasets, and results demonstrated that the semi-supervised self-training methodology significantly enhances accuracy compared to the standard supervised approach. Specifically, the proposed method achieved an average accuracy improvement of 6.3% on the tested datasets. In summary, this research provides a contemporary approach to augment decision tree classifier accuracy when only a limited amount of labeled data is present. The proposed self-training method combines the advantages of semi-supervised techniques with the iterative self-training algorithm, enhancing the classification accuracy of decision tree models [24]. In our research, the proposed Semi-CART algorithm employs a novel approach by integrating weights derived from test data into the training set, significantly improving the model's performance. This methodology strategically refrains from utilizing unreliable predicted labels for training, ensuring that the model's generalization capabilities are independent of label prediction accuracy.

The authors of "Ensemble of decision tree reveals potential miRNA-disease associations" propose a semi-supervised learning approach to predict potential miRNA-disease associations. They address the limitations of traditional supervised learning methods, which require a great deal of labeled data that is often costly and time-consuming. To overcome this limitation, the authors introduce semi-supervised learning methods that harness labeled and unlabeled data to improve prediction accuracy. Their technique involves partitioning the feature space into multiple subspaces Using the random subspace method. They assemble a decision tree for each subspace utilizing labeled and unlabeled data, then aggregate the decision trees into an ensemble via majority voting on their predictions. To tackle the class imbalance problem, the authors incorporate an oversampling technique known as the "Synthetic Minority Over-sampling Technique" (SMOTE). The authors evaluate their approach using a miRNA-disease association dataset and compare it with several state-of-the-art methods. The results show that their ensemble of decision trees outperforms all the other ways in terms of AUC, F1-score, and precision-recall curve. Overall, the paper demonstrates the effectiveness of using SSL in combination with ensemble learning for predicting miRNA-disease associations [25]. In "Ensemble of Decision Trees Reveals Potential miRNA-Disease Associations," the study highlights that prediction accuracy is closely linked to input data quality, with incomplete or biased data leading to potential inaccuracies. In contrast, our Semi-CART method's performance has been rigorously tested across diverse datasets featuring varying label ratios to validate its robustness and accuracy.

Kim, Kyoungok introduces, in the paper titled "A Hybrid Classification Algorithm by Subspace Partitioning through Semi-supervised Decision Tree," a new algorithm that combines the strengths of decision trees and subspace partitioning techniques in a semi-supervised manner. The proposed approach involves constructing a decision tree using the labeled data sets. Based on the obtained decision tree, the algorithm proceeds to partition the unlabeled data into appropriate subspaces. The resulting subspaces are classified by applying subspace partitioning methods such as k-NN or SVM, utilizing the respective labeled and unlabeled data sets. The study's findings reveal that the presented algorithm surpasses traditional decision tree classifiers and semi-supervised algorithms, such as self-training and co-training, in terms of performance. Furthermore, the research demonstrates that the proposed algorithm exhibits a remarkable impact, particularly when labeled data is scarce and the abundance of unlabeled data points is high [26]. In the study "A Hybrid Classification Algorithm by Subspace Partitioning through Semi-supervised Decision Tree," a critical limitation is identified in the traditional decision tree algorithm's capacity for subspace partitioning, which often results in subsets lacking uniform structural or topological properties. Our proposed methodology diverges from conventional space partitioning techniques instead of utilizing weights calculated from the training set to the test set. This innovative approach, coupled with a comprehensive array of distance measurement techniques, is specifically designed to rectify these identified shortcomings, thereby significantly enhancing the overall efficacy of the model.

The research paper "Fast Semi-Supervised Self-Training Algorithm Based on Data Editing" introduces an innovative algorithm within semi-supervised learning, a technique that integrates labeled and unlabeled data for training purposes. This study details the development of a rapid self-training algorithm, which employs data editing to enhance the quality of high-confidence samples. Drawing inspiration from the Ball-k-means algorithm, it introduces a ball-cluster partitioning and editing method. A notable aspect of this algorithm is its linear time complexity relative to the sample size. Comprehensive experimental evaluations conducted on 20 benchmark datasets demonstrate that this algorithm exhibits expedited processing speeds and outperforms comparative algorithms in classification efficacy [27]. In semi-supervised learning, self-training methods reliant on high-confidence pseudo-labels for unlabeled data often face the challenge of label noise, leading to error accumulation during iterative training. Such methods also exhibit susceptibility to training instability, compromising their reliability. Additionally, bias in semi-supervised learning arises from intrinsic problem characteristics and potentially incorrect pseudo-labels. Contrarily, our Semi-CART method diverges from using pseudo labels, known for their instability across datasets, thus ensuring more excellent reliability and consistent performance across various datasets.

The research delineated in "Semi-Supervised Learning with Decision Trees: Graph Laplacian Tree Alternating Optimization" introduces an innovative semi-supervised learning paradigm for decision trees. This framework is characterized by its scalability and efficacy, particularly notable in contexts with minimal labeled instances. It facilitates the derivation of precise and interpretable models predicated on decision trees under such constraints. The resolution of the underlying problem is achieved through a novel reformulation, necessitating the iterative resolution of two less complex sub-problems: a supervised tree learning problem, addressable via the Tree Alternating Optimization algorithm, and a label smoothing problem, resolvable by engaging a sparse linear system. This methodological approach directly integrates the labeled data into the operator, bolstering the efficacy of semi-supervised learning through spectral clustering techniques [28].

Contrasting with traditional methodologies, where the algorithm labels the unlabeled data by minimizing a quadratic objective influenced by the graph's structure, our proposed approach diverges significantly. The conventional method's reliance on this procedure results in polynomial complexity correlated with the sample size n, potentially leading to substantial computational demands in large datasets. Our method, in contrast, calculates weights derived from the test dataset, distinctively abstaining from predicting labels for the test data. These computed weights are then employed in the training of decision trees. Our methodology has demonstrated superior performance over the traditional approach across various datasets through comprehensive empirical assessments, showing particular effectiveness in situations involving varying label ratios.

3 Methodology

We will introduce the "Distance-based Weighting" algorithm for training decision trees in semi-supervised settings. One of the challenges in such settings is dealing with unlabeled datasets in the training phase. We utilize a "Distance-based Weighting" algorithm to overcome this challenge that calculates similarities between training and test data instances. We assign weights to the training instances based on these similarities, which are added to the training dataset as "weights." The algorithm then removes training instances with zero weights to increase model efficiency and speed up the training process with fewer data. We refer to these steps as the "Distance-based Weighting" methodology. Our proposed algorithm aims to improve decision tree learning efficiency and accuracy in semi-supervised settings while utilizing unlabeled data. We introduce a novel algorithm called "Semi-CART," designed to augment the predictive ability of the decision tree model. Our methodology incorporates a weight column into the decision tree model to improve prediction accuracy significantly. Specifically, we adapt the GINI impurity equation to include the weight-based computations during the learning phase of Semi-CART. The resulting algorithm can enhance prediction accuracy by integrating relevant weight-related factors into the decision tree model.

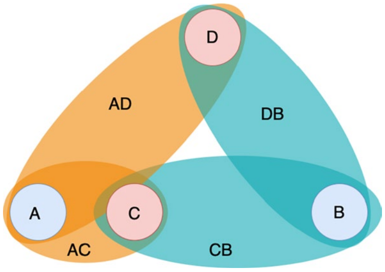


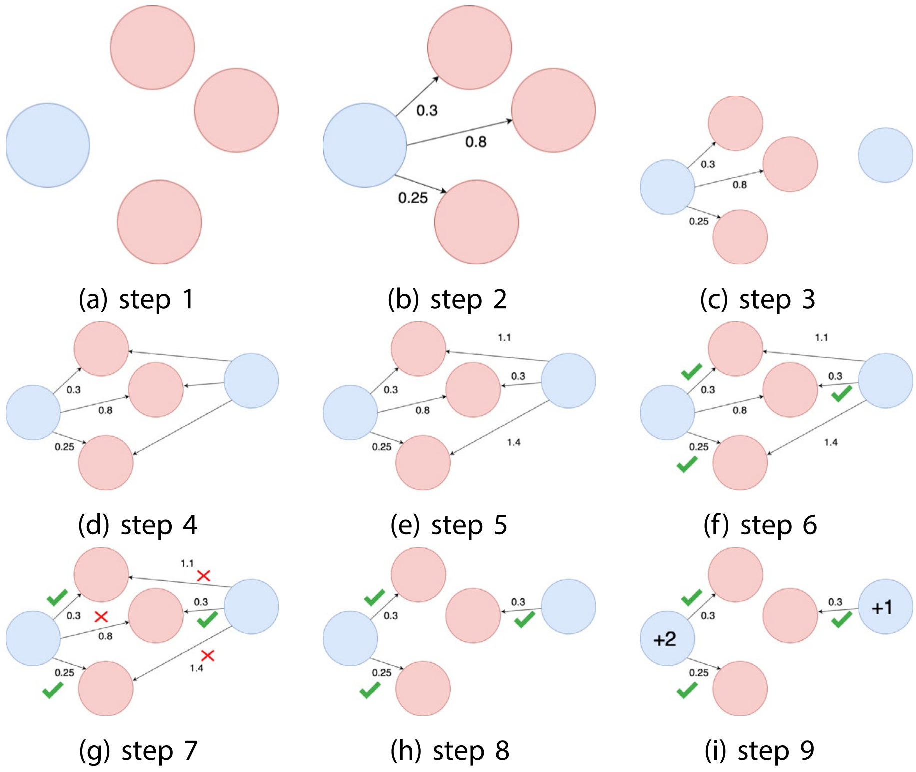
Fig. 5 Distance measuring in weighing algorithm

3.1 Distance-based weighting

The "Distance-based Weighting" method is a novel approach that facilitates a non-streaming process that considers the similarities between training and test data points. Specifically, the methodology uses the Euclidean distance metric to calculate the similarity between each training record and the test dataset. Furthermore, each training row is append one as a weight value based on its similarity to the test data, which ranges from zero to the total number of test data points. A zero weight value implies that the training row does not correspond closely with any test data points and is therefore excluded from the training process. Our proposed approach allows for removing training data that may interrupt model accuracy, hence filtering out irrelevant data noise and improving the decision tree's predictive capacity, particularly for an extensive test dataset.

Figure 5 portrays four data points, where the training set comprises points A and B, and points C and D represent the

Fig. 6 Steps of distance-based weighting algorithm works



(a) step 1 (b) step 2 (c) step 3

(d) step 4 (e) step 5 (f) step 6

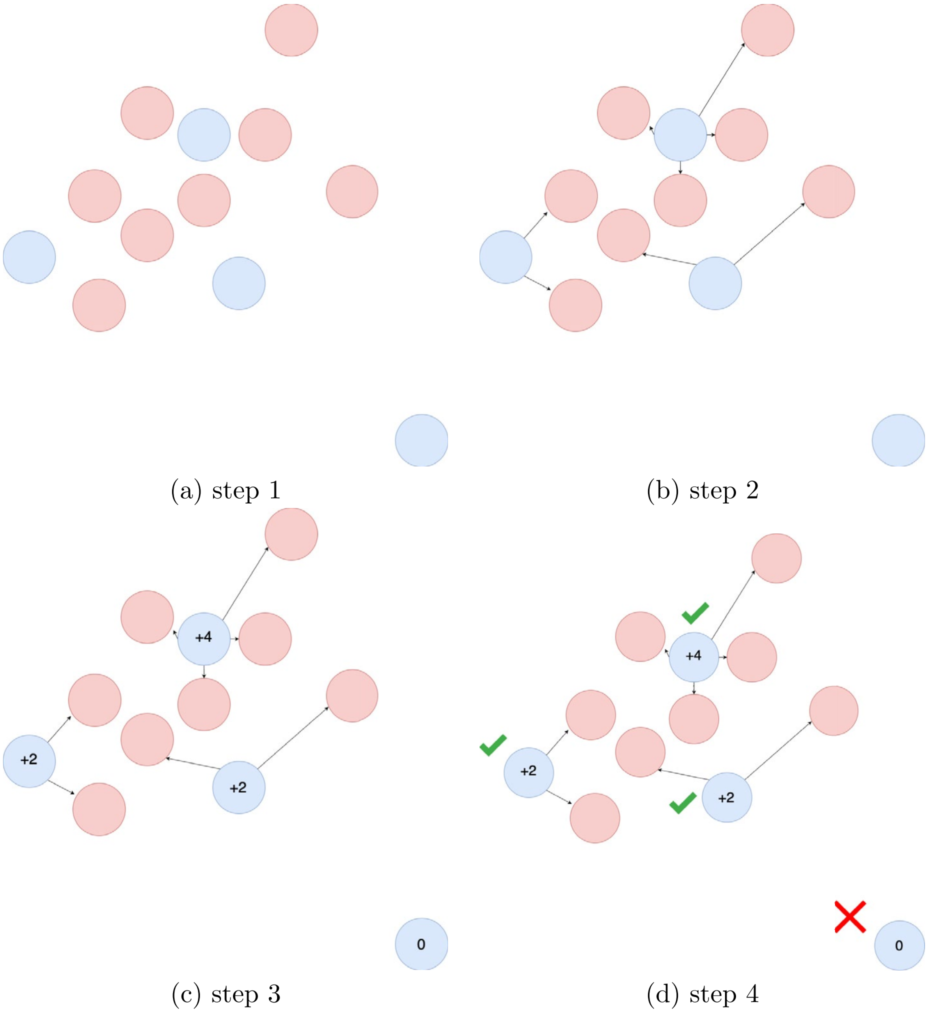
(g) step 7 (h) step 8 (i) step 9

test set. The Distance-based Weighting algorithm adopted in our methodology employs the Euclidean distance metric. The computation results show that points A and C (AC) are closer to C and B (CB). Similarly, D and B (DB) are found to be closer to A and D (AD). The neighbor one approach adds +1 to a node's weights with the closest test data. In the first step, the algorithm adds +1 to the weight of point A because of its proximity to C. Consequently, point C is removed, and point B is selected in the next step. In the second step, the algorithm compares the distances AD and DB and adds +1 to the weight of B while removing point D. This process is repeated until all test data is used. The neighbor two approaches are similar, except that in each step, the algorithm adds +1 to the weights of the two nearest neighbors and removes the used test data after each iteration. The count of test data measures the training instance's proximity to the entire set of test data. The Distance-based Weighting algorithm is responsible for computing the nearest neighbors of each training row concerning a given test data. Ultimately, this enables the Distance-based Weighting algorithm to assign appropriate weights to each training instance based on their proximity to the test data. The proposed algorithm eliminates redundant or irrelevant instances in the training dataset by assigning zero weights to the corresponding rows. The resultant refined dataset not only expedites the learning process but also improves the accuracy of the prediction model. Upon augmenting the number of neighbors to three, every training instance determines its closest neighbor using the Euclidean distance equation. The proposed Distance-based Weighting algorithm computes the distances between every record in the training set and the three closest neighbors in the test set. Subsequently, it excludes training instances that receive zero weights, thus refining the dataset for training. This approach is preferred over using one of the nearest neighbors because data scattering carries more weight in three neighbors. Moreover, rows with zero weight are less common than in one neighbor. As the nearest neighbors' value increases, the probability of individual training rows being the closest neighbors increases, thereby avoiding their removal in the purge process. To illustrate how the weights are assigned and how the training data is purged, we visualize the process in the following section.

Figure 6 illustrates the distribution of training and test data points on a chart, with training data marked in blue and test data marked in red. Subsequently, one neighbor is drawn in Fig. 6(a), along with the weights assigned to the training data. The proposed Distance-based Weighting algorithm employs the Euclidean distance measurement method, similar to the KNN[5] algorithm. The algorithm calculates the distance between two points in the multidimensional feature space by considering all the feature values, as shown in Eq. 6(b). Additionally, Fig. 6(c) presents a new training point on the right side, which needs to be considered

[5] K-Nearest Neighbor.

Fig. 7 Steps of distance-based weighting algorithm with more data points



(a) step 1

(b) step 2

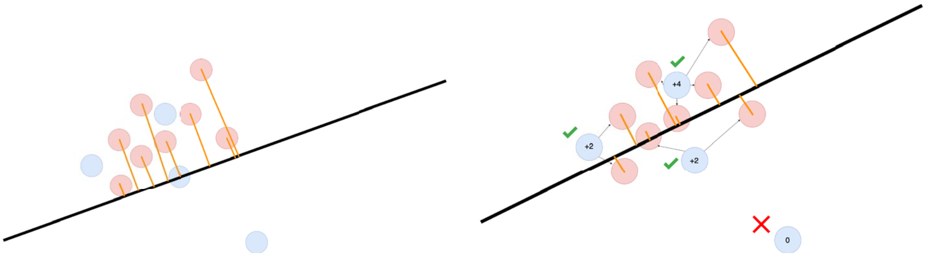
(c) step 3

(d) step 4

for calculating the distance via the Distance-based Weighting algorithm. Subsequently, in Fig. 6(e), the distance values between the new training row and three test rows are depicted. Next, the Distance-based Weighting algorithm selects the smallest distance, as shown in Fig. 6(f), and presents the status of each line in Fig. 6(g). Subsequently, long-distance lines are removed in Fig. 6(h), and the Distance-based Weighting algorithm calculates the weight of training rows in Fig. 6(i). If a training row is significantly far from test records, its weight is calculated with a zero value. The Distance-based Weighting preprocessing algorithm, as previously discussed, utilizes the Euclidean distance to assign weights to the training data. It can utilize other distance measurements techniques like Manhattan, Mahalanobis, Cosine, Jaccard, and Hamming.

Figure 7 depicts an expanded view of the training and test data points. Specifically, Fig. 7(a) presents four points from the training set and eight points from the test set. Subsequently, in Fig. 7(b), the Distance-based Weighting algorithm computes the distance between each training instance and its nearest test point. Figure 7(c) displays the training points with weights assigned by the Semi-Cart approach for the training phase. Conversely, in Fig. 7(d), a blue point in the lower-left corner has a zero weight due to its long distance from all test sets. The Distance-based Weighting algorithm draws inspiration from GOSS to eliminate redundant training points, resulting in a cleaner dataset that enhances the accuracy of the constructed tree. Figure 8 presents a simple baseline whereby a straight line is trained from all the training data points. In Fig. 8(a), this baseline is visualized and compared against the test data points represented by the orange lines. It is observed that the baseline exhibits significant errors when compared to the test data points. The Distance-based Weighting algorithm is applied to remove noisy data points from the training set to address this issue. As shown in Fig. 8(b), this approach yields a cleaner dataset for training, resulting in a baseline that performs better than the noisy dataset baseline. Specifically, the errors between the baseline and the test data points are reduced, indicating the Distance-based Weighting algorithm's efficacy in improving the trained model's accuracy.

Fig. 8 Baselines for noisy data vs. clean data



(a) baseline with noisy dataset

(b) baseline without noise

3.2 Semi-cart

Improving the accuracy of decision trees is a crucial area of research due to its widespread applications in different fields. In this section, we introduce a new algorithm called Semi-CART, which aims to improve the accuracy of decision trees in classification and regression problems. One of the critical components of Semi-CART is the Distance-based Weighting algorithm, which calculates the distance between test data and training data and removes ineffective training rows. This preprocessing step ensures that only valuable data is used in training, which can lead to improved accuracy in the resulting decision tree. In the standard CART algorithm, GINI impurity is used to split the dataset and select the best candidate for each split based on information gain. Semi-CART builds upon this by using a new formula for GINI impurity that considers the weights of the training rows. Specifically, the new formula Eq. 2 replaces "pi" with "w/S," where "w" is the weight of the training row and "S" is the sum of all train data weights. The implementation of Semi-CART is described with pseudo-code, and we show that it outperforms the traditional CART algorithm in terms of accuracy on thirteen different datasets with varying features and rows.

Algorithm 1 Distance-based weighting: set weights to train data

```
Require: $k$: number of neighbors, $c$: similarity ratio, $train$: train data, $test$: test data
Ensure: $train$: weighted train data
$train = train$.copy()
$train$['weight'] = 0
for each $i$ in $test$ do
    for each $j$ in $train$ do
        $train_d = train$.nsmallest(k, ['distance', i])
        for each neighbor in $train_d$ do
            neighbor['weight'] += $c$
        end for
    end for
end for
Return $train$
```

$$GINI = 1 - \sum_{i=1}^{N} (\mathrm{w}/\mathrm{S})^2 i \tag{2}$$

Moreover, We discuss the benefits of using the Distance-based Weighting algorithm and how it can remove useless or noisy training data, leading to a more accurate decision tree. The Distance-based Weighting algorithm benefits non-streaming processes and large datasets with noisy data. The Semi-Cart method uses the new Gini index formula, which utilizes weights to select a better value for separating the tree. This approach can be efficient when dealing with extensive test data sets and when the test and training data distribution is similar. By incorporating weights into decision-making, the Semi-Cart method can effectively identify and remove noisy or irrelevant data points, resulting in a cleaner and more accurate dataset. The Semi-Cart method offers a robust and efficient approach to building decision trees, mainly when dealing with complex or high-dimensional data sets. In the forthcoming section, we will present the comparison results between Semi-CART and CART on various datasets. Additionally, we will visually demonstrate the accuracy outcomes and the optimal number of neighbors for which Semi-Cart outperforms CART.

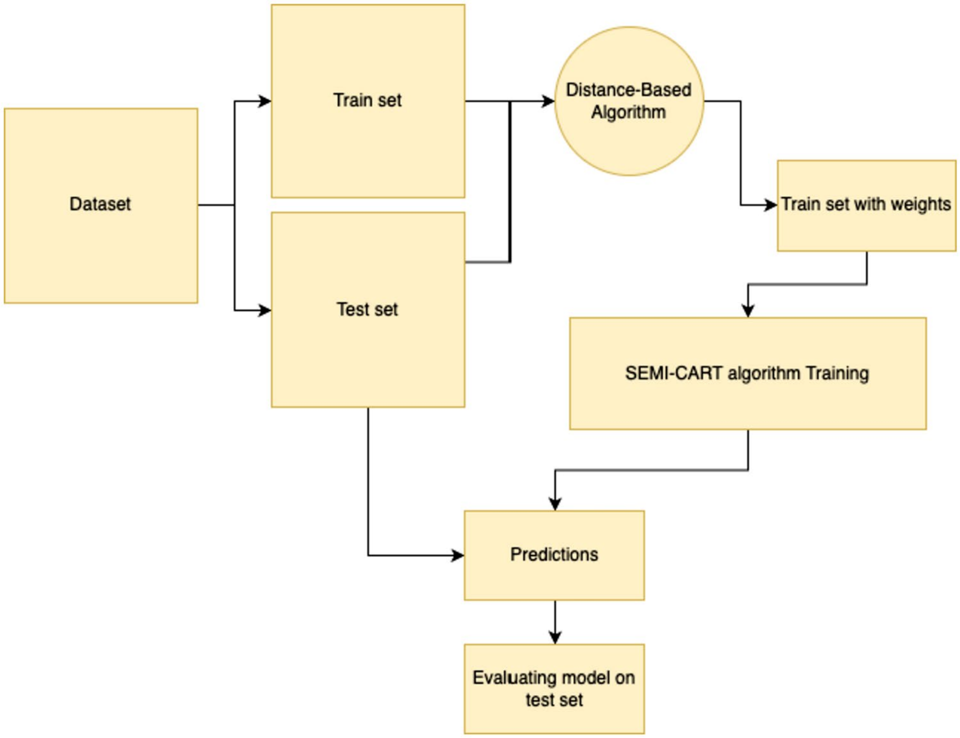


Fig. 9 Training and evaluating of distance weighted algorithm and semi-CART

Algorithm 2 Weighting alg: remove useless data

```
Require: $train$: train data
Ensure: $train$: train data without useless data
$train = train$.copy()
for each $i$ in $train$ do
    if $train[i]['weight'] == 0$ then
        Remove $train[i]$
    end if
end for
Return $train$
```

In summarizing the effectiveness and methodology of the distance-based weighting algorithm within our proposed framework, it is pivotal to outline the sequential steps and their impact on the algorithm's performance. Initially, the dataset is collated and partitioned into training and test subsets. This approach's core hinges on applying the distance-based weighting algorithm. This algorithm employs a diverse array of methods, including Mahalanobis, city block, Euclidean, squared Euclidean, cosine, correlation, Hamming, Jaccard, Chebyshev, Canberra, matching, dice, Rogers-Tanimoto, Russell-Rao, Sokal-Michener, and Sokal-Sneath, to calculate the distances between each element in the training set and the test set. This calculation is instrumental in determining the proximity of each training set element to the test set. For instance, a training row assigned a weight of 1 indicates it is the nearest neighbor to a record in the test set. Conversely, a training set element with a weight of 34 signifies its closest proximity to 35 records in the test set, highlighting its greater significance during the training phase. Upon computing these weights, the training set retains its sample size but varies in column size due to the inclusion of weights. This necessitates an adaptation of the Gini formula to accommodate these weights. Unlike traditional approaches where dataset splitting is based on the value of a candidate with weight 1, our modified semi-CART approach utilizes candidates with higher weights for dataset division. Finally, the model is evaluated using the test dataset upon concluding the training process. This strategy, which integrates the calculated weights, allows for predictions that align with the distribution of the test dataset, thereby enhancing the model's accuracy and relevance all these steps have been depicted in 9.

Table 1 Dataset division into training and testing sets

Dataset	Total length	Training length	Test length	Unnoisy length
Banknote	1371	685	686	685
Fertility	100	50	50	48
Wdbc	569	284	285	280
Biodeg	1055	527	528	527
Haberman	305	152	153	140
Transfusion	748	374	374	323
Hepatitis	80	40	40	37
Tictactoe	957	478	479	457
Vote	232	116	116	97
Bupa	344	172	172	147
Breast	276	138	138	123
Glass	213	106	107	93
Mammographic_ masses	829	414	415	389

4 Results

In this section, we delineate the outcomes of our experimental research, wherein we juxtaposed the performance of the Semi-CART and CART algorithms across a spectrum of datasets. These datasets were characterized by a diversity in the number of features and rows. A pivotal aspect of our methodology involved the implementation of the Distance-based Weighting algorithm. This algorithm was crucial in excluding non-contributory training rows and in the computation of weights, which were determined based on the proximity between the test and training data sets. Table 1 provides a comprehensive overview of thirteen distinct datasets, each varying in the size of their features. Our study systematically divided the dataset into training and testing subsets. This division, crucial for assessing algorithmic efficacy, is visually depicted in Fig. 10.

As previously noted, the datasets employed in our study exhibit variability in the number of features and records. Additionally, each dataset is characterized by a distinct ratio of labels. Figure 11 provides a clearer understanding of these variations. This figure graphically represents the label ratios across the various datasets, visually elucidating the disparities and patterns within each dataset's composition. This visualization aids in a more comprehensive interpretation of the dataset characteristics and their potential influence on the research outcomes.

As we can see, we have balanced and imbalanced datasets; in Table 2, the results of cart vs. semi-cart have been mentioned. In our comparative analysis of the CART and Semi-CART algorithms, we employed a 10-fold cross-validation technique. This approach was executed iteratively, running CART and SEMI-CART 10 times to ensure robustness in our findings. The results obtained from these iterations were then meticulously compared, providing a comprehensive evaluation of the algorithms' performance and efficacy. In Fig. 12, The analysis reveals that across all datasets, the Semi-CART algorithm either outperformed or matched the performance of the CART method while

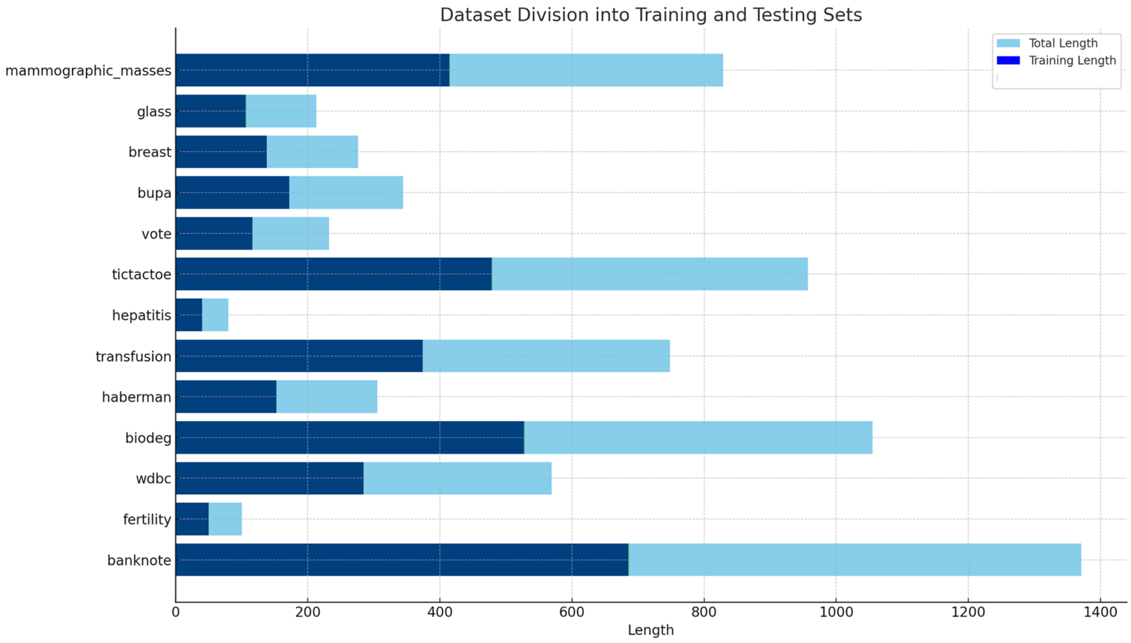


Fig. 10 Total, train, test

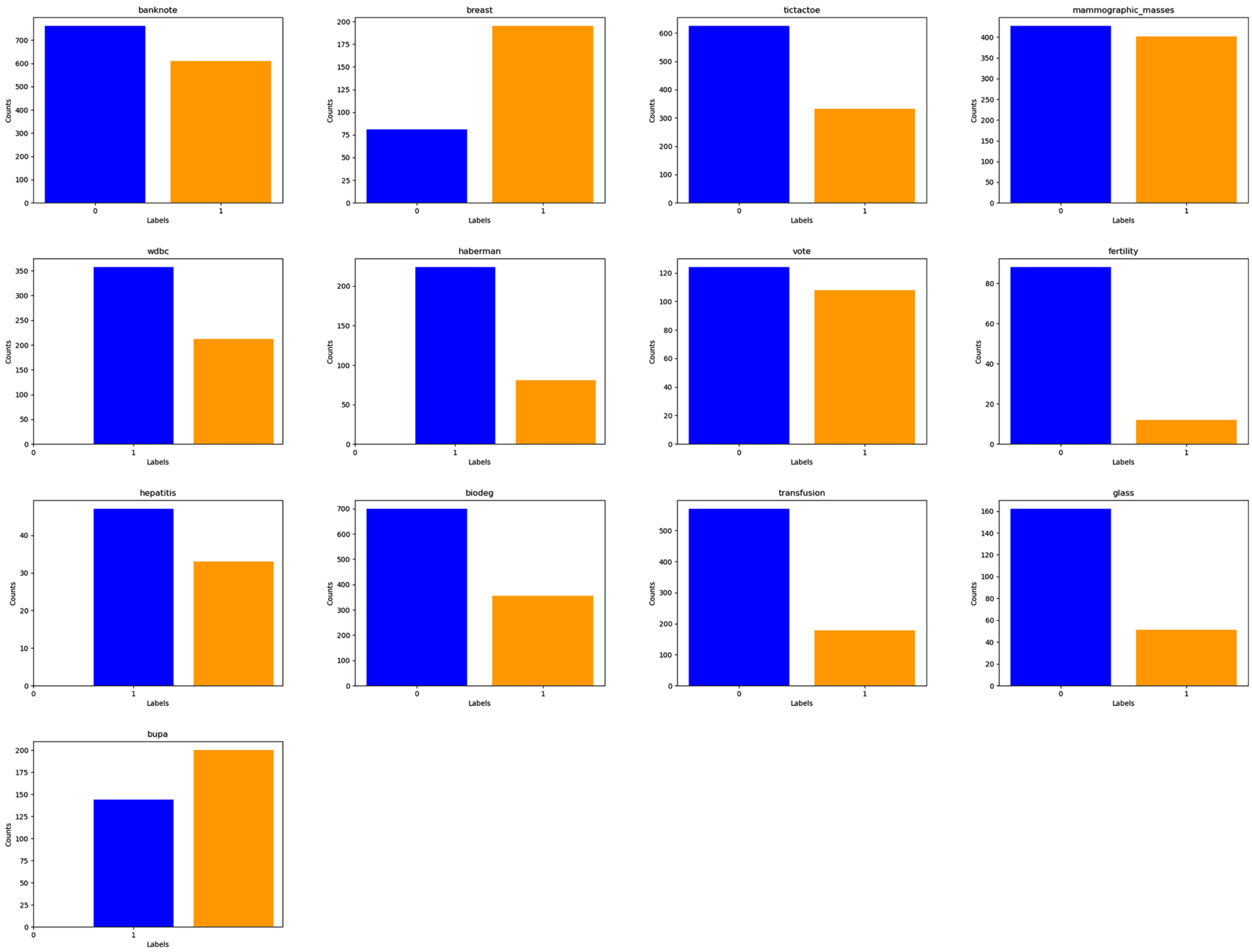


Fig. 11 Labels ratio

requiring a smaller training set. This consistent efficacy of Semi-CART, evident in our results, underscores its potential advantages in terms of efficiency and effectiveness in various data contexts. The accuracy metric is a crucial focal point within our comparative analysis, albeit representing only a single facet of the evaluation between the CART and Semi-CART algorithms. In order to offer a more comprehensive assessment, we have augmented our evaluation with additional metrics, including precision, recall, and the F1 score, as illustrated in Fig. 13. These collectively provide a multifaceted perspective on the performance of the algorithms, enabling a nuanced comparison across various dimensions of effectiveness. Subsequent sections will expound upon these comparative findings, shedding light on the strengths and limitations of each algorithm within the context of our study.

To ensure a fair and controlled environment for comparing the CART and Semi-CART algorithms, we conducted the analysis iteratively, executing the comparison ten times. This approach allowed us to calculate the maximum, minimum, average, and standard deviation for key performance metrics: accuracy, precision, recall, and F1 score. This rigorous methodology underscores the reliability and depth of our comparative analysis. In each iteration of our analysis, we meticulously tracked the accuracy, precision, recall, and F1 score values. These metrics were then mentioned in Tables 3 and 4, providing a dynamic and detailed portrayal of the algorithms' performance across successive runs. This visualization facilitates a deeper understanding of the effectiveness of the consistency and variability in the algorithms. Our investigation delves into the algorithm's dynamic behavior, focusing on its statistical metrics as they evolve throughout the iterations.

Table 2 Performance metrics for CART and SEMI_CART

Dataset	Method	Max accuracy	Max precision	Max recall	Max F1
Biodeg	CART	0.99714	0.995	0.998	0.996
	SEMI_CART	0.99714	0.998	0.998	0.996
Transfusion	CART	0.77027	0.536	0.354	0.409
	SEMI_CART	0.78784	0.7	0.394	0.436
Glass	CART	0.89524	0.801	0.88	0.789
	SEMI_CART	0.92381	0.857	0.893	0.853
Habermanm	CART	0.74333	0.794	0.899	0.829
	SEMI_CART	0.75333	0.802	0.996	0.852
Fertility	CART	0.83	0.133	0.2	0.117
	SEMI_CART	0.87	0.25	0.35	0.267
Bupa	CART	0.74118	0.701	0.706	0.685
	SEMI_CART	0.75	0.901	0.706	0.689
Breast	CART	0.9963	0.995	1.0	0.998
	SEMI_CART	0.9963	0.996	1.0	0.998
Bank_note	CART	0.98321	0.981	0.986	0.981
	SEMI_CART	0.99051	0.987	0.995	0.99
Wdbc	CART	0.93393	0.948	0.956	0.949
	SEMI_CART	0.95714	0.961	0.974	0.966
Vote	CART	0.96087	0.954	0.968	0.952
	SEMI_CART	0.97826	0.968	0.993	0.978
Tictactoe	CART	0.99895	1.0	0.997	0.999
	SEMI_CART	0.99895	1.0	0.998	0.999
MM	CART	0.79878	0.807	0.794	0.789
	SEMI_CART	0.82683	0.886	0.972	0.818
Hepatitis	CART	0.9625	0.948	1.0	0.969
	SEMI_CART	0.9625	0.957	1.0	0.976

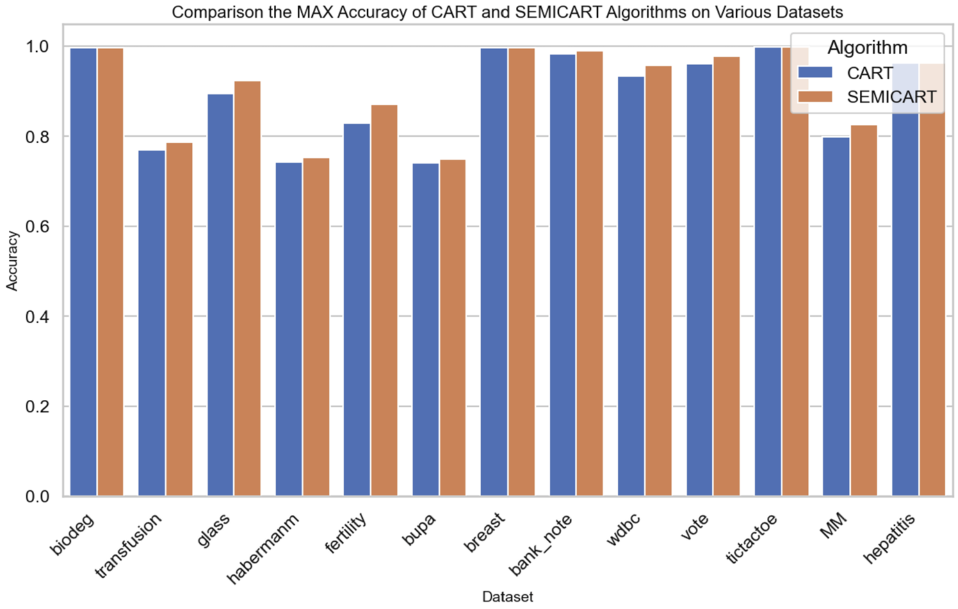


Fig. 12 Accuracy of cart vs. semi-CART

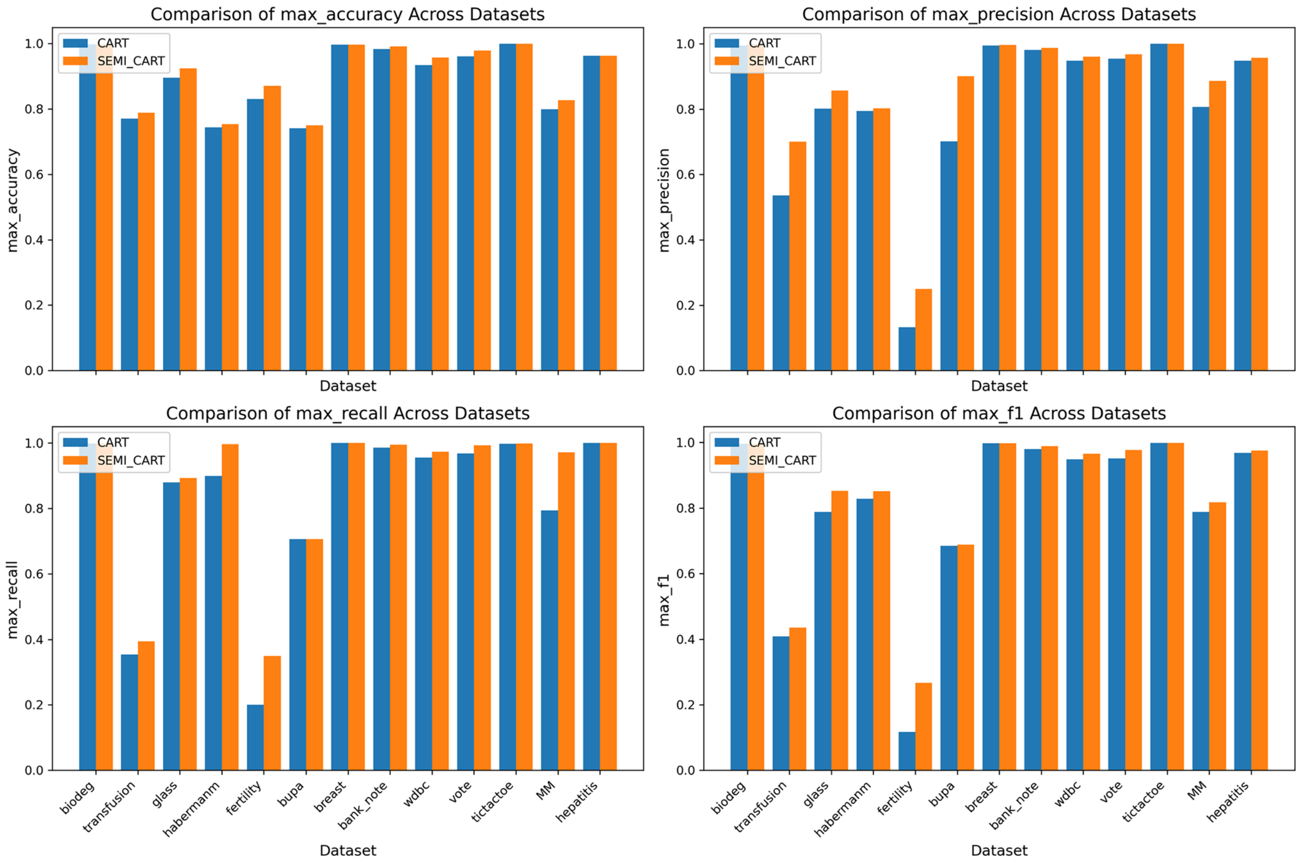


Fig. 13 Accuracy, precision, recall, F1

5 Future works

The weighting algorithm uses Euclidean distance measuring and uses all feature values to measure the distance of the train and test data. If this distance measurement owned the importance of each feature, it could result in higher speed and more accuracy. It is recommended to check the distribution of that specific feature in test and train data To find each feature's importance. The other important thing in The weighting algorithm is the algorithm's time complexity, and we propose to Shard the data and parallelize the algorithm to reduce time complexity. Another important thing is building a new boosting algorithm like GBM or XGBoost with The weighting algorithm and Semi-CART, but we should assign The weighting algorithm weights in each step of the learning process.

Table 3 Dataset statistics for CART and semi-CART performance (first seven datasets)

Dataset	Metric	CART				Semi-CART			
		Min	Max	Avg	Std	Min	Max	Avg	Std
Bank_note	Accuracy	0.9759	0.9832	0.9804	0.00206	0.9759	0.9905	0.9818	0.0434
	Precision	0.961	0.981	0.973	0.00542	0.968	0.987	0.9759	0.0629
	Recall	0.98	0.986	0.9838	0.00215	0.98	0.995	0.9848	0.1004
	F1-Score	0.973	0.981	0.9781	0.00238	0.973	0.99	0.9791	0.0608
Biodeg	Accuracy	0.9952	0.9971	0.9968	0.000664	0.9952	0.9971	0.9967	0.0264
	Precision	0.991	0.995	0.9935	0.00118	0.991	0.998	0.9932	0.0359
	Recall	0.994	0.998	0.9966	0.00117	0.994	0.998	0.9966	0.0437
	F1-Score	0.993	0.996	0.9951	0.0011	0.993	0.996	0.9949	0.0399
Breast	Accuracy	0.9926	0.9963	0.9959	0.00117	0.9963	0.9963	0.9963	0.0018
	Precision	0.99	0.995	0.9935	0.00151	0.994	0.996	0.9946	0.0021
	Recall	1.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0
	F1-Score	0.995	0.998	0.9968	0.000789	0.997	0.998	0.9973	0.0012
Bupa	Accuracy	0.6588	0.7412	0.6982	0.02554	0.6588	0.75	0.7020	0.0385
	Precision	0.608	0.701	0.6481	0.02877	0.608	0.901	0.6522	0.1356
	Recall	0.541	0.706	0.6401	0.05241	0.541	0.706	0.6464	0.1347
	F1-Score	0.562	0.685	0.6328	0.03786	0.562	0.689	0.6376	0.0866
Fertility	Accuracy	0.74	0.83	0.78	0.0287	0.8	0.87	0.8338	0.0325
	Precision	0.0	0.133	0.0464	0.0423	0.0	0.25	0.0869	0.0845
	Recall	0.0	0.2	0.0649	0.0655	0.0	0.35	0.0791	0.0949
	F1-Score	0.0	0.117	0.0448	0.0354	0.0	0.267	0.0775	0.0781
Glass	Accuracy	0.8524	0.8952	0.8805	0.01513	0.7486	0.7878	0.7671	0.0125
	Precision	0.667	0.801	0.732	0.04279	0.426	0.7	0.5135	0.0854
	Recall	0.664	0.88	0.7789	0.07736	0.29	0.394	0.3271	0.1145
	F1-Score	0.676	0.789	0.7346	0.04027	0.349	0.436	0.3803	0.1323
Haberman	Accuracy	0.6867	0.7433	0.712	0.0201	0.6933	0.7533	0.7265	0.0238
	Precision	0.761	0.794	0.7751	0.01118	0.759	0.802	0.7722	0.0196
	Recall	0.825	0.899	0.8593	0.0244	0.863	0.996	0.9189	0.0497
	F1-Score	0.796	0.829	0.811	0.01402	0.799	0.852	0.8282	0.0193

Table 4 Dataset statistics for CART and Semi-CART performance (Next Six Datasets)

Dataset	Metric	CART				Semi-CART			
		Min	Max	Avg	Std	Min	Max	Avg	Std
Hepatitis	Accuracy	0.9625	0.9625	0.9625	1.17e−16	0.9625	0.9625	0.9625	0.0279
	Precision	0.922	0.948	0.9362	0.00989	0.935	0.957	0.9427	0.0183
	Recall	1.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0376
	F1-Score	0.955	0.969	0.9629	0.0059	0.958	0.976	0.9666	0.0259
Mammographic_masses	Accuracy	0.7634	0.7988	0.7845	0.01339	0.7683	0.8268	0.7973	0.0937
	Precision	0.76	0.807	0.7845	0.01474	0.76	0.886	0.8107	0.1048
	Recall	0.747	0.794	0.7663	0.01632	0.747	0.972	0.8016	0.1988
	F1-Score	0.755	0.789	0.7727	0.013	0.767	0.818	0.7875	0.1807
Tictactoe	Accuracy	0.9979	0.9989	0.9988	0.000335	0.9990	0.9990	0.9990	0.0006
	Precision	1.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0
	Recall	0.995	0.997	0.9966	0.000699	0.997	0.998	0.9971	0.0023
	F1-Score	0.997	0.999	0.9984	0.000699	0.998	0.999	0.9985	0.0012
Transfusion	Accuracy	0.7405	0.7703	0.7558	0.00798	0.7486	0.7878	0.7671	0.0125
	Precision	0.426	0.536	0.4877	0.02934	0.426	0.7	0.5135	0.0854
	Recall	0.29	0.354	0.33	0.0202	0.29	0.394	0.3271	0.1145
	F1-Score	0.349	0.409	0.3824	0.01952	0.349	0.436	0.3803	0.1323
Vote	Accuracy	0.9435	0.9609	0.9522	0.00615	0.9478	0.9783	0.9609	0.0132
	Precision	0.934	0.954	0.9421	0.00635	0.935	0.968	0.9492	0.0124
	Recall	0.933	0.968	0.9525	0.01196	0.943	0.993	0.9708	0.0206
	F1-Score	0.94	0.978	0.9571	0.0146	0.94	0.978	0.9571	0.0146
Wdbc	Accuracy	0.9107	0.9339	0.9213	0.00721	0.9143	0.9571	0.9307	0.0133
	Precision	0.913	0.948	0.9309	0.01183	0.926	0.961	0.9419	0.0129
	Recall	0.926	0.956	0.9457	0.00879	0.935	0.974	0.9517	0.0184
	F1-Score	0.929	0.949	0.9371	0.00601	0.932	0.966	0.9442	0.0113

6 Conclusion

In This paper, We propose a novel Semi-CART algorithm to construct decision trees that can achieve higher accuracy than the base algorithm, CART. We also provide a detailed explanation of how the Weighting algorithm works and how it incorporates weights into the training data. We illustrate how the Weighting algorithm removes ineffective training data and improves the accuracy of classification and regression problems. Furthermore, we introduce a new formula of GINI impurity with weights to select the best candidate for splitting. This new formula replaces “p” with “w/S,” where “w” is the weight of the training row and “S” is the sum of all train data weights. We present the implementation of Semi-CART with pseudo-code and demonstrate its effectiveness through experiments.

Additionally, we discuss removing useless data and comparing CART with Semi-CART. The Weighting algorithm is appropriate for non-streaming processes, mainly when the extensive test data contains noisy training data. Finally, we highlight that CART is the fundamental concept in state-of-the-art algorithms like XGBoost and LightGBM.

Data availability In this study, we utilized datasets from the UCI Machine Learning Repository. This prominent online resource offers a comprehensive collection of datasets for empirical analysis in machine learning and data science. This repository, managed by the Center for Machine Learning and Intelligent Systems at the University of California, Irvine, provides well-prepared data facilitating robust statistical testing and validation of innovative algorithms. For our research, specific datasets were downloaded and employed, including Banknote, Fertility, WDBC (Wisconsin et al.), Biodegradability, Haberman’s Survival, Blood Transfusion Service Center, Hepatitis, Tic-Tac-Toe, Congressional Voting Records, BUPA Liver Disorders, Breast Cancer Wisconsin (Original), Glass Identification, and Mammographic Masses. Each dataset was chosen for its relevance to the specific machine-learning tasks, ranging from classification challenges to predictive modeling. The wide range and varied nature of these datasets render the UCI Machine Learning Repository an essential resource for conducting comprehensive empirical research across different areas of machine learning. Further information about the datasets and access to the repository can be found at the UCI Machine Learning Repository website. https://archive.ics.uci.edu/.

References

1. Wickramarachchi DC, Robertson BL, Reale M, Price CJ, Brown J (2016) Hhcart: an oblique decision tree. Comput Stat Data Anal 96:12–23
2. Chary S, Rama B (2017) A survey on comparative analysis of decision tree algorithms in data mining. In: International Conference On Innovative Applications In Engineering and Information Technology (ICIAEIT-2017), vol. **3**, pp. 91–95
3. Li X, Sun Q, Liu Y, Zhou Q, Zheng S, Chua T-S, Schiele B (2019) Learning to self-train for semi-supervised few-shot classification. Advances in neural information processing systems 32
4. Chen M, Du Y, Zhang Y, Qian S, Wang C (2022) Semi-supervised learning with multi-head co-training. Proc AAAI Conf Artif Intell 36:6278–6286
5. Cascante-Bonilla P, Tan F, Qi Y, Ordonez V (2021) Curriculum labeling: revisiting pseudo-labeling for semi-supervised learning. Proc AAAI Conf Artif Intell 35:6912–6920
6. Iscen A, Tolias G, Avrithis Y, Chum O (2019) Label propagation for deep semi-supervised learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5070–5079
7. Chen B, Jiang J, Wang X, Wan P, Wang J, Long M (2022) Debiased self-training for semi-supervised learning. In: Advances in Neural Information Processing Systems
8. Rizve MN, Duarte K, Rawat YS, Shah M (2021) In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning. arXiv preprint arXiv:2101.06329
9. Hssina B, Merbouha A, Ezzikouri H, Erritali M (2014) A comparative study of decision tree id3 and c4. 5. Int J Adv Comput Sci Appl 4(2):13–19
10. Morgan JN, Sonquist JA (1963) Problems in the analysis of survey data, and a proposal. J Am Stat Assoc 58:415–434
11. Quinlan JR (1986) Induction of decision trees. Mach Learn 1:81–106
12. Denison DG, Mallick BK, Smith AF (1998) A bayesian cart algorithm. Biometrika 85(2):363–377
13. Breiman L, Friedman J, Olshen R, Stone C (1984) Cart. Classification and regression trees
14. Singh S, Gupta P (2014) Comparative study id3, cart and c4. 5 decision tree algorithm: a survey. Int J Adv Inform Sci Technol (IJAIST) 27(27):97–103
15. Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, Ye Q, Liu T-Y (2017) Lightgbm: A highly efficient gradient boosting decision tree. Advances in neural information processing systems 30
16. Friedman JH (2001) Greedy function approximation: a gradient boosting machine. Ann stat 29:1189–1232
17. Chen T, Guestrin C (2016) Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining, pp. 785–794
18. Zhu X, Ghahramani Z, Lafferty JD (2003) Semi-supervised learning using gaussian fields and harmonic functions. In: Proceedings of the 20th International Conference on Machine Learning (ICML-03), pp. 912–919
19. Tarvainen A, Valpola H (2017) Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. Advances in neural information processing systems **30**
20. Kahn J, Lee A, Hannun A (2020) Self-training for end-to-end speech recognition. In: ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 7084–7088 . IEEE
21. Wang W, Zhou Z-H (2007) Analyzing co-training style algorithms. In: Machine Learning: ECML 2007: 18th European Conference on Machine Learning, Warsaw, Poland, September 17-21, 2007. Proceedings 18, pp. 454–465. Springer
22. Zhou D, Bousquet O, Lal T, Weston J, Schölkopf B (2003) Learning with local and global consistency. Advances in neural information processing systems **16**
23. Song Z, Yang X, Xu Z, King I (2022) Graph-based semi-supervised learning: A comprehensive review. IEEE Trans Neural Netw Learn Syst 34:8174–8194
24. Tanha J, Van Someren M, Afsarmanesh H (2017) Semi-supervised self-training for decision tree classifiers. Int J Mach Learn Cybern 8:355–370
25. Chen X, Zhu C-C, Yin J (2019) Ensemble of decision tree reveals potential mirna-disease associations. PLoS Comput Biol 15(7):1007209
26. Kim K (2016) A hybrid classification algorithm by subspace partitioning through semi-supervised decision tree. Pattern Recogn 60:157–163
27. Li B, Wang J, Yang Z, Yi J, Nie F (2023) Fast semi-supervised self-training algorithm based on data editing. Inform Sci 626:293–314
28. Zharmagambetov A, Carreira-Perpiñán MÁ (2022) Semi-supervised learning with decision trees: Graph laplacian tree alternating optimization. Adv Neural Inform Process Syst 35:2392–2405

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.